

Geradores de Números Pseudo-Aleatórios via Recorrências Não Lineares em $GF(2^e)$

Afonso S. D. Neto, Carlos E. C. Souza, Cecilio Pimentel e Daniel P. B. Chaves

Resumo—Este trabalho apresenta uma arquitetura de geradores de números pseudo-aleatórios (PRNGs, *pseudo-random number generators*) baseado em uma recorrência não linear sobre $GF(2^e)$. São propostas operações de pós-processamento não lineares aplicadas à saída do gerador, com o objetivo de melhorar a qualidade estatística das sequências binárias produzidas. Resultados experimentais mostram que o PRNG proposto é aprovado nas baterias de testes NIST SP 800-22 e TestU01 para polinômios de grau $e = 113$. A arquitetura proposta apresenta baixo custo computacional, mantendo as operações restritas à aritmética binária, o que implica em impacto reduzido em área e latência na implementação em circuito.

Palavras-Chave—geradores de números pseudo-aleatórios, recorrência não linear, corpos de Galois, circuito digital.

Abstract—In this work we propose a pseudo-random number generators (PRNGs) architecture based on a nonlinear recurrence over $GF(2^e)$. Nonlinear post-processing operations are applied to the generator output in order to improve the statistical quality of the generated binary sequences. Experimental results show that the proposed PRNG passes the NIST SP 800-22 and TestU01 statistical test suites for polynomials of degree $e = 113$. The proposed architecture exhibits low computational cost, restricting the operations to binary arithmetic, leading to reduced area and latency for the circuit implementation.

Keywords—pseudo-random number generators, nonlinear recurrence, Galois fields, digital circuit.

I. INTRODUÇÃO

Recorrências lineares sobre estruturas algébricas finitas, tais como anéis de inteiros e corpos finitos, são modelos eficientes para o projeto de geradores de números pseudo-aleatórios (PRNGs, *pseudo-random number generators*) [1], [2]. A aplicação iterativa de uma recorrência linear sobre uma condição inicial gera sequências numéricas, das quais são extraídos bits de sua representação binária. A sequência binária resultante é então empregada no projeto de PRNGs. Quando a arquitetura desses geradores é adequadamente projetada, as sequências produzidas exibem comportamento pseudoaleatório, atendendo a critérios estatísticos de aleatoriedade; além disso, tais PRNGs podem ser implementados em hardware dedicado com uso eficiente de recursos [3], [4].

Uma vantagem de PRNGs baseados em estruturas algébricas finitas é o fato dessas estruturas utilizarem exclusivamente

aritmética inteira. Como consequência, evitam-se possíveis degradações na dinâmica, decorrentes das operações de ponto flutuante típicas de implementações de funções definidas sobre os números reais [5]. Além disso, o consumo de recursos é reduzido em comparação com arquiteturas baseadas em números reais. Em particular, o corpo de extensão $GF(2^e)$ mostra-se adequado para implementação em circuitos digitais, pois as operações no corpo base são realizadas sobre $GF(2)$, reduzindo-se a operações binárias (e.g., XOR e deslocamentos). Portanto, essas implementações tipicamente apresentam baixa latência e alta eficiência em termos de área (número de portas lógicas) e consumo de energia [6].

Em [7] é proposto um PRNG baseado em uma recorrência afim sobre $GF(2^e)$. Nesse contexto, foram estabelecidas condições necessárias e suficientes para que as sequências geradas alcancem período máximo igual a $2^e - 1$. Além da análise teórica do período, foi apresentada uma arquitetura de circuito digital, cuja complexidade foi avaliada em termos de área e comprimento do caminho crítico (CCC). O projeto lógico explorou estruturas de árvore binária balanceada para otimizar o compromisso entre área e desempenho. A análise estatística, conduzida por meio das baterias NIST SP 800-22 [8] e TestU01 [9], revelou que o desempenho do PRNG é dependente da configuração adotada. Embora diferentes instâncias sejam aprovadas em subconjuntos dos testes, apenas uma configuração específica (com $e = 451$) é aprovada em todos os testes considerados. Tal limitação motiva a investigação de técnicas de pós-processamento, visando ampliar o conjunto de configurações com propriedades estatísticas adequadas.

Neste trabalho, a recorrência linear sobre $GF(2^e)$ empregada na arquitetura do PRNG introduzido em [7] é generalizada para uma recorrência não linear com a inclusão de um termo não linear baseado no automorfismo de Frobenius. Em seguida, é proposto um subsequente conjunto de operações de pós-processamento não lineares na saída do gerador. Essas operações têm o objetivo de melhorar a qualidade estatística das sequências geradas em relação a recorrências afins. É mostrado que o PRNG baseado na recorrência não linear proposta é aprovado nas baterias de testes estatísticos NIST SP 800-22 e TestU01 quando são empregados polinômios de graus maiores ou iguais a $e = 113$. A análise da implementação em circuito mostra que a arquitetura proposta preserva algumas vantagens das implementações associadas às recorrências lineares sobre estruturas algébricas finitas. Especificamente, a arquitetura proposta mantém operações restritas à aritmética binária, possibilitando a realização eficiente das etapas adicionais de pós-processamento, com impacto limitado em área e CCC.

Os autores são do Laboratório de Comunicações e Segurança da Informação, Departamento de Eletrônica e Sistemas, Universidade Federal de Pernambuco, Recife-PE; E-mails: {afonso.delgado, carlos.ecsouza, cecilio.pimentel, daniel.chaves}@ufpe.br.

Este trabalho foi parcialmente financiado pelo Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), pela Fundação de Amparo à Ciência e Tecnologia do Estado de Pernambuco (FACEPE) e pela Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

O restante deste trabalho está organizado da seguinte forma. Na Seção II é apresentada a recorrência afim sobre $\text{GF}(2^e)$. A recorrência não linear, operações de pós processamento e o PRNG proposto são detalhados na Seção III. Na Seção IV mostra-se alguns detalhes sobre a implementação em circuito do PRNG proposto. As considerações finais são discutidas na Seção VI.

II. RECORRÊNCIA AFIM SOBRE $\text{GF}(2^e)$

Seja $e \geq 2$ um número inteiro. O corpo de extensão $\text{GF}(2^e)$ é o conjunto das classes de equivalências dos polinômios com coeficientes em $\text{GF}(2)$ modulo $h(x)$, em que $h(x)$ é um polinômio irredutível em $\text{GF}(2)[x]$ com grau e . Seja $f : \text{GF}(2^e) \rightarrow \text{GF}(2^e)$ a relação de recorrência afim (RRA) definida por

$$p_{n+1}(x) = f(p_n) = (a(x)p_n(x) + c(x)) \bmod h(x) \quad (1)$$

em que $n \geq 0$ é um inteiro, $p_n(x) \in \text{GF}(2^e)$ é o polinômio que representa o n -ésimo estado, $1 \neq a(x) \in \text{GF}(2^e)$ é uma constante multiplicativa não nula fixa, e $c(x) \in \text{GF}(2^e)$ é uma constante aditiva não nula fixa. A aplicação recursiva de (1) sobre uma condição inicial $p_0(x) \in \text{GF}(2^e)$ define uma sequência $S = \{p_n(x)\}_{n \geq 0}$, em que $p_n(x) \in \text{GF}(2)[x]$ é um polinômio com grau menor que e , que representam elementos de $\text{GF}(2^e)$.

Em [7] é demonstrado que o máximo período possível da RRA (1) é $2^e - 1$, obtido quando $a(x)$ é um elemento primitivo em $\text{GF}(2^e)[x]$. A identificação de cada $p_n(x)$ com o vetor de seus coeficientes em $\text{GF}(2)$ induz uma sequência binária de comprimento e , que constitui a saída do PRNG a cada iteração.

As configurações do PRNG baseado em (1) são obtidas a partir de uma busca estruturada no espaço de parâmetros introduzida em [7]. Para um grau fixo e , consideram-se configurações da forma $(h(x), a(x), c(x))$, em que $h(x)$ é um polinômio primitivo com peso de Hamming fixo, $a(x)$ é uma raiz primitiva em $\text{GF}(2^e)$, e $c(x)$ é escolhido entre polinômios de baixa complexidade. As configurações são avaliadas por meio das baterias NIST e TestU01, sendo selecionadas aquelas com melhor desempenho estatístico. Observa-se empiricamente que a escolha de $c(x)$ tem impacto limitado, adotando-se, portanto, $c(x) = 1$.

III. RECORRÊNCIA NÃO LINEAR SOBRE $\text{GF}(2^e)$ E OPERAÇÕES DE PÓS-PROCESSAMENTO

Para mitigar os efeitos da estrutura linear da RRA em (1), cuja linearidade sobre $\text{GF}(2)$ limita a qualidade estatística das sequências geradas, conforme observado em [7], introduz-se um mecanismo não-linear composto por duas abordagens complementares: a modificação da própria recorrência e a aplicação de um algoritmo de pós-processamento sobre o vetor binário $\mathbf{u} \in \text{GF}(2)^e$ que representa os coeficientes de $p_n(x)$. Em particular, emprega-se uma transformação estruturada que combina três camadas principais.

- **Camada χ :** A função $\chi : \text{GF}(2^e) \rightarrow \text{GF}(2^e)$ de Keccak [10]

$$\chi(\mathbf{u})[j] = \mathbf{u}[j] \oplus (\neg \mathbf{u}[j+1]) \wedge \mathbf{u}[j+2], \quad j \in \mathbb{Z}/e\mathbb{Z} \quad (2)$$

aplica uma transformação não linear a cada bit j da sua entrada $\mathbf{u} \in \text{GF}(2)^e$, em que $\mathbb{Z}/e\mathbb{Z}$ é o conjunto dos inteiros módulo e e $\neg$ denota a negação lógica (complemento bit-a-bit). Essa aplicação é bijetiva se, e somente se, e é ímpar [11], [12].

- **Camada θ_r :** Define-se uma camada linear de difusão $\theta_r : \text{GF}(2)^e \rightarrow \text{GF}(2)^e$ baseada em deslocamento cíclico e operação XOR, dada por

$$\theta_r(\mathbf{u})[j] = \mathbf{u}[j] \oplus \mathbf{u}[(j+r) \bmod e], \quad j \in \mathbb{Z}/e\mathbb{Z} \quad (3)$$

em que r é coprimo com e , garantindo mistura global dos bits.

- **Camada $\mathbf{F}_k$:** O automorfismo de Frobenius $\mathbf{F}_k : \text{GF}(2^e) \rightarrow \text{GF}(2^e)$ definido por $\mathbf{F}_k : x \mapsto x^{2^k}$ no corpo $\text{GF}(2^e)$ [13], preserva a estrutura algébrica do corpo e induz uma permutação dos coeficientes. Adota-se $k = 7$, de modo que $\gcd(k, e) = 1$ para todos os graus e ímpares considerados, garantindo que $\mathbf{F}_k$ atue como um automorfismo de ordem e no corpo.

A escolha dessas três camadas é motivada por sua eficiência em hardware: a camada χ requer apenas uma porta NOT, uma AND e uma XOR por bit; θ_r consiste em e portas XOR com interconexões fixas; e $\mathbf{F}_k$, embora de maior conectividade, é implementável como uma rede puramente linear de XOR e multiplexadores, uma vez que a elevação ao quadrado é linear em corpos de característica 2. A composição dessas camadas promove simultaneamente não linearidade e difusão, contribuindo para a melhoria das propriedades estatísticas das sequências geradas.

No PRNG proposto, a composição $\mathbf{F}_k \circ \chi$ é incorporada diretamente à equação de atualização do estado, introduzindo não linearidade na própria recorrência. Assim, a dinâmica do estado passa a ser descrita pela relação de recorrência não linear (RRNL) $\varphi : \text{GF}(2^e) \rightarrow \text{GF}(2^e)$ definida por $p_{n+1}(x) = \varphi(p_n(x))$ tal que

$$\varphi(p_n(x)) = (a(x)p_n(x) + \mathbf{F}_k(\chi(\mathbf{u}_n)) + c(x)) \bmod h(x) \quad (4)$$

com condição inicial $p_0(x) \in \text{GF}(2^e)$, em que $\mathbf{u}_n \in \text{GF}(2)^e$ denota o vetor dos coeficientes binários de $p_n(x)$ e $\chi(\mathbf{u}_n) \in \text{GF}(2)^e$ é identificado com o elemento correspondente de $\text{GF}(2^e)$ antes da aplicação de $\mathbf{F}_k$. A camada χ introduz não linearidade sobre a representação binária, enquanto $\mathbf{F}_k$ opera algebricamente sobre o corpo $\text{GF}(2^e)$, perturbando a estrutura linear da recorrência afim.

No pós-processamento, adota-se um mecanismo baseado na composição $\chi \circ \theta_r \circ \chi$, denotada por Chi-Theta-Chi (CTC), aplicado ao vetor binário $\mathbf{u}_n$ associado a $p_n(x)$:

$$\text{CTC}(\mathbf{u}_n) = \chi(\theta_r(\chi(\mathbf{u}_n))). \quad (5)$$

Essa composição introduz não linearidade e difusão entre os bits de saída, combinando a ação não linear de χ com a mistura linear promovida por θ_r . As camadas χ e θ_r são puramente combinacionais e de custo reduzido. A cada iteração, uma palavra de e bits é produzida pela aplicação da CTC ao vetor binário que representa o estado.

A não linearidade introduzida em (4) torna a aplicação não injetiva, de modo que o período máximo de $2^e - 1$ não é mais

garantido. Sob a heurística de função aleatória [14] em um espaço de estados com $N = 2^e$ elementos, o comprimento esperado dos ciclos é dado por

$$\mathbb{E}[\text{comprimento dos ciclos}] \approx \sqrt{\pi N/2} = \sqrt{\pi 2^{e-1}}. \quad (6)$$

Nesse regime, o comprimento máximo dos ciclos é da ordem de $\mathcal{O}(\sqrt{N})$. Esse comportamento, característico de funções aleatórias, foi verificado empiricamente por meio de testes computacionais para valores reduzidos de e , especificamente $e \in \{5, 7, 11, 13, 17, 19\}$.

IV. IMPLEMENTAÇÃO EM CIRCUITO

A proposta de diagrama lógico para o PRNG proposto exige a análise do compromisso entre a área ocupada e o CCC do circuito. A área está diretamente relacionada à quantidade de componentes lógicos empregados (como portas XOR), enquanto o CCC corresponde ao atraso de propagação do maior caminho combinacional entre os estágios de entrada e saída, determinando assim a frequência máxima de operação do circuito.

A seguir são analisadas as complexidades de área e CCC de cada camada e da composição de pós-processamento CTC. Nesta análise, denota-se por τ_{XOR} , τ_{AND} , τ_{NOT} e τ_{MUX} os atrasos de propagação de uma porta XOR, AND, NOT e de um multiplexador 2-para-1, respectivamente. Assume-se que os atrasos de portas lógicas básicas são comparáveis, adotando-se a aproximação $\tau_{\text{NOT}} \approx \tau_{\text{AND}} \approx \tau_{\text{XOR}}$.

A. Camada χ

Cada bit de saída da função χ em (2) requer uma porta NOT, uma porta AND e uma porta XOR, totalizando e portas NOT + e AND + e XOR, com um CCC de $1\tau_{\text{NOT}} + 1\tau_{\text{AND}} + 1\tau_{\text{XOR}} \approx 3\tau_{\text{XOR}}$. As três entradas para o bit j são $u[j]$, $u[j+1]$ e $u[j+2]$ (índices módulo e), de modo que a célula é completamente local, sem necessidade de interconexões de longo alcance.

B. Camada θ_r

Cada bit de saída da função θ_r em (3) é o XOR de exatamente dois bits de entrada, um local ($u[j]$) e outro deslocado de r posições ($u[j+r]$). Toda a camada requer e portas XOR, com CCC de $1\tau_{\text{XOR}}$.

C. Camada $\mathbf{F}_k$ (automorfismo de Frobenius)

O automorfismo de Frobenius $\mathbf{F}_k : \text{GF}(2^e) \rightarrow \text{GF}(2^e)$ definido por $\mathbf{F}_k(p(x)) = p(x)^{2^k}$ atua sobre elementos de $\text{GF}(2^e)$ (polinômios), sendo o mais complexo das três camadas em termos de conectividade. Em corpos de característica 2, a elevação ao quadrado é uma operação linear, pois $(a+b)^2 = a^2+b^2$ em $\text{GF}(2)$. Por indução, $(a+b)^{2^k} = a^{2^k}+b^{2^k}$ para todo $k \geq 1$. Essa linearidade implica que $\mathbf{F}_k$ pode ser representado sem a necessidade de multiplicadores em $\text{GF}(2^e)$ e resultando em uma implementação eficiente em hardware. Assim, temos

$$p(x)^{2^k} = \left(\sum_{i=0}^{e-1} p_i x^i \right)^{2^k} = \sum_{i=0}^{e-1} p_i x^{i \cdot 2^k} \pmod{h(x)} \quad (7)$$

em que $p_i \in \text{GF}(2)$ é o i -ésimo coeficiente de $p(x)$. Cada termo $x^{i \cdot 2^k} \pmod{h(x)}$ é um polinômio pré-computado em $\text{GF}(2^e)$, fixado pela escolha de e , k e $h(x)$.

O circuito consiste, portanto, em e caminhos de *seleção e acumulação*, um para cada bit de entrada p_i :

- 1) Para cada $i = 0, \dots, e-1$, calcular a constante $C_i = x^{i \cdot 2^k} \pmod{h(x)} \in \text{GF}(2^e)$.
- 2) Selecionar entre C_i e o vetor nulo usando p_i como sinal de controle. Isso corresponde a um multiplexador 2-para-1 de e bits.
- 3) Combinar os e vetores selecionados por meio de uma árvore balanceada de XOR com profundidade $\lceil \log_2 e \rceil$.

Assim, a camada $\mathbf{F}_k$ utiliza e estágios de multiplexadores 2-para-1 de e bits e uma árvore XOR de $e-1$ nós (também de e bits), totalizando e^2 multiplexadores + $e(e-1)$ portas XOR, com CCC

$$\tau(\mathbf{F}_k) = \tau_{\text{MUX}} + \lceil \log_2 e \rceil \tau_{\text{XOR}}. \quad (8)$$

D. Composição em CTC

Utilizando os custos por camada derivados nesta seção, o pipeline $\chi \rightarrow \theta_r \rightarrow \chi$ da CTC em (5) requer $3e$ portas XOR + $2e$ portas AND + $2e$ portas NOT, com CCC

$$\tau(\text{CTC}) = 3\tau_{\text{XOR}} + 1\tau_{\text{XOR}} + 3\tau_{\text{XOR}} = 7\tau_{\text{XOR}}. \quad (9)$$

A Tabela I resume o compromisso entre o PRNG baseado na RRA e o PRNG proposto em termos de área e CCC. Em relação ao circuito da RRA em [7], o circuito do PRNG proposto adiciona e^2 multiplexadores e $e(e-1)$ portas XOR referentes à camada $\mathbf{F}_k$, além de $3e$ portas AND e $3e$ portas NOT relativas às três camadas χ (uma na atualização de estado e duas no pós-processamento). Em contrapartida, o CCC aumenta em apenas $1\tau_{\text{XOR}}$ em relação à RRA, uma vez que o caminho crítico da camada $\mathbf{F}_k \circ \chi$ opera em paralelo ao caminho afim principal. O período esperado do sistema proposto, discutido na Seção III, é da ordem de $\sqrt{\pi 2^{e-1}}$, conforme a heurística de função aleatória [14].

V. RESULTADOS

O desempenho estatístico do PRNG proposto foi avaliado por meio de duas suítes de testes amplamente utilizadas: TestU01, composta por oito baterias com número crescente de testes estatísticos (FIPS 140-2 (4), Alhabit (9), BlockAlhabit (9), pseudoDIEHARD (15), Rabbit (26), SmallCrush (10), Crush (96) e BigCrush (106)), e NIST SP 800-22, composta por 15 testes estatísticos. Para cada configuração, o gerador é considerado aprovado em uma bateria quando todos os seus sub-testes são aprovados (indicado por ✓); caso contrário, a configuração é reprovada (✗). Para a suíte NIST SP 800-22, adotou-se nível de significância $\alpha = 0.01$, conforme recomendado em [8].

A avaliação foi conduzida para graus ímpares do corpo de extensão considerado em [7], $e \in \{31, 113, 243, 451\}$. A Tabela II apresenta os polinômios irredutíveis $h(x)$ e os multiplicadores $a(x)$ empregados em cada configuração, seguindo a seleção de parâmetros de [7]. Em todos os casos, adotou-se $k = 7$ para o automorfismo de Frobenius.

TABELA I

COMPARAÇÃO ARQUITETURAL ENTRE O PRNG BASEADO NA RRA [7] E O PRNG PROPOSTO. A COLUNA ÁREA INDICA O NÚMERO DE PORTAS LÓGICAS. O CCC É EXPRESSO EM FUNÇÃO DOS ATRASOS τ_{XOR} E τ_{MUX} . O PESO DE HAMMING E O GRAU DE $a(x)$ SÃO DENOTADOS POR w_a E d_a , RESPECTIVAMENTE.

PRNG	Área	CCC	Período
RRA [7]	$e(w_a + d_a)$ XOR + ed_a MUX	$(\lceil \log_2(w_a+1) \rceil + \lceil \log_2(d_a+1) \rceil) \tau_{\text{XOR}} + \tau_{\text{MUX}}$	$2^e - 1$
Proposto	$e(w_a + d_a + e + 4)$ XOR + $e(d_a + e)$ MUX + $3e$ AND + $3e$ NOT	$(\lceil \log_2(w_a+1) \rceil + \lceil \log_2(d_a+1) \rceil + 1) \tau_{\text{XOR}} + \tau_{\text{MUX}}$	$\sim \sqrt{\pi 2^{e-1}}$

TABELA II

CONFIGURAÇÕES POLINOMIAIS PARA CADA GRAU e .

e	$h(x)$	$a(x)$
31	$x^{31} + x^{13} + x^8 + x^3 + 1$	$x^4 + 1$
113	$x^{113} + x^{83} + x^{72} + x^{59} + 1$	$x^{94} + x^{90} + x^{25} + x + 1$
243	$x^{243} + x^{216} + x^{183} + x^{134} + 1$	$x^{179} + x^{160} + x^{75} + x^{33} + 1$
451	$x^{451} + x^{402} + x^{292} + x^9 + 1$	$x^{406} + x^{360} + x^{168} + x^7 + 1$

TABELA III

RESULTADOS DAS BATERIAS TESTU01.

e	PRNG	FIPS	Alphabit	BlkAlpha	PsDH	Rabbit	SmallCrush	Crush	BigCrush
31	RRA [7]	✓	×	×	×	×	×	×	×
	Proposto	✓	×	×	×	×	×	×	×
113	RRA [7]	✓	✓	✓	✓	✓	✓	×	×
	Proposto	✓	✓	✓	✓	✓	✓	✓	✓
243	RRA [7]	✓	✓	✓	✓	✓	✓	✓	×
	Proposto	✓	✓	✓	✓	✓	✓	✓	✓
451	RRA [7]	✓	✓	✓	✓	✓	✓	✓	✓
	Proposto	✓	✓	✓	✓	✓	✓	✓	✓

TABELA IV

RESULTADOS NIST SP 800-22.

e	RRA [7]	Proposto
31	✓	×
113	✓	✓
243	✓	✓
451	✓	✓

A Tabela III resume os resultados das oito baterias do TestU01 para o PRNG baseado na RRA e para o PRNG proposto. Observa-se que, no conjunto de parâmetros avaliados, o PRNG proposto atinge aprovação em todas as oito baterias a partir de $e = 113$, incluindo Crush e BigCrush. Em contraste, o PRNG baseado na RRA somente alcança esse nível de desempenho para $e = 451$, evidenciando uma redução significativa no grau necessário para aprovação completa. Do ponto de vista prático, a redução do grau de $e = 451$ para $e = 113$ implica uma diminuição significativa na área do circuito e na latência, conforme analisado na Seção IV. Para $e = 31$, nenhuma das abordagens é capaz de superar as baterias mais exigentes, o que é consistente com o espaço de estados limitado (2^{31}), que impõe restrições ao período e à complexidade estatística das sequências geradas.

A Tabela IV apresenta os resultados da bateria NIST SP 800-22. Observa-se que o PRNG proposto é aprovado para $e \in \{113, 243, 451\}$, sendo reprovado para $e = 31$. Por outro lado, o PRNG baseado na RRA é aprovado para todos os valores de e considerados, incluindo $e = 31$. Essa diferença pode ser atribuída à maior sensibilidade da estrutura proposta às limitações do espaço de estados em baixos graus, embora, para graus mais elevados, ambos os geradores apresentem desempenho estatístico satisfatório nessa bateria.

VI. CONCLUSÕES

Este trabalho apresentou uma arquitetura de PRNG baseada em uma recorrência não linear sobre o corpo de extensão $\text{GF}(2^e)$. Adicionalmente, foram propostas operações de pós processamento não lineares na saída do gerador. Os resultados obtidos nas baterias NIST SP 800-22 e TestU01 indicam que, no conjunto de parâmetros avaliados, o PRNG proposto atinge aprovação em todos os testes a partir de um grau mínimo de $e = 113$, representando uma melhoria significativa em relação à abordagem RRA em [7], que requer grau $e = 451$ para alcançar desempenho estatístico equivalente. Do ponto de vista de implementação, como o corpo base utiliza aritmética binária, o aumento de complexidade decorrente da introdução de não linearidades é limitado, permitindo que a arquitetura preserve

eficiência em termos de área e latência em implementações em hardware.

Trabalhos futuros incluem a análise de segurança do PRNG proposto, investigando sua resistência a ataques de predição de estado e distinguibilidade estatística. Outra possibilidade é desenvolver a implementação em arranjo de portas programável em campo (FPGA, *field-programmable gate array*) para validação experimental das frequências máximas de operação previstas pela análise teórica do CCC, consolidando a viabilidade da arquitetura em cenários reais de aplicação.

REFERÊNCIAS

- [1] C. E. C. Souza, D. Moreno, M. H. S. Sousa, D. P. B. Chaves, and C. Pimentel, “High-throughput pseudo-random number generators over discrete chaos,” *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 72, no. 9, pp. 1303–1307, Sep 2025.
- [2] C. E. C. Souza, D. Moreno, D. P. B. Chaves, and C. Pimentel, “Pseudo-chaotic sequences generated by the discrete Arnold’s map over $\mathbb{Z}_{2^m}$: Period analysis and FPGA implementation,” *IEEE Trans. Instrum. Meas.*, vol. 71, pp. 1–10, Aug 2022.
- [3] P. Kietzmann, T. C. Schmidt, and M. Wählisch, “A guideline on pseudorandom number generation (PRNG) in the IoT,” *ACM Comput. Surv.*, vol. 54, no. 6, Jul 2021.
- [4] U. Zia, M. McCartney, B. Scotney, J. Martinez, and A. Sajjad, “A resource efficient pseudo random generator based on sawtooth maps for Internet of Things,” *Secur Privacy*, vol. 6, no. 5, p. e304, Feb 2023.
- [5] B. Chirikov and F. Vivaldi, “An algorithmic view of pseudochaos,” *Physica D: Nonlinear Phenomena*, vol. 129, no. 3, pp. 223–235, May 1999.
- [6] T.-C. Chen, S.-W. Wei, and H.-J. Tsai, “Arithmetic unit for finite field $GF(2^m)$,” *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 55, no. 3, pp. 828–837, Apr 2008.
- [7] A. de Sá Delgado Neto, C. E. C. Souza, D. C. M. de Almeida, M. H. da Silva Sousa, D. P. B. Chaves, and C. Pimentel, “Geradores de números pseudo-aleatórios baseados em recorrências lineares sobre $GF(2^e)$,” in *XLIII Simpósio Brasileiro de Telecomunicações e Processamento de Sinais*, Sep 2025, pp. 1–5.
- [8] L. E. Bassham III *et al.*, “SP 800-22 Rev. 1a. A statistical test suite for random and pseudorandom number generators for cryptographic applications,” Nat. Inst. Std. & Technol., Gaithersburg, MD, United States, Tech. Rep., 2010.
- [9] P. L’ecuyer and R. Simard, “TestU01: A C library for empirical testing of random number generators,” *ACM Trans. Math. Softw.*, vol. 33, no. 4, pp. 1–40, 2007.
- [10] G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche, “The Keccak reference,” Tech. Rep., Jan. 2011, version 3.0, <https://keccak.team/files/Keccak-reference-3.0.pdf>.
- [11] A.-M. Graner, B. Kriepke, L. Krompholz, and G. M. Kyureghyan, “On the bijectivity of the map χ ,” *Cryptogr. Commun.*, vol. 18, pp. 97–105, 2025.
- [12] J. Schoone and J. Daemen, “Algebraic properties of the maps χ_n ,” *Des. Codes Cryptogr.*, vol. 92, no. 8, pp. 2341–2365, 2024.
- [13] R. J. McEliece, *Finite fields for computer scientists and engineers*, 1st ed., ser. The Kluwer international series in engineering and computer science Information theory SECS23. Kluwer Academic Publishers, 1987.
- [14] P. Flajolet and A. M. Odlyzko, “Random mapping statistics,” in *Advances in Cryptology — EUROCRYPT ’89*, ser. Lecture Notes in Computer Science, vol. 434. Springer, 1990, pp. 329–354.