A Hardware-Efficient AEAD Stream Cipher Based on a Hybrid Nonlinear Feedback Register Structure

Arman Allahverdi*

Vincent John Mooney III*,^

*School of Electrical and Computer Engineering

^School of Computer Science

Georgia Institute of Technology

Atlanta, Georgia, USA

Outline

- Introduction and Motivation
- Background
- Design Overview
- Statistical Testing and Security Analysis
- Hardware Implementation Results
- Scalability
- Future Work
- References

Outline

- Introduction and Motivation
- Background
- Design Overview
- Statistical Testing and Security Analysis
- Hardware Implementation Results
- Scalability
- Future Work
- References

Introduction and Motivation: FSRs and Crypto

- **Lightweight cryptography** is crucial for hardware-constrained environments such as **IoT** and **embedded systems**
- **Hardware-based cryptography** gaining traction with the introduction of trusted platform modules (TPMs) on consumer workstations
- Common primitives in hardware-based cryptography include two classes of feedback registers: **linear feedback shift registers (LFSRs)** and **nonlinear feedback shift registers (NLFSRs)**
- LFSRs and NLFSRs, both inherently lightweight in hardware, can generate pseudorandom sequences with high period
- **NLFSRs are not scalable:** the largest full-period NLFSR is 24 bits [1]

Introduction and Motivation: AEAD

- **AEAD** stands for *Authenticated Encryption with Associated Data*
- In most practical situations, plaintext “associated data” ¬ such as an IP address ¬ needs to be sent together with encrypted data
- Authentication then needs to be applied simultaneously to the encrypted as well as the plaintext data so that an adversary may not have any reasonable chance to alter any of the data undetected

Introduction and Motivation: CMPRs

- This work proposes an authenticated and scalable stream cipher design based on **Composite Mersenne Product Registers (CMPRs)** [2]; the new design is named the **Hybrid Register Stream Cipher**
- CMPRs are a form of feedback register with guaranteed periodicity, nonlinearity, and scalability
- We also compare our work to other feedback register-based schemes, such as TRIVIUM [3], Espresso [4], and Grain-128AEADv2 [5]

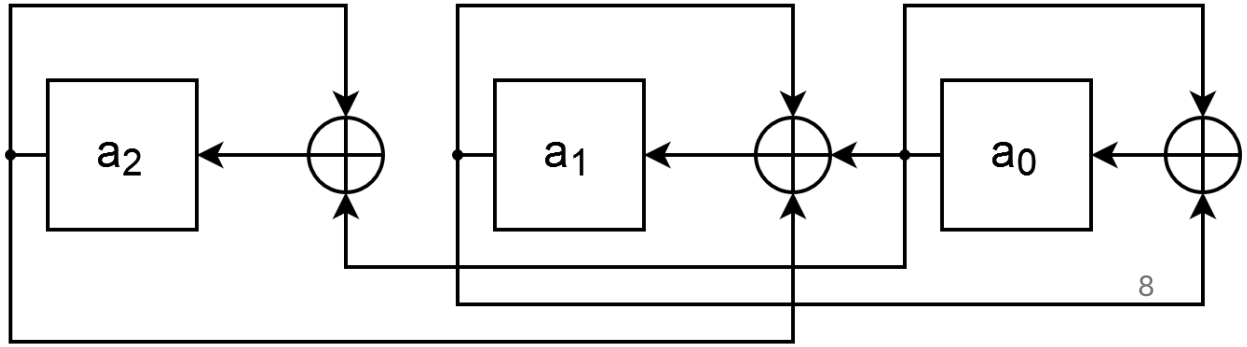
[2] D. Gordon *et al.*, "Scalable Nonlinear Sequence Generation using Composite Mersenne Product Registers," *IACR Communications in Cryptology*, vol. 1, no. 4, pp. 1-77, Jan. 2025, doi: https://doi.org/10.62056/a3tx11zn4.

Outline

- Introduction and Motivation
- Background
- Design Overview
- Statistical Testing and Security Analysis
- Hardware Implementation Results
- Scalability
- Future Work
- References

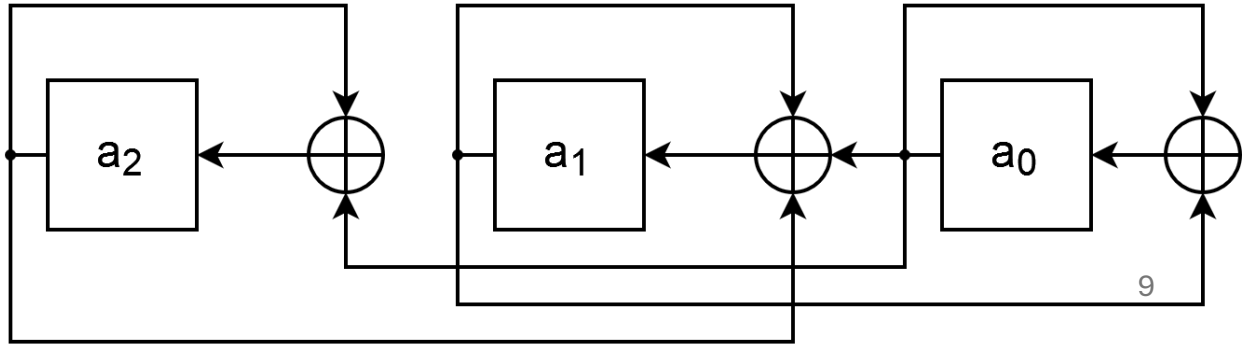
Background: Mersenne Product Registers

- Let A[t] denote the current state of a feedback register. The next state, A[t+1], is given by $A[t+1] = f(A[t])$
- f can be a linear or nonlinear function of the current state
- This work uses **Mersenne Product Registers (MPRs)** [2]
- An MPR is a feedback register with the restriction that the register size must be a Mersenne exponent, e.g., a number n such that $2^n - 1$ is prime
- For an n-bit MPR, the state update is given by $A[t+1] = U(x)A[t] \bmod P(x)$
  - U(x): update polynomial, degree $n - 1$, $U(x) \neq 0, 1$
  - P(x): feedback polynomial, degree n
- P(x) must be a **primitive polynomial** for an n-bit MPR to be full-period (period of $2^n - 1$)
- MPR example for $n = 3$ [2, Fig. 3]:
  - $U(x) = x^2 + x + 1$
  - $P(x) = x^3 + x^2 + 1$



Background: Mersenne Product Registers

- Let A[t] denote the current state of a feedback register. The next state, A[t+1], is given by A[t+1] = *f* (A[t])
- *f* can be a linear or nonlinear function of the current state
- This work uses **Mersenne Product Registers (MPRs)** [2]
- An MPR is a feedback register with the restriction that the register size must be a Mersenne exponent, e.g., a number *n* such that $2^n - 1$ is prime
- For an n-bit MPR, the state update is given by A[t+1] = U(x)A[t] mod P(x)
  - U(x): update polynomial, degree $n - 1$, $U(x) \neq 0, 1$
  - P(x): feedback polynomial, degree *n*
- P(x) must be a **primitive polynomial** for an *n*-bit MPR to be full-period (period of $2^n - 1$)
- MPR example for $n = 3$ [2, Fig. 3]:
  - $U(x) = x^2 + x + 1$
  - $P(x) = x^3 + x^2 + 1$



Background: Finite Fields Base 2

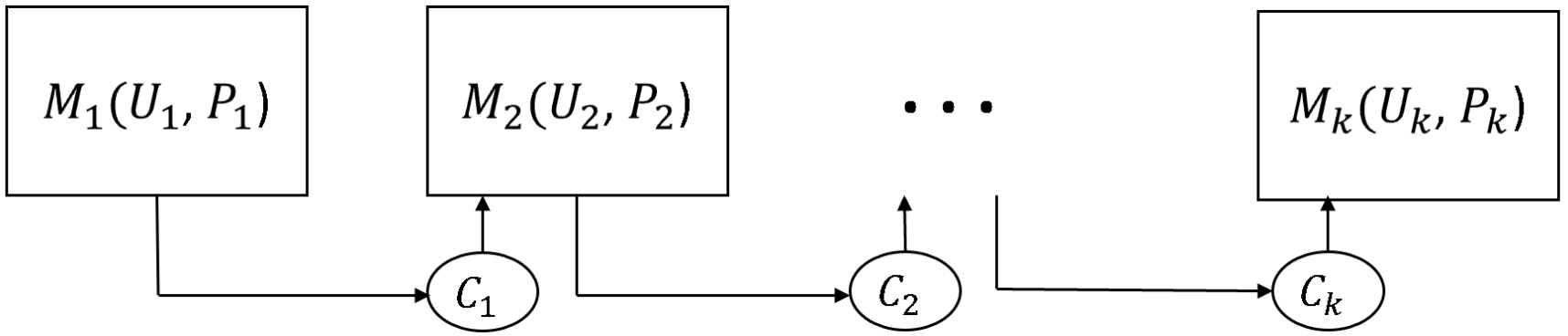
- Registers composed of *n* bits can be viewed as polynomials
- The element zero (all register bit values equal to 0) is not part of the multiplicative field, therefore the multiplication of U(x) by the current state A[t] occurs inside of a finite field (also named Galois field)
- When *n* is a Mersenne exponent, $2^n - 1$ is prime and therefore the *n*-bit MPR has two cycles: one cycle of size one (for the case of zero) and one cycle of size $2^n - 1$; this is a known result from field theory [2]
  - Note: we start our MPR-based registers with all bits equal to one or a random number
- We have not been able to find the equation $A[t+1] = U(x)A[t] \bmod P(x)$ in any work prior to [2], although it can be argued that this equation follows in a rather straightforward fashion from known field theory
- Additional proofs of exponential state size for combined MPRs are available in [2] with plenty of references to prior work and results in field theory

Background: Finite Fields Base 2

- Registers composed of *n* bits can be viewed as polynomials
- The element zero (all register bit values equal to 0) is not part of the multiplicative field, therefore the multiplication of U(x) by the current state A[t] occurs inside of a finite field (also named Galois field)
- When *n* is a Mersenne exponent, $2^n - 1$ is prime and therefore the *n*-bit MPR has two cycles: one cycle of size one (for the case of zero) and one cycle of size $2^n - 1$; this is a known result from field theory [2]
  - Note: we start our MPR-based registers with all bits equal to one or a random number
- We have not been able to find the equation A[t+1] = U(x)A[t] mod P(x) in any work prior to [2], although it can be argued that this equation follows in a rather straightforward fashion from known field theory
- Additional proofs of exponential state size for combined MPRs are available in [2] with plenty of references to prior work and results in field theory

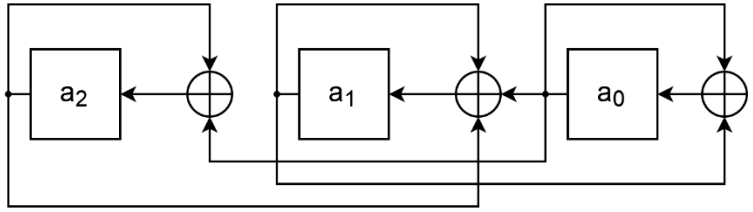
Background: CMPRs

- On their own, MPRs are linear; however, to introduce nonlinearity, several MPRs can be combined to form a Compound MPR (CMPR)
- MPRs are combined using **chaining functions**
- Chaining functions are sets of Boolean equations that determine how the state of one MPR affects the state of another MPR [2]
- To instantiate a nonlinear CMPR, the chaining functions must also be nonlinear (e.g., include multiplicative terms, i.e., AND gates)



Background: CMPR state size

- Unlike all know prior NLFSR approaches, CMPRs are straightforward to scale up or down while retaining exponential periodicity
  - U(x) can be any value other than 0 or 1
    - Note: if U(x) = x, the result is a shift register; we argue that a Product Register (PR) is a generalization of a Feedback Shift Register (FSR) [2]



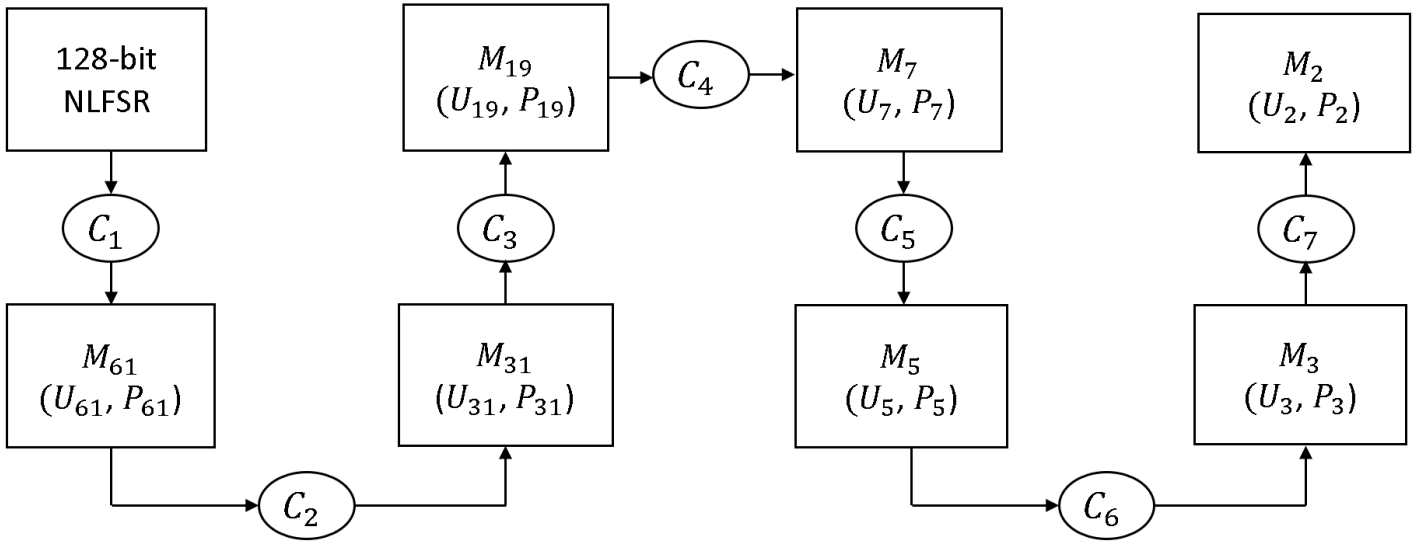
  - P(x) only need be shown to be **irreducible** in order to be primitive due to the fact that, for a polynomial of Mersenne exponent degree, irreducibility implies primitivity [2]
    - For example, $x^2 + x = x(x + 1)$ is reducible but $x^2 + x + 1$ is irreducible; there is no way to give two polynomials whose multiplication results in $x^2 + x + 1$; algorithms to deduce irreducibility are well-known
    - The fundamental issue is that if *n* is not a Mersenne exponent this implies that $2^n - 1$ is not prime, and then there exist cases where using an irreducible polynomial for P(x) does not result in full period
- Altering the size of a CMPR requires adding or removing MPRs
  - For compatibility with the periodicity and linear complexity math from [2], the CMPR construction requires the use of unique Mersenne exponents
  - => Using multiple MPRs of the same size is prohibited
- Periodicity of the CMPR is guaranteed even when scaling up or down. For an *n*-bit CMPR, the **worst-case expected period size is 0.45 * 2^n [2]**

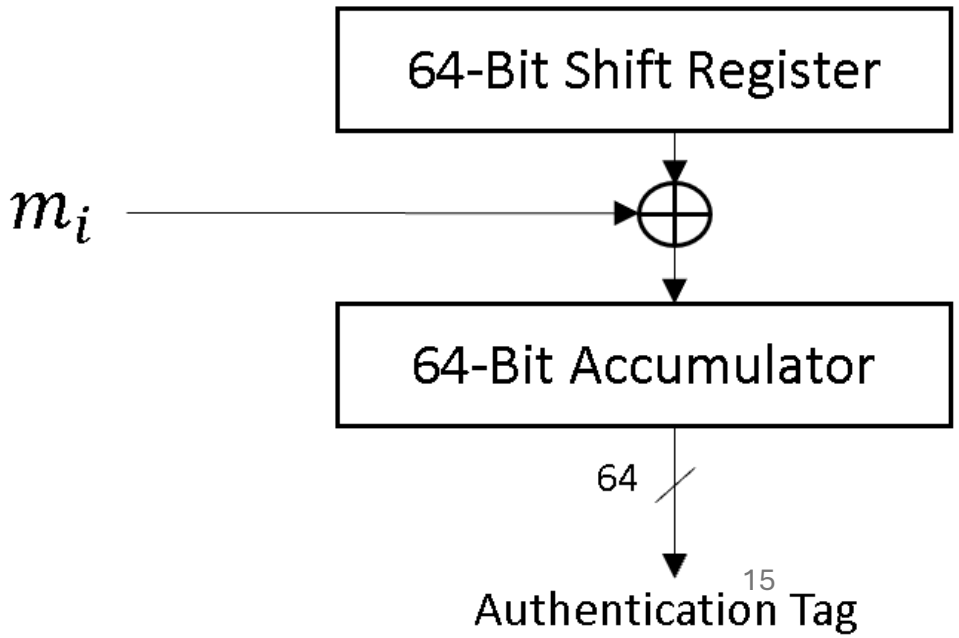
Outline

- Introduction and Motivation
- Background
- Design Overview
- Statistical Testing and Security Analysis
- Hardware Implementation Results
- Scalability
- Future Work
- References

Design Overview: Architecture

- **Architecture:**
  - **256-bit hybrid register:** 128-bit NLFSR from Grain-128AEADv2 [5] connected to a 128-bit CMPR
- **Main Components:**
  - **Keystream Generator:** 256-bit hybrid register
  - **Authentication Generator:** 64-bit accumulator, 64-bit shift register
- **Note that the Chaining Period Theorem (Theorem 1 of [2]) still applies**





Design Overview: Architecture

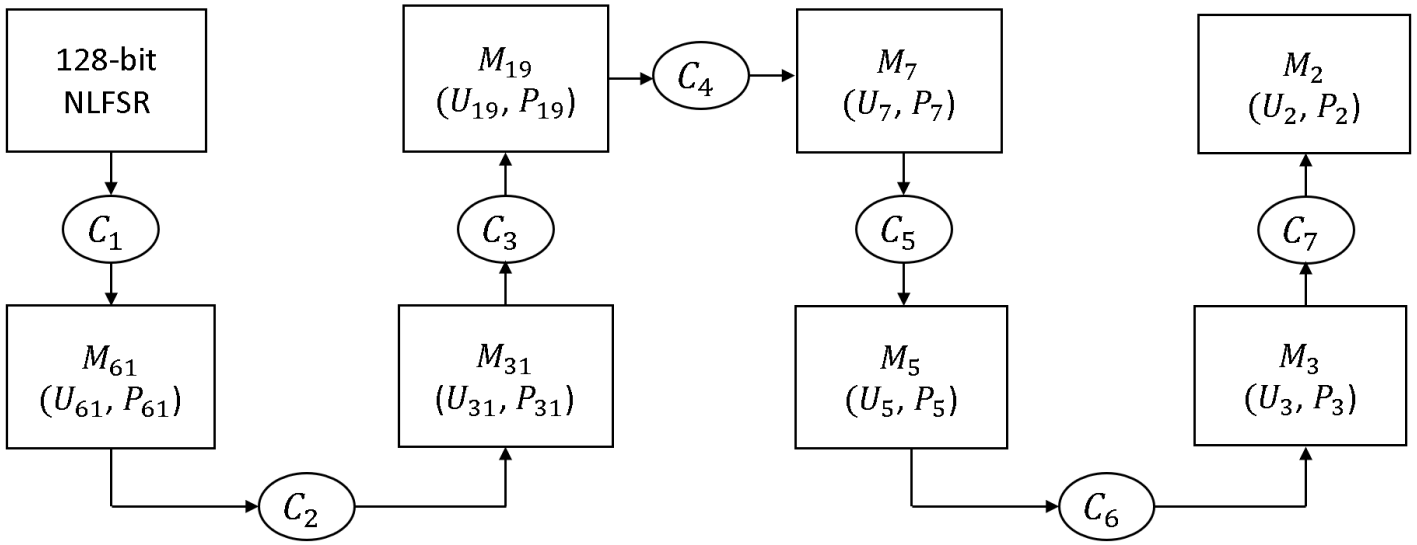
MPR Size, i	Update Polynomial, $U_i(x)$	Feedback Polynomial, $P_i(x)$
61 bits	$U_{61}(x) = x^3 + 1$	$P_{61}(x) = x^{61} + x^{44} + x^{19} + x^{15} + 1$
31 bits	$U_{31}(x) = x^2 + 1$	$P_{31}(x) = x^{31} + x^3 + x^2 + x + 1$
19 bits	$U_{19}(x) = x^4 + x^2$	$P_{19}(x) = x^{19} + x^5 + x^2 + x + 1$
7 bits	$U_7(x) = x^4 + x$	$P_7(x) = x^7 + x + 1$
5 bits	$U_5(x) = x^3$	$P_5(x) = x^5 + x + 1$
3 bits	$U_3(x) = x^2$	$P_3(x) = x^3 + x + 1$
2 bits	$U_2(x) = x + 1$	$P_2(x) = x^2 + x + 1$

MPR Specifications for the 128-bit CMPR used in the Hybrid Register Construction

- For a given MPR, the choice of U(x) and P(x) is flexible
- For U(x), any U(x) ≠ 0,1 is valid
- For P(x), the polynomial should be irreducible to ensure the MPR has full period

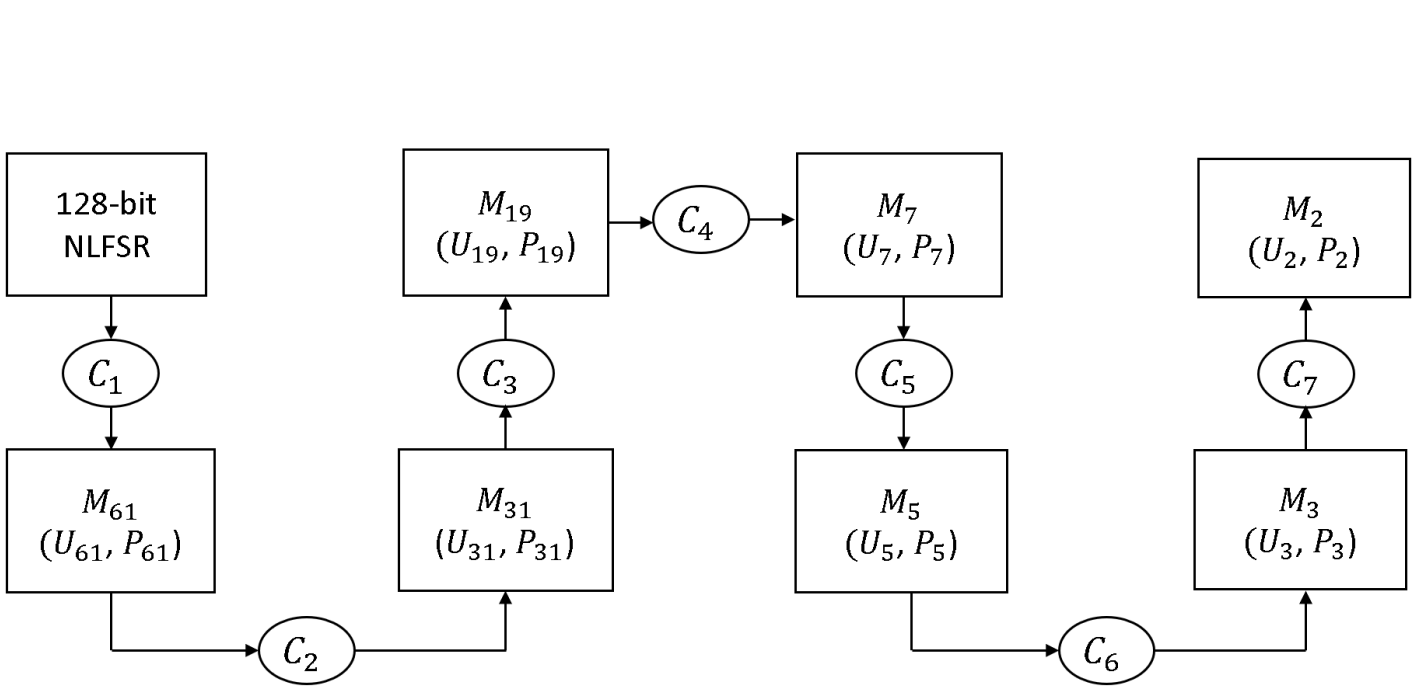
Design Overview: Architecture

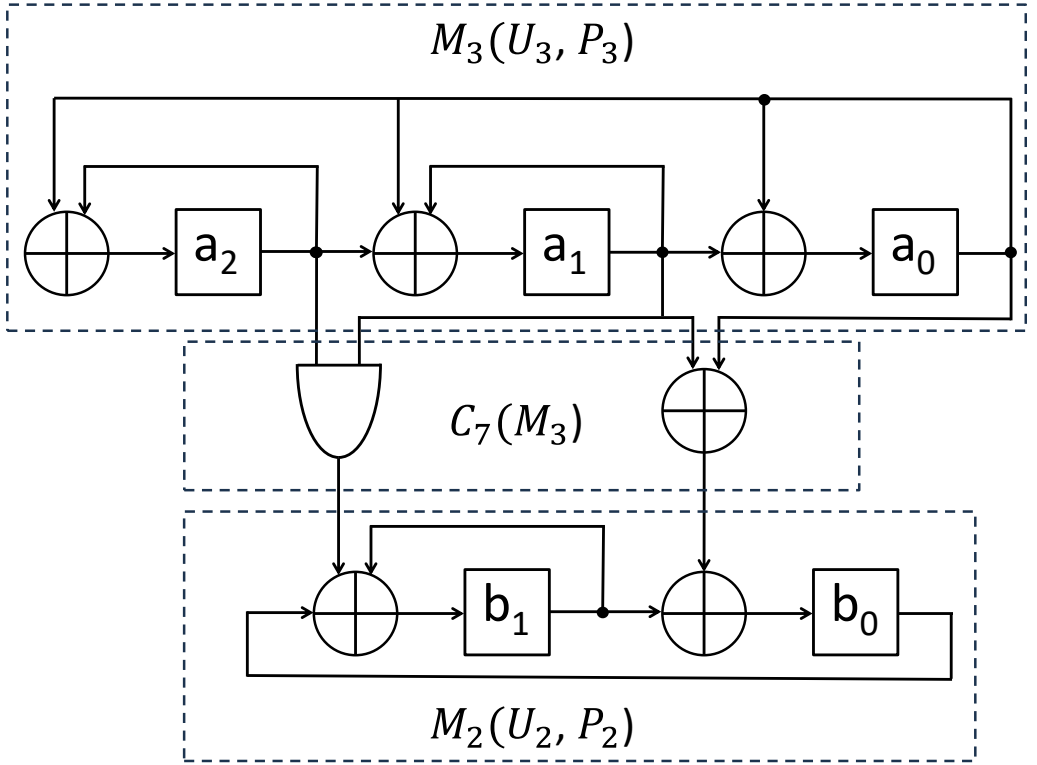
- **Chaining Functions:**
  - Depicted below, (C_1, …, C_7) are sets of nonlinear Boolean equations constructed with XOR gates of at most 4 inputs along with AND gates of at most 4 inputs
  - Our design uses a **chaining density** of 40%, meaning all MPRs in the 128-bit CMPR receive chaining functions to approximately 40% of their bits



Design Overview: Architecture

- **Chaining Functions:**
  - Depicted below, (C_1, …, C_7) are sets of nonlinear Boolean equations constructed with XOR gates of at most 4 inputs along with AND gates of at most 4 inputs
  - Our design uses a **chaining density** of 40%, meaning all MPRs in the 128-bit CMPR receive chaining functions to approximately 40% of their bits





Design Overview: Architecture

- **Parameter Sizes:**
  - **(Input) Key:** 128 bits -> equal to initial state of the NLFSR
  - **(Input) Initialization Vector (IV):** 96 bits -> equal to initial state of the 96 MSBs of the CMPR
  - **(Input) Plaintext Message:** Arbitrary length
  - **(Output) Ciphertext:** Equal to message length
  - **(Output) Authentication Tag:** 64 bits
- **Restrictions:**
  - A maximum of 2^{81} keystream bits can be generated from a single (key, IV) pair
  - This restriction is borrowed from Grain-128AEADv2 [5], since our work uses the same NLFSR

Design Overview: Operation

- **Initialization:**
  - **Keystream Generator Initialization:** Hybrid register is clocked 128 times to sufficiently randomize its state
  - **Authentication Generator Initialization:** Hybrid register is clocked an additional 128 times; accumulator and shift register are populated with keystream generator bits
- **Keystream and Tag Generation:**
  - After initialization, keystream bits are generated by computing the XOR of bits 0, 3, and 7 of the hybrid register for each post-initialization clock cycle
  - For even clock cycles, the keystream bit is used for encryption
  - For odd clock cycles, the keystream bit is used for authentication

Outline

- Introduction and Motivation
- Background
- Design Overview
- Statistical Testing and Security Analysis
- Hardware Implementation Results
- Scalability
- Future Work
- References

Statistical Testing and Security Analysis

- **Statistical Testing:**
  - We perform statistical testing of ciphertexts on two fronts: (i) bit contribution tests [10] to **verify random relationship between stream cipher inputs and outputs**, and (ii) NIST statistical test suite [11] to **verify ciphertexts do not exhibit hidden statistical weaknesses**
- **Security Analysis:**
  - Security analysis of CMPRs [2] indicates that **CMPRs are resilient against algebraic attacks, cube attacks (and distinguishers), correlation attacks, and linear cryptanalysi**s, provided the CMPR is well-designed
  - A "well-designed" CMPR should: (i) incorporate nonlinear chaining functions, (ii) ensure that chaining functions do not skip over any MPRs (chaining functions from a given MPR should connect to the next smallest MPR in the design), and (iii) ensure that at least two MPRs not containing tweakable inputs (key, IV) are present before the MPR from which the keystream is extracted
  - Our hybrid register design follows aforementioned design guidelines (i)-(iii)

Outline

- Introduction and Motivation
- Background
- Design Overview
- Statistical Testing and Security Analysis
- Hardware Implementation Results
- Scalability
- Future Work
- References

Hardware Implementation Results

- ASIC hardware implementation performed using NCSU FreePDK45 process (45nm) [6]
- **Synthesis:** Synopsys DesignCompiler [7]
- **Auto Place & Route (APR):** Cadence Innovus [8]
- **Layout:** Cadence Virtuoso [9]
- Hardware implementation and comparison performed for Hybrid Register Stream Cipher, TRIVIUM, Espresso, and Grain-128AEADv2

Hardware Implementation Results

Design	Authentication	Security	Latency* (Clock Cycles)
Hybrid Register Stream Cipher	Yes	128 bits (i.e., 2^{128})	128
Grain-128AEADv2	Yes	128 bits	512
Espresso	No	128 bits	256
TRIVIUM	No	80 bits	1152

*Latency: # of clock cycles prior to generation of encrypted bits

- We perform comparisons to authentication-capable and non-authenticating designs separately
- To compare the hybrid register stream cipher to non-authenticating designs, we do not include the authentication generator as part of the hardware implementation process

Hardware Implementation Results

Design	Area (μm^2)	Power (mW)	PDP* (pJ)
Hybrid Register Stream Cipher	**4402**	**0.981**	**3.27**
Espresso	5894	1.949	6.49
TRIVIUM	5627	3.0226	10.08

Hardware Implementation Comparison (Non-Authenticating Designs)

Design	Area (μm^2)	Power (mW)	PDP* (pJ)
Hybrid Register Stream Cipher	**5650**	**1.313**	**4.37**
Grain-128AEADv2	6401	2.936	9.78

Hardware Implementation Comparison (Authentication-Capable Designs)

*PDP: power-delay product, equal to power times clock period

Hardware Implementation Results: Summary

- The Hybrid Register Stream cipher utilizes lower area and consumes less energy than TRIVIUM, Espresso, and Grain-128AEADv2
- 11.7% lower area and 55% lower power than Grain-128AEADv2
- 21.8% lower area and 67.5% lower power than TRIVIUM
- 25.3% lower area and 49.7% lower power than Espresso

Outline

- Introduction and Motivation
- Background
- Design Overview
- Statistical Testing and Security Analysis
- Hardware Implementation Results
- Scalability
- Future Work
- References

Outline

- Introduction and Motivation
- Background
- Design Overview
- Statistical Testing and Security Analysis
- Hardware Implementation Results
- Scalability
- Future Work
- References

Scalability

- The Hybrid Register Stream cipher can be designed with more bit if (i) a larger key is desired, (ii) a larger IV is desired, or (iii) both a larger key and a larger IV are desired
- Many, many bit sizes can be designed with Mersenne exponents
- To showcase scalability, we designed two additional sizes of Hybrid Register Stream Ciphers: 320 bits and 384 bits

CMPR Specification for 384-bit Hybrid Register Stream Cipher

MPR Size, i	Update Polynomial, $U_i(x)$	Feedback Polynomial, $P_i(x)$
107 bits	$U_{107}(x) = x^7$	$P_{107}(x) = x^{107} + x^{59} + x^{54} + x^{39} + 1$
89 bits	$U_{89}(x) = x^7 + x^6$	$P_{89}(x) = x^{89} + x^{38} + 1$
31 bits	$U_{31}(x) = x^3 + 1$	$P_{31}(x) = x^{31} + x^3 + 1$
17 bits	$U_{17}(x) = x^4$	$P_{17}(x) = x^{17} + x^3 + 1$
7 bits	$U_7(x) = x^5 + x$	$P_7(x) = x^7 + x + 1$
3 bits	$U_3(x) = x^2$	$P_3(x) = x^3 + x + 1$
2 bits	$U_2(x) = x + 1$	$P_2(x) = x^2 + x + 1$

MPR Specifications for the 256-bit CMPR used in the 384-bit Hybrid Register Construction

TABLE VIII

Non-AEAD ASIC Hardware Implementation Results for the Larger Hybrid Register Stream Cipher Variants

Design	**Area (μm^2)**	**Power (mW)**	**PDP (pJ)**
320-bit Hybrid Register	6453	1.435	4.78
384-bit Hybrid Register	8367	1.511	5.03

Outline

- Introduction and Motivation
- Background
- Design Overview
- Statistical Testing and Security Analysis
- Hardware Implementation Results
- Scalability
- Future Work
- References

Outline

- Introduction and Motivation
- Background
- Design Overview
- Statistical Testing and Security Analysis
- Hardware Implementation Results
- Scalability
- Future Work
- References

Future Work

- Possible avenues for future work include the following:
  - Evaluating use of the hybrid register in other types of cryptosystems such as hash functions, message authentication codes (MACs), and block ciphers
  - Side channel-secure implementations of the hybrid register primitive

Outline

- Introduction and Motivation
- Background
- Design Overview
- Statistical Testing and Security Analysis
- Hardware Implementation Results
- Future Work
- References

References

[1] E. Dubrova, “A List of Maximum Period NLFSRs.” Accessed: Jul. 03, 2025. [Online]. Available: https://eprint.iacr.org/2012/166.pdf

[2] D. Gordon *et al.*, “Scalable Nonlinear Sequence Generation using Composite Mersenne Product Registers,” *IACR Communications in Cryptology*, vol. 1, no. 4, Jan. 2025, doi: https://doi.org/10.62056/a3tx11zn4.

[3] C. De Canniere, “Trivium: A stream cipher construction inspired by ` block cipher design principles,” in Lecture Notes in Computer Science, vol. 4377, A. Biryukov, Ed. Berlin, Germany: Springer, 2006, pp. 171– 186, doi: 10.1007/11836810 13.

[4] E. Dubrova and M. Hell, “Espresso: A stream cipher for 5G wireless communication systems,” Cryptogr. Commun., vol. 9, no. 2, pp. 273–289, Dec. 2015, doi: 10.1007/s12095-015-0173-2.

[5] M. Hell et al., “Grain-128AEADv2 – A lightweight AEAD stream cipher.” [Online]. Available: https://csrc.nist.gov/CSRC/media/Proje cts/lightweight-cryptography/documents/finalist-round/updated-spec-d oc/grain-128aead-spec-final.pdf.

[6] NC State EDA, “FreePDK45.” [Online]. Available: https://eda.ncsu.edu /freepdk/freepdk45/.

[7] Synopsys, Design Compiler. [Online]. Available: https://www.synopsys .com/implementation-and-signoff/rtl-synthesis-test/dc-ultra.html.

[8] Cadence, Innovus Implementation System, 2020. [Online]. Available: ht tps://www.cadence.com/en\ US/home/tools/digital-design-and-signoff/ soc-implementation-and-floorplanning/innovus-implementation-system.html.

[9] Cadence, Virtuoso Layout Suite. [Online]. Available: https://www.cade nce.com/en\ US/home/tools/custom-ic-analog-rf-design/layout-design/ virtuoso-layout-suite.html.

[10] S. Pugh, S. Raunak, D. Kuhn, and R. Kacker, “Systematic Testing of Lightweight Cryptographic Implementations [Extended Abstract].” Available: https://csrc.nist.gov/CSRC/media/Events/lightweight-cryptography-workshop-2019/documents/papers/systematic-testing-of-lightweight-crypto-lwc2019.pdf

[11] A. Rukhin *et al.*, “Special Publication 800-22 Revision 1a A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications,” Apr. 2010. Available: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-22r1a.pdf