

Equivalent Non-linear Shift Register Transformations and Espresso

Anirban Ghatak

ghatak.anirban@gmail.com

Indian Statistical Institute

Kaushik Sarkar

Narula Institute of Technology, Kolkata

Anupam Chattopadhyay

Nanyang Technological University

Ambrish Awasthi

Scientific Analysis Group, DRDO

Indivar Gupta

Scientific Analysis Group, DRDO

Research Article

Keywords: Non-linear shift registers, Galois and Fibonacci implementation , Espresso security

Posted Date: May 29th, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-9706347/v1>

License: © ⓘ This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

Equivalent Non-linear Shift Register Transformations and Espresso

Anirban Ghatak¹[0000–0002–8654–2411], Kaushik Sarkar², Anupam Chattopadhyay³, Ambrish Awasthi⁴[0009–0000–9066–5708], and Indivar Gupta⁴

¹ Indian Statistical Institute, Kolkata, India
`ghatak.anirban@gmail.com`

² Narula Institute of Technology, Kolkata
`kaushik.srkr@gmail.com`

³ CCDS, Nanyang Technological University, Singapore
`anupam@ntu.edu.sg`

⁴ Scientific Analysis Group, DRDO, Metcalfe House, Delhi, India, 110054
`ambrishawasthi.sag@gov.in`

Abstract. The stream cipher Espresso was designed by E. Dubrova and M. Hell as a hardware-efficient candidate for 5G security. It has two components - a 256-bit binary non-linear feedback shift register (NFSR), with 14 feedback taps, and a 20-variable non-linear output function. However, the security of Espresso with regard to several standard cryptanalytic attacks are estimated using an alternate NFSR, which has only two feedback taps. The above analysis is based on the claim that this alternate NFSR is obtained using a transformation algorithm which preserves the output sequence of the original NFSR. This issue has become all the more significant in the light of recent results on hardware performance evaluations, which assume that the stated equivalence holds.

This article presents the first successful attempt in settling the above question of Espresso NFSR equivalence. We arrive at our main result by a careful analysis of existing transformation methods relevant to the question. In doing so, we first identify and correct a critical flaw in the proof of a foundational result of E. Dubrova. Next, in the context of Espresso equivalence, we analyze and unify the transformation framework of Dubrova and the one proposed by Yao and Parampalli. Based on our analysis, we have succeeded in transforming the LFSR in an Espresso-equivalent design reported by Yao and Parampalli to the Espresso alternate NFSR. Further, we have obtained a corresponding non-linear output function which achieves output-equivalence with the LFSR design, and hence, with Espresso.

Keywords: Non-linear shift registers · Galois and Fibonacci implementation · Espresso security

1 Introduction

Linear and non-linear feedback shift registers (LFSRs and NFSRs/NLFSRs) are key design components of several families of stream ciphers. The properties of

LFSRs have been well-studied since the seminal work of S. Golomb [9], and a recent work [1] rigorously showcases both new analyses/construction and a survey of key contributions pertaining to LFSR-based designs. In comparison, the analysis of NFSRs is still a work in progress. But recent advances indicate that it may be possible to provide the same level of cryptographic security as LFSR-based designs using NFSRs but with a noticeably reduced hardware footprint. Efforts in this direction have resulted in the design of NFSR-based stream ciphers like LIZARD [10], TRIVIUM [2], Grain [11] and, more recently, Espresso [7]. However, providing security guarantees for NFSR-based stream ciphers is more of a challenge than LFSR-based designs owing to the comparative difficulty in NFSR analysis. A possible way out may be to formulate an *equivalent design* involving an LFSR, or a simpler NFSR, whose analysis is more tractable.

Another aspect of significance in the design of shift-register based stream ciphers is the choice between *Galois* and *Fibonacci* configurations of shift registers. In a Fibonacci NFSR there is only a single feedback update function from the intermediate stages to the terminal stage, all the other stages accept a single input from just the preceding stage. Hence, the Fibonacci configuration is a restricted case of the general Galois configuration, which potentially allows feedback between any pair of stages in the shift register. In [12], it is claimed that a stream cipher based on a Galois-configuration shift register can achieve higher clock frequency on application-specific integrated circuit (ASIC). Again, the security analysis of a stream cipher using the Fibonacci configuration is comparatively easier and well-studied in literature.

The stream cipher Espresso was proposed by E. Dubrova and M. Hell [7] as a hardware-efficient candidate for providing security in 5G communication. This cipher offers an excellent case study for the aforementioned competing design choices - Galois vs. Fibonacci and LFSR vs. NFSR. As a contender for 5G security, Espresso claims better performance than Grain-128 and Trivium in the under-1500 Gate Equivalence (GE) chip-size regime with the stated parameters: 1497 GE area, 2.22 Gbits/sec throughput and 232 nanosecond latency.

Espresso is based on a 256-bit Galois NFSR, having 14 non-linear feedback update functions at different stages, and a non-linear output function, involving the outputs of 20 different stages. However, the security of Espresso against standard cryptanalysis strategies like linear approximation attacks and TMDTO attacks is actually discussed in [7] with respect to an “equivalent” NFSR (cf. p. 277 of [7]), having only two non-linear update functions. Hence, Espresso is clearly an instance of the “analysis-on-equivalent-design” strategy mentioned earlier.

The motivation for the present work arose from an examination of the claimed equivalence of the two NFSRs presented in the Espresso proposal paper in the context of cryptanalysis of Espresso. An immediate issue had already been reported in [16]: no “equivalent” non-linear output function is provided in [7] corresponding to the alternative Galois NFSR. This precludes any comparison between the actual keystreams generated by Espresso and an “equivalent” cipher based on the alternative NFSR. Further unsubstantiated claims made in [7] in this context were reported in [8].

A series of papers by Yao and Parampalli ([16], [15], [17]) culminated in defining the so-termed ‘compensation method’ for transformations of several types of Fibonacci and Galois NFSRs. Their method applied to Espresso is claimed to yield (cf. [16], Section 6.1):

“... a Fibonacci LFSR with a nonlinear output function. Specifically, the output function consists of 2289 monomials. There are 104 variables in the output function and the algebraic degree is 12.”

But the question of whether or not the two NFSRs presented in [7] are actually equivalent was still not resolved. Notwithstanding the existence of this gap, some recent works have already attempted improved implementation of Espresso configurations (e.g. [14]), assuming the validity of the equivalence as claimed in [7].

In this article, we expand on the results reported in [8] and attempt to conclude a proposed resolution to the *Espresso equivalence problem* reported therein. We begin with a careful analysis of a sequence of transformation techniques which are outlined in E. Dubrova’s works [3], [4],[6], and those of G. Yao and U. Parampalli [16], [17]) in the context of the equivalence problem. We then proceed to define a unified framework of the Dubrova and Yao-Parampalli algorithms, the *Dubrova-Yao-Parampalli (D-Y-P) framework*, to gain new insights as to the claimed equivalence. Finally, we present experimental results and theoretical analysis to *achieve this claimed equivalence* using the D-Y-P framework. Specifically, our contributions, collated with those reported in [8], are as follows.

1. We identify a critical mis-step in the proof of the foundational theorem of Dubrova’s first algorithm (cf. Theorem 1, Section IV, [3]) and provide a correct proof (Section 3).
2. We show that Dubrova’s first algorithm [3], is essentially subsumed into the framework of the second algorithm [6] (cf. [8]) (Subsection 4.1).
3. We establish that the Yao-Parampalli algorithm ([16], [17]) is equivalent to Dubrova’s second algorithm in the case of Fibonacci-to-uniform Galois NFSR transformation (cf. [8]) for computing the feedback functions and matching initial states (Subsection 4).
4. Using the D-Y-P framework, we obtain a NFSR/non-linear function ensemble from the Yao-Parampalli LFSR-based design, featuring the Espresso alternate NFSR F (cf. Subsection 5.1). Specifically, we have obtained a non-linear output function corresponding to the NFSR F , thereby closing the gap in the equivalence claims made in [7].
5. We have supported our experimental results⁵ using the obtained ‘cipher’ featuring the Espresso alternate NFSR F , with theoretical analysis (cf. Section 5). Our results allow us to conclusively state that:

Using the D-Y-P transformation framework, we have positively settled the Espresso Equivalence Problem by proving the equivalence of the Espresso designs in the sense of matched outputs of the two NFSR/non-linear function ensembles.

⁵ The Sagemath code for our reference implementation is available at:
<https://github.com/kaushiksrkr/YPFibonacci2EspressoFGalois.git>

2 Preliminaries and Prior Work

A foundational analysis of non-linear feedback shift registers, in particular, the cycle structure of the *state transition graph*, is given, like all matters “shift register”, in Golomb’s text ([9], Part Three, Chapter VI onwards). We proceed to outline two major algorithms in the works of E. Dubrova, dealing with the transformations between Fibonacci and the so-called uniform Galois NFSRs. This is followed by a brief account of the NFSR transformation framework in the works of G. Yao and U. Parampalli.

2.1 The Transformation Framework of E. Dubrova

Our treatment of the transformation framework of E. Dubrova is based on the results in [3], [4], and [6]. In later sections we show that there is an evolutionary thread underlying these works which culminated in the Espresso design in [7]. In the following brief exposition, we segregate Dubrova’s results in the three representative papers under the following heads.

- Transformation from Fibonacci to *uniform* Galois NFSR;
- Matching initial states in Fibonacci-uniform Galois NFSR transformations;
- Extension to (uniform) Galois-Galois NFSR transformations.

Fibonacci to Uniform Galois NFSR Transformation [3] The first algorithm of E. Dubrova in [3] considers the transformation of a Fibonacci NFSR to a *uniform* Galois NFSR, which guarantees the following notion of equivalence between the NFSRs (cf. *Definition 1*, p.5265, [3]):

“Two NLFSRs are equivalent if their sets of output sequences are equal.”

In [3], the *output bit* of an NFSR is specified to be the 0-th bit (cf. Section II), and so, the transformation attempts to match the output sequence at the 0-th stage of the original Fibonacci NFSR with that at the 0-th stage of the transformed Galois NFSR.

The validity of this transformation is based on the uniform property of a Galois NFSR, which is defined as follows(cf. *Definition 7*, [3]).

Definition 1. *The terminal bit of an n -bit NFSR is the bit with the minimal index which satisfies the following condition: For all bits i such that $i < \tau$, the feedback function f_i is of type $f_i = x_{(i+1)_n}$, where $(i+1)_n := (i+1) \bmod n$.*

Definition 2. *An n -bit NFSR is uniform if*

(a) *all its feedback functions are singular functions of the following form:*

$$f_i(x_0, x_1, \dots, x_{n-1}) = x_{(i+1)_n} \oplus g_i(x_0, x_1, \dots, x_{n-1}) \quad (1)$$

$g_i : \{0, 1\}^{n-1} \rightarrow \{0, 1\}$ be such that $(i+1)_n \notin \text{dep}(g_i)$.

(b) for all its bits i such that $i > \tau$, the following condition holds:

$$\max_index(g_i) \leq \tau \quad (2)$$

where τ is the terminal bit of the NFSR.

In the above context, the *dependence set* of a Boolean function f is denoted by:

$$\text{dep}(f) := \{i \mid f|_{x_i=0} \neq f|_{x_i=1}\}$$

where $f|_{x_i=j} = f(x_0, x_1, \dots, x_{i-1}, j, x_{i+1}, \dots, x_{n-1})$, for $j \in \{0, 1\}$.

The main algorithm presented in [3] (cf. Section V) is a transformation algorithm from a Fibonacci to a *fully shifted*, uniform Galois NFSR, which is claimed to preserve the output sequence at the 0-th stage. The term “fully shifted” refers to the fact that this construction guarantees the maximum possible shift from the Fibonacci configuration while maintaining the uniform property.

Matching Initial States in Fibonacci-Galois Transformations [4] In this work dealing with transformations between equivalent NFSRs, in the sense of [3], a formula for the *matching* initial state of a uniform Galois NFSR is given in terms of the initial state of the Fibonacci NFSR.

Theorem 1 (Theorem 3, p. 2965, [4]). *Let N_F be an n -bit Fibonacci NLFSR and N_G be an equivalent to N_F uniform Galois NLFSR with the terminal bit τ , $0 \leq \tau < n - 1$, and the feedback functions of type*

$$\begin{aligned} f_{n-1}(x) &= x_0 \oplus g_{n-1}(x_0, \dots, x_\tau) \\ f_{n-1}(x) &= x_0 \oplus g_{n-1}(x_0, \dots, x_\tau) \\ &\dots \\ f_\tau(x) &= x_{\tau+1} \oplus g_{n-1}(x_0, \dots, x_\tau) \\ f_{\tau-1}(x) &= x_\tau \\ &\dots \\ f_0(x) &= x_1 \end{aligned}$$

where $g_{n-1}, g_{n-2}, \dots, g_\tau$ are Boolean functions.

If N_F is initialized to a state $s = (s_0, s_1, \dots, s_{n-1})$, and N_G is initialized to the state $r = (s_0, s_1, \dots, s_\tau, r_{\tau+1}, r_{\tau+2}, \dots, r_{n-1})$ such that

$$r_i = s_i \oplus g_{i-1}(s) \oplus g_{i-2}|_{+1}(s) \oplus \dots \oplus g_\tau|_{+(i-\tau-1)}(s) \quad (3)$$

for all $i \in \{n-1, n-2, \dots, \tau+1\}$, then N_F and N_G generate the same output sequences.

In Section 4, it will be shown that this result can be interpreted in terms of *simple compensation* in the Yao-Parampalli transformation framework [17] (cf. Subsection 2.2).

General Framework for Galois-to-Galois NFSR Transformations [6]

In [6], a more general algorithm was presented which handles transformations between two NFSRs in the Galois configuration via the operation of *valid shifting* described as follows.

Definition 3. *Given an n -variate mapping defined by functions of the form in Equation 1, a shifting $f_i \xrightarrow{m} f_j$, $i, j \in \{0, 1, \dots, n-1\}$, $i \neq j$, shifts the monomial m in the algebraic normal form (ANF) of f_i to the ANF of f_j , such that every index $a \in \text{dep}(m)$ is changed to $b = (a - i + j) \bmod n$.*

Definition 4. *Consider an n -variate mapping with functions of the form in Equation 1, such that the values computed by f_z , $z \in \{0, 1, \dots, n-1\}$ form the output sequence. Then a shifting $g_i \xrightarrow{m} g_j$ is valid if, for every index $a \in \text{dep}(m)$ and b as in Definition 3, the following conditions hold.*

1. *For each $c \in [a, b] \setminus i$, $g_c = 0$; if $i \in [a, b]$, then $g_i^* = 0$ ($g_i^* = g_i \oplus m$);*
2. *For each $k \in [i, j]$, $k \notin \text{dep}(g_i^*)$.
Further, $k \notin \text{dep}(g_p)$ for all $p \in \{0, 1, \dots, n-1\} \setminus i$;*
3. *Neither $[a, b]$ nor $[i, j]$ contain both z and $z-1$.*

In the above definition, $[i, j]$ stands for the set $\{i, i-1, \dots, j\}$ if $i > j$, and the set $\{i, i+1, \dots, j\}$ if $i < j$, with similar definitions for $[a, b]$. Further, if multiple indices z constitute the output sequence, the third condition must be fulfilled for all pairs $(z, z-1)$.

The following result from [6] establishes that when a Galois NFSR is transformed to another through valid shiftings, the output sequences of the two NFSRs may differ only in a specific subset of the stages of the internal state.

Theorem 2 (E. Dubrova [6]). *Let F be a mapping as defined by Equation 1 and F' be a mapping obtained from F by applying a valid shifting $g_i \xrightarrow{m} g_j$, where $i, j \in \{0, \dots, n-1\}$ and $i \neq j$. Denote $\mathcal{J} = \{i+1, i+2, \dots, j\}$ and $\mathcal{I} = \{i, i-1, \dots, j+1\}$. Let F be initialized to the state $(s_0, s_1, \dots, s_{n-1})$ and F' to the state $(r_0, r_1, \dots, r_{n-1})$ such that:*

1. *For $k \in \mathcal{I}$, $r_k = s_k \oplus m|_{k-i-1}$*
2. *For $k \in \mathcal{J}$, $r_k = s_k \oplus m|_{k-j-1}$*
3. *$r_k = s_k$ for all $k \in \{0, 1, \dots, n-1\}$ not in either $\mathcal{I}$ or $\mathcal{J}$.*

Then the sequences of states generated by F and F' may differ only in the bit positions in $\mathcal{I}$ or $\mathcal{J}$, respectively.

Remark 1. The above theorem is a generalization of a result in [4] (Theorem 2, p. 2964), which was proved for the shift of a feedback function from the terminal bit τ to a new terminal bit $\tau-1$.

2.2 The Yao-Parampalli Framework

In [17], a classification of Galois NFSRs based on the forms of the feedback functions is discussed, where four types of NFSRs are described and four sets of transformation algorithms are presented. Of interest in this article is the Type-II Galois NFSR, which corresponds to Dubrova's uniform Galois NFSR. Henceforth, we will refer to this configuration as uniform Galois.

The so-called *compensation method* was defined to satisfy the notion of output equivalence where the output is a function of several stages of the shift register. In this context, we state the following definitions from [17].

Definition 5. A compensation list is a list with n elements denoted as $C = [c_0, c_1, \dots, c_{n-1}]$, where each element $c_i(x_0, \dots, x_{n-1})$, $i \in \{0, \dots, n-1\}$ is a combination of variants of shifted monomials.

The combination of two compensation lists C_{m1} and C_{m2} , for two monomials $m1$ and $m2$, is defined as the bitwise XOR $C = C_{m1} \oplus C_{m2}$.

Given an n -bit NFSR, the compensation list for shifting a monomial m from f_a to f_b with $a, b \in [0, n-1]$ and $a > b$, is constructed as

$$C = [c_0, c_1, \dots, c_{n-1}] = [0, \dots, 0, m|_{-(a-b)}, \dots, m|_{-1}, 0, \dots, 0]$$

where $c_i = 0$ for $i \in [0, b]$ or $[a+1, n-1]$ and $c_i = m|_{-(a-i+1)}$ for $i \in [b+1, a]$.

Compensation vs. Iterative Compensation In addition to the compensation operation, where the state variables are directly XOR-ed with the corresponding entries in the compensation list, Yao and Parampalli define the operation of *iterative compensation in descending (or ascending) order*. We discuss this with an illustration, following [17], and henceforth term the direct compensation operation as *simple* compensation to emphasize the difference.

Let the indices x_i, x_j , $j = i-1$, be compensated by the entries c_i, c_j of the compensation list, respectively.

Simple compensation results in $x_i \rightarrow x_i \oplus c_i$ and $x_j \rightarrow x_j \oplus c_j$.

For iterative compensation *in the descending order*, x_i is first transformed to $x_i \oplus c_i$. If $c_i = c_i(\dots, x_j, \dots)$, then, in the next step, $x_i \rightarrow x_i \oplus c_i(\dots, x_j \oplus c_j, \dots)$, i.e. the effect of compensation at the next lower index $j = i-1$ is applied iteratively on the compensated x_i .

$$\begin{aligned} f(x_0, x_1, x_2) &\xrightarrow{x_2 \rightarrow x_2 \oplus c_2(x_1)} f(x_0, x_1, x_2 \oplus c_2(x_1)) \\ &\xrightarrow{x_1 \rightarrow x_1 \oplus c_1} f(x_0, x_1 \oplus c_1, x_2 \oplus c_2(x_1 \oplus c_1)) \end{aligned}$$

Yao-Parampalli Algorithms for Transformations between Fibonacci and Uniform Galois NFSRs [17]

– *Fibonacci-to-Uniform Galois*

Input: n -bit Fibonacci NFSR; output function f_z ; initial state X^0 .

Output: n -bit uniform Galois NFSR; transformed output function $\overline{f_z}$; matched initial state $\hat{X}^0$.

1. For each monomial m_j , $j \in [1, r]$, compute the lowest permissible shifting index p_j using Dubrova's algorithm in [3] and choose a position τ_j , such that $p_j \leq \tau_j \leq n - 1$.
2. Shift the r monomials from feedback function f_{n-1} to $f_{\tau_1}, f_{\tau_2}, \dots, f_{\tau_r}$, where $0 \leq \tau_1 \leq \tau_2 \leq \dots \leq \tau_r \leq n - 1$.
3. For each monomial m_j , $j \in [1, r]$, construct compensation list

$$C_{m_j} = [0, \dots, 0, m_j|_{-(n-1-\tau_j)}, \dots, m_j|_{-1}]$$

XOR all such lists to compute a combined compensation list:

$$C = C_{m_1} \oplus C_{m_2} \oplus \dots \oplus C_{m_r}.$$

4. Compensate the output function f_z by C iteratively in the descending order to obtain the output function $\overline{f_z}$ for the transformed NFSR.
 5. The initial state computed as: $\hat{x}_i^0 = x_i^0 \oplus c_i(X^0)$, $i \in [0, n - 1]$.
- *Uniform Galois-to-Fibonacci*

Input: n -bit uniform Galois NFSR; output function f_z ; initial state X^0 .

Output: n -bit Fibonacci NFSR; transformed output function $\overline{f_z}$; matched initial state $\hat{X}^0$.

1. Shift every monomial $g_i \neq 0$, from f_i , $i \in [\tau, n - 2]$ to f_{n-1} .
2. For each $g_i \neq 0$ in the first step, compute the compensation list

$$C_{g_i} = [0, \dots, 0, g_i, g_i|_{+1}, \dots, g_i|_{+(n-2-i)}]$$

XOR all such lists to compute a combined compensation list:

$$C = C_{g_0} \oplus C_{g_1} \oplus \dots \oplus C_{g_{n-2}}.$$

3. Compensate the output function f_z by C *without iteration* to obtain the output function $\overline{f_z}$ for the transformed Fibonacci NFSR as

$$\overline{f_z} = f_z(x_0 \oplus c_0, \dots, x_{n-1} \oplus c_{n-1})$$

4. The initial state $\hat{X}^0 = [\hat{x}_0^0, \hat{x}_1^0, \dots, \hat{x}_{n-1}^0]$ is given by:

$$\hat{x}_i^0 = x_i^0 \oplus \overline{c_i}(X^0), \quad i \in [0, n - 1]$$

where $\overline{c_i}$ is obtained from c_i by iterative compensation in descending order using C .

Remark 2. The Yao-Parampalli algorithms directly use Dubrova's formulas in [3] to define the shifted monomials as well as the terminal bits associated with each shift.

We conclude this section with a summary comparison of the relevant features of the Dubrova and Y-P frameworks in the following Table 1. This serves as an overture to the analyses and results in subsequent sections, in particular, Section 4, where the connection between the frameworks is analysed further in the context of Fibonacci-to-uniform Galois transformations.

Framework	Transformation Profile	Equivalence	Matching Initial States	Validation Status
Dubrova	Fibonacci to Uniform Galois [3]	Output at final stage	No	Proved
	Fibonacci to Uniform Galois [4]	Output at final stage	Yes	Proved
	Galois to Galois [6]	Output function on restricted subset of stages	Yes	Proved
Yao-Parampalli	Fibonacci to Uniform Galois [17]	Output function on any subset of stages	Yes	Claimed
	Uniform Galois to Fibonacci [17]	Output function on any subset of stages	Yes	—

Table 1: Comparison of Dubrova and Yao-Parampalli Frameworks

3 “Fixing” Dubrova’s Main Theorem

All the transformation algorithms analysed in the subsequent sections of this article build on Dubrova’s Fibonacci-to-uniform Galois NFSR transformation in [3]. However, we show that there is a non-trivial flaw in the proof of the key result - termed the “main theorem” in [3]. In the remainder of this section we rectify this discrepancy and present a correct proof of this theorem, thereby validating a keystone result in Dubrova’s transformation framework.

We begin by stating the result in question from [3]:

Theorem 3. *Given a uniform NLFSR with terminal bit a , a shifting $g_a \xrightarrow{P} g_b$, $P \subseteq A_{g_a}$, $b < a$, results in an equivalent NLFSR if the transformed NLFSR is uniform as well.*

The proof uses the *feedback graph* description of an NFSR as defined below [3].

Definition 6. *The feedback graph of an n -bit NFSR is a directed graph with n vertices $v_0, \dots, v_{n-1}$ which represents bits $0, \dots, n-1$ of the NFSR, respectively. There is an edge from v_i to v_j if $i \in \text{dep}(f_j)$.*

It is noted that the discrepancies in the proof of this theorem stem from an incorrect application of the ‘substitution’ operation (Definition 3 of Section III in [3]), stated below.

Definition 7. *The operation substitution, denoted by $\text{sub}(v_i, v_j)$, is defined for any vertex v_i , which has a unique predecessor v_j . The substitution $\text{sub}(v_i, v_j)$ removes v_i from the feedback graph and, for each successor v_k of v_i , replaces the edge (v_i, v_k) by an edge (v_j, v_k) .*

Clearly the operation is defined as $\mathbf{sub}(v(\mathbf{successor}), v(\mathbf{predecessor}))$.

3.1 The Discrepancy and Proposed Correction

We first outline the relevant steps of the proof provided in [3]. While we adhere to the original notation in [3] as much as possible, we label the first uniform NFSR as N_a and the second as N_b , according to the respective terminal bits. The NFSR transformation is effected by the shift of a single product-term $x_k x_a$, $k < a$, from g_a to g_b . Express g_a as $g_a = g_a^* \oplus x_k x_a$.

The strategy of Dubrova's proof may be stated as follows:

If the NFSR N_b , with terminal bit b , is uniform, its feedback graph is reducible to a single vertex v_b . Then there exists a non-linear recurrence describing the sequence at v_b . The equivalence between N_a, N_b is established if this recurrence is shown to be identical with that describing the sequence at v_a for N_a .

Relevant Steps in Dubrova's Proof:

Let $\tau = a$ and $\tau = b$ represent the respective terminal bits for N_a and N_b . As outlined in the steps of the proof to Theorem 1 of [3], the substitution operation is invoked twice in arriving at the non-linear recurrence for each terminal bit $\tau, \tau \in \{a, b\}$. For instance, for $\tau = a$, the vertices $i < a$ are first removed via the substitution sequence:

$$\mathbf{sub}(v_0, v_1), \mathbf{sub}(v_1, v_2), \dots, \mathbf{sub}(v_{a-1}, v_a)$$

Then the vertices $v_i, i > \tau = a$ are removed as follows in Section IV of [3], page 5268 (we quote in italics):

"After a sequence of $n - a - 1$ substitutions $\mathbf{sub}(v_{n-1}, v_{n-2}), \dots, \mathbf{sub}(v_{a+1}, v_a)$, we get a non-linear recurrence describing the sequence of values of bit a

$$s_a(t) = s_a(t - n) \oplus g_{n-1}(\tilde{s}_a(-n + a + 1)) \oplus g_{n-2}(\tilde{s}_a(-n + a)) \\ \oplus \dots \oplus g_a^*(\tilde{s}_a) \oplus s_a(t - 1 - a + k) s_a(t - 1)."$$

The notation $\tilde{s}_a(i)$ has been defined (cf. pages 5267, 5268) to mean that each element $s_a(x)$ of $\tilde{s}_a$ has been replaced by $s_a(x + i)$, where $\tilde{s}_a$ stands for

$$\tilde{s}_a := (s_a(t - a - 1), s_a(t - a), \dots, s_a(t - 1 - (a - b))) \quad (4)$$

A similar step is described in arriving at the non-linear recurrence for the values of the bit b .

The Discrepancy

We observe that:

- The vertices $i > \tau$, for $\tau = a$ or b , are removed via the sequence:

$$\mathbf{sub}(v_{n-1}, v_{n-2}), \dots, \mathbf{sub}(v_{\tau+1}, v_\tau) \quad (5)$$

It is noted that the arguments of the terms in Equation (5) are not consistent with the definition of the substitution operation (cf. Definition 7) as they have the form

$$(v(\mathbf{predecessor}), v(\mathbf{successor})).$$

- Even if the order of the arguments is reversed in each operation, the second substitution sequence fails to achieve the presumed objective of reducing all the remaining vertices to v_τ .

However, a simple modification to the strategy of applying the substitution operation suffices to remove this discrepancy in the proof, as discussed below.

The Correction

We aver that the correct strategy to evaluate the non-linear recurrence for the sequence of values of the terminal bit is to apply the following sequence of substitutions, *after* all the vertices labelled $i < \tau$ have been removed:

$$\text{sub}(v_{n-1}, v_\tau), \text{sub}(v_{n-2}, v_\tau), \dots, \text{sub}(v_{\tau+1}, v_\tau)$$

Changing the time-shifts of $s_\tau(t)$ appropriately at each stage will result in identical non-linear recurrences for both $\tau = a$ and $\tau = b$, thereby proving the theorem.

3.2 Proposed Correct Proof of Dubrova's Theorem

We now present our proof of Theorem 3 in accordance with the correct formulation proposed in the previous subsection. We have retained Dubrova's notation from [3] as much as possible and elaborate only those steps where there are significant modifications.

Proof of Theorem 3:

We adhere to Dubrova's proof in the setting of the theorem as well as the relevant steps in [3] prior to the discrepancy. The transformation of the uniform NFSR N_a , with terminal bit a , into N_b , with terminal bit $b < a$, is effected by the shift of $x_k x_a$, $k < a$, from g_a to g_b . As the transformed N_b is uniform by hypothesis, and $b < a$, the feedback components $g_{n-1}, g_{n-2}, \dots, g_a^*$ of N_a depend on variables s_j with $j \leq b$. Accordingly, the state-variable values $s_{n-1}(t), \dots, s_0(t)$ of N_a are described in terms of the state-variable values $s_i(t-1)$ and the functions $g_j(s_0(t-1), s_1(t-1), \dots, s_b(t-1))$.

For instance,

$$s_{n-1}(t) = s_0(t-1) \oplus g_{n-1}(s_0(t-1), s_1(t-1), \dots, s_b(t-1)) \quad (6)$$

and the equation for the terminal bit of N_a is:

$$s_a(t) = s_{a+1}(t) \oplus g_a^*(s_0(t-1), s_1(t-1), \dots, s_b(t-1)) \oplus s_k(t-1)s_a(t-1) \quad (7)$$

Successive application of the substitutions $\text{sub}(v_i, v_{i+1})$ from the stage $i = 0$ to the stage $i = a - 1$, results in the expression of all the remaining state-variable values as follows ([3]).

$$\begin{aligned} s_{n-1}(t) &= s_a(t-a-1) \oplus g_{n-1}(s_a(t-a-1), s_a(t-a), \dots, s_a(t-1-(a-b))) \\ s_{n-2}(t) &= s_{n-1}(t-1) \oplus g_{n-2}(s_a(t-a-1), s_a(t-a), \dots, s_a(t-1-(a-b))) \\ &\dots = \dots \\ s_a(t) &= s_{a+1}(t-1) \oplus s_a(t-1-a+k)s_a(t-1) \\ &\quad \oplus g_a^*(s_a(t-a-1), s_a(t-a), \dots, s_a(t-1-(a-b))) \end{aligned}$$

The above sequence of substitutions results in replacing all $s_i(t)$, $i < a$, with time-shifted $s_a(t)$'s. Our next step involves the *proposed second substitution sequence* $\text{sub}(v_i, v_a)$ from the stage $i = n - 1$ to the stage $i = a + 1$.

The initial substitution in the second sequence, $\text{sub}(v_{n-1}, v_a)$, has already been computed at the end of the first substitution sequence. This is because, at the end of the first sequence, v_a is the unique predecessor of v_{n-1} and all the $s_i(t-1)$ have been replaced by $s_a(t-a-1+i)$ for $i \leq b < a$. Using the notation $\tilde{s}_a$ from Equation 4, we have:

$$s_{n-1}(t) = s_a(t-a-1) \oplus g_{n-1}(\tilde{s}_a)$$

Next we arrive at the expression for $s_{n-2}(t)$ as follows.

$$\begin{aligned} s_{n-2}(t) &= s_{n-1}(t-1) \oplus g_{n-2}(\tilde{s}_a) \\ &= s_a(t-a-2) \oplus g_{n-1}(\tilde{s}_a(-1)) \oplus g_{n-2}(\tilde{s}_a) \end{aligned}$$

Claim: At the i -th stage of the second substitution sequence, $i < n - a$, the state-variable value expression is given by:

$$s_{n-i}(t) = s_a(t-i-a) \oplus g_{n-1}(\tilde{s}_a(-i+1)) \oplus g_{n-2}(\tilde{s}_a(-i+2)) \oplus \dots \oplus g_{n-i}(\tilde{s}_a) \quad (8)$$

Proof of Claim: Proved by induction on i , where $1 \leq i \leq n - a - 1$.

Assuming Equation (8) holds for some $i = m - 1$, we have, for $i = m$,

$$s_{n-m}(t) = s_{n-m+1}(t-1) \oplus g_{n-m}(\tilde{s}_a)$$

Substituting the formula for $s_{n-m+1}(t-1)$ proves the claim.

Applying the above formula for $i = n - a - 1$, we have

$$\begin{aligned} s_{a+1}(t) &= s_a(t - (n - a - 1) - a) \oplus g_{n-1}(\tilde{s}_a(-n + a + 2)) \oplus \dots \oplus g_{a+1}(\tilde{s}_a) \\ &= s_a(t - n + 1) \oplus g_{n-1}(\tilde{s}_a(-n + a + 2)) \oplus g_{n-2}(\tilde{s}_a(-n + a + 3)) \oplus \\ &\quad \dots \oplus g_{a+1}(\tilde{s}_a) \end{aligned}$$

Therefore

$$\begin{aligned} s_a(t) &= s_{a+1}(t-1) \oplus s_a(t-1-a+k)s_a(t-1) \\ &\quad \oplus g_a^*(s_a(t-a-1), s_a(t-a), \dots, s_a(t-1-(a-b))) \\ &= s_a(t-n) \oplus g_{n-1}(\tilde{s}_a(-n+a+1)) \oplus g_{n-2}(\tilde{s}_a(-n+a+2)) \oplus \dots \\ &\quad \oplus g_{a+1}(\tilde{s}_a(-1)) \oplus g_a^*(\tilde{s}_a) \oplus s_a(t-1-a+k)s_a(t-1) \end{aligned}$$

Expanding the abbreviated notation for $\tilde{s}_a(-i)$ in each term we get:

$$\begin{aligned} s_a(t) &= s_a(t-n) \\ &\quad \oplus g_{n-1}(s_a(t-n), s_a(t-n+1), \dots, s_a(t-n+b)) \\ &\quad \oplus g_{n-2}(s_a(t-n+1), s_a(t-n+2), \dots, s_a(t-n+b+1)) \oplus \dots \quad (9) \\ &\quad \oplus g_a^*(s_a(t-a-1), s_a(t-a), \dots, s_a(t-1-(a-b))) \\ &\quad \oplus s_a(t-1-a+k)s_a(t-1) \end{aligned}$$

For the transformed NFSR N_b , the state-variable values are given by the following set of equations (cf. [3]):

$$\begin{aligned}
s_{n-1}(t) &= s_0(t-1) \oplus g_{n-1}(s_0(t-1), s_1(t-1), \dots, s_b(t-1)) \\
&\dots = \dots \\
s_a(t) &= s_{a+1}(t-1) \oplus g_a^*(s_0(t-1), s_1(t-1), \dots, s_b(t-1)) \\
&\dots = \dots \\
s_{a-1}(t) &= s_a(t-1) \\
&\dots = \dots \\
s_b(t) &= s_{b+1}(t-1) \oplus s_{k-a+b}(t-1)s_b(t-1) \\
&\dots = \dots \\
s_0(t) &= s_1(t-1)
\end{aligned}$$

It is noted that the product-term $x_k x_a$ has been shifted from $s_a(t)$ to $s_b(t)$ with the appropriate change in the indices: $k \rightarrow k - a + b, a \rightarrow b$. Using the notation ([3])

$$\tilde{s}_b := (s_b(t-b-1), s_b(t-b), \dots, s_b(t-1))$$

we obtain, as follows (cf. [3]), a similar set of equations after the first sequence of substitutions $\text{sub}(v_i, v_{i+1})$, for $i = 0, \dots, b-1$.

$$\begin{aligned}
s_{n-1}(t) &= s_b(t-b-1) \oplus g_{n-1}(\tilde{s}_b) \\
&\dots = \dots \\
s_a(t) &= s_{a+1}(t-1) \oplus g_a^*(\tilde{s}_b) \\
s_{a-1}(t) &= s_a(t-1) \\
&\dots = \dots \\
s_b(t) &= s_{b+1}(t-1) \oplus s_b(t-1-a+k)s_b(t-1)
\end{aligned}$$

where, in the last line, the argument in the first term of the product-term on the r.h.s. is obtained thus: $t-1-(b-(k-a+b)) = t-1-a+k$; i.e. it is *implicitly assumed that* $k-a+b \leq b$.

However, as demonstrated earlier for the NFSR N_a , the second sequence of substitutions to compute the recurrence for the bit b should be

$$\text{sub}(v_{n-1}, v_b), \text{sub}(v_{n-2}, v_b), \dots, \text{sub}(v_{b+1}, v_b)$$

With a development similar to Equation (8), we present the corrected form for $s_b(t)$ as follows.

$$\begin{aligned}
s_b(t) &= s_b(t-n) \oplus g_{n-1}(\tilde{s}_b(-n+b+1)) \oplus g_{n-2}(\tilde{s}_b(-n+b+2)) \oplus \dots \\
&\oplus g_a^*(\tilde{s}_b(-a+b)) \oplus s_b(t-1-a+k)s_b(t-1)
\end{aligned}$$

In the recurrence for $s_b(t)$, the term g_a^* is contributed from the stage a , and so, the argument of g_a^* becomes $\tilde{s}_b(-(a-b))$. This is in contrast with the term $g_a^*(\tilde{s}_a)$ which appears in the final stage of the recurrence for $s_a(t)$. However, substituting the full form of $\tilde{s}_b$ in the g_a^* term in $s_b(t)$ yields:

$$g_a^*(s_b(t-a-1), s_b(t-a), \dots, s_b(t-1-(a-b)))$$

Hence, expanding the notation for all the $\tilde{s}_b(-i)$'s in the expression of $s_b(t)$ we obtain:

$$\begin{aligned} s_b(t) = & s_b(t-n) \\ & \oplus g_{n-1}(s_b(t-n), s_b(t-n+1), \dots, s_b(t-n+b)) \\ & \oplus g_{n-2}(s_b(t-n+1), s_b(t-n+2), \dots, s_b(t-n+b+1)) \oplus \dots \quad (10) \\ & \oplus g_a^*(s_b(t-a-1), s_b(t-a), \dots, s_b(t-1-(a-b))) \\ & \oplus s_b(t-1-a+k)s_b(t-1) \end{aligned}$$

Comparing Equations (9) and (10), it is readily seen that the recurrence for the bit a in NFSR N_a is identical with that for b in NFSR N_b .

This completes the proof. $\square$

Remark 3. It is noted that the final expression of the recurrence for the bit a in the original NFSR, given as Equation (7) in [3] (p. 5268), and its abbreviated form using the notation $\tilde{s}_a$, are both **erroneous**. The argument of the term g_{n-2} is shown in Equation (7) to be:

$$s_a(t-n-1), s_a(t-n), \dots, s_a(t-n+b-1)$$

when the correct argument is (cf. Equation 9):

$$s_a(t-n+1), s_a(t-n+2), \dots, s_a(t-n+b+1)$$

The same error in the argument is present in the corresponding expressions for the bit b in the transformed NFSR (Equation (10), p. 5268 in [3]) as well, and the correct expression is presented here in Equation (10).

To conclude, we have established that there are serious errors in the proof of E. Dubrova's theorem in [3] and presented a correct proof. It is important to note that the statement of the theorem, which is a key result in numerous subsequent works by Dubrova and others, has been verified to be true - only the validation was flawed. In fact, the idea of transforming a uniform Galois NFSR via shifts of feedback functions and relating the notion of equivalence of NFSRs with equality of recurrence relations at designated stages have proved to be a powerful toolkit for analysis of shift register-based ciphers.

4 Unifying Dubrova and Yao-Parampalli Transformation Frameworks

As indicated in the introductory section, there is a strong connection between the transformation algorithms defined in the works of Dubrova ([3], [4], [6]) and those of Yao and Parampalli ([16],[17]). In this section, we attempt to formalize these connections in the context of transformations between Fibonacci and uniform Galois NFSRs. The proofs of most of the results used in this section have been presented earlier in [8].

4.1 Unifying Dubrova's Algorithms

In this subsection we state the evolutionary relation among the transformation algorithms discussed in [3] and [6]. We restrict our discussion to the type of NFSRs featured in the Espresso proposal as captured in the following definition.

Definition 8. *A uniform NFSR is termed a strict-terminal uniform NFSR, denoted st-uniform NFSR, if it satisfies the following condition for all stages indexed by $i > \tau$:*

$$\max_index(g_i) < \tau$$

where τ is the terminal bit.

For instance, the Espresso NFSR G has terminal bit $\tau = 193$ with the largest $\max_index(g_i) = 189$, while the Espresso alternate NFSR F has terminal bit $\tau = 217$, with $\max_index(g_i) \leq 183$. Hence they are both st-uniform NFSRs.

The first algorithm D1 in [3] effects a transformation between a uniform Fibonacci to a uniform, fully shifted Galois NFSR, which shifts all the monomials from the stage $n - 1$ to a stage with index greater than or equal to the terminal bit τ . The rules of this shifting are given in [3] as follows.

Let $\max_index(m)$ denote the largest index of variables in $\text{dep}(m)$; likewise $\min_index(m)$ denotes the smallest index. For all $m \in S_{g_{n-1}}$, where $S_{g_{n-1}}$ is the set of monomials of g_{n-1} ,

1. If $\min_index(m) \leq (n - 1) - \tau$, then the monomial m is shifted as

$$g_{n-1} \xrightarrow{m} g_{n-1-\min_index(m)}$$

2. If $\min_index(m) > (n - 1) - \tau$, then the monomial is shifted to g_τ .

The following result from [8] shows that the above method in the D1 algorithm for generating a fully shifted st-uniform Galois NFSR from a (uniform) Fibonacci NFSR, which preserves the output sequence at the 0-th stage, is consistent with a valid shifting in the D2 algorithm [6].

Theorem 4. *Let F be a Fibonacci NFSR with feedback functions of the form given in Equation 1, with the set of feedback monomials at the $(n-1)$ -th stage denoted by $\{x_0\} \cup S_{g_{n-1}}$. Then the shift of any monomial $m \in S_{g_{n-1}}$ under the algorithm D1 for st-uniform NFSRs is a valid shifting in the sense of the algorithm D2.*

Remark 4. Theorem 4 shows that for Fibonacci to st-uniform Galois transformation, D1 in [3] is subsumed in the framework of D2 in [6]. Hence the post-transformation equivalence conditions (cf. Theorem 2) that are satisfied for D2 will hold for D1 as well.

Dubrova’s Algorithms and the Espresso Equivalence Problem As discussed in the introductory section, an alternate Galois NFSR F is defined in [7] to discuss the security of Espresso, which is claimed to be “equivalent” to the cipher design. However, we point out that there are two aspects to any meaningful equivalence, in the context of security analysis, of the two NFSR configurations so defined.

1. Establishing the equivalence of the generated keystreams at designated stages of the 2-tap Galois NFSR F and the Espresso NFSR G .
2. Defining a corresponding non-linear output function for F , so that both systems generate *identical keystreams*. No such function is provided in [7].

These unresolved issues constitute the *Espresso Equivalence Problem*, graphically depicted in Figure 1.

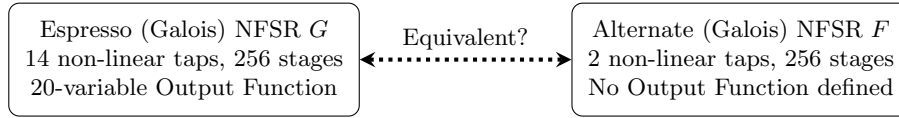


Fig. 1: The Espresso Equivalence Problem

In [7], the only pointer to establishing the claimed equivalence is a reference, on p. 278, to Dubrova’s algorithm D1 in [3] which we quote below:

“The equivalence of G and F can be shown by applying the Fibonacci-to-Galois transformation [6].”

In the light of Theorem 4, we now comment on the applicability of Dubrova’s algorithms to the Espresso equivalence problem using the following Table 2 (cf. [8]), which displays the relevant information culled from Table 1 in place for convenience. It is noted that Dubrova’s first algorithm in [3], while introducing several key notions of the shift formula, uniform NFSR configuration and associated terminal bit, does not provide explicit formulation for matching initial

state. Further, it guarantees equivalence across the transformation only for the output at a single stage (e.g. stage 0). Hence, it has been argued in [8] that, contrary to the claim in [7], the algorithm D1 cannot be used to resolve the Espresso equivalence problem, while it may be possible to use the algorithm D2 to gain further insights into the problem. Further analysis in later sections confirms this initial observation.

Algorithm	NFSR Types	Multi-stage Equivalence	Matching Initial States
D1 [3]	Fibonacci to Uniform Galois	No	No
D2 [6]	Galois to Galois	Yes	Yes

Table 2: Comparison of Dubrova’s NFSR Transformation Algorithms

4.2 Relating Yao-Parampalli and Dubrova’s Algorithms

As discussed in Subsection 2.2, the Yao-Parampalli transformation framework, henceforth denoted as the Y-P framework, analyzes four configurations of Galois NFSRs and back-and-forth transformations with “equivalent” Fibonacci NFSRs, when feasible. However, the only proven technical results in [17] deal with the transformation of a Fibonacci NFSR to a uniform Galois NFSR, which is the precisely the case treated in [3]. Accordingly, we compare the Y-P Fibonacci-to-uniform Galois algorithm with Dubrova’s generalized transformation D2 restricted to this particular case.

The three proven results in [17] are as follows:

- In Section 2 of [17], Lemma 1 and Lemma 2 relate the internal states of an “original Fibonacci NLFSR” with that of the “transformed Type-II Galois NLFSR”.
- The third result is Theorem 1 in Section 3 of [17], which states that for “an n -bit Fibonacci NLFSR transformed to a Type-II Galois NLFSR according to the Fibonacci-to-Type-II algorithm”, the internal states are related by the relations in Lemma 1 and Lemma 2, and, further, the NFSRs are “equivalent”.

It was proved in [8] that matching the initial states before and after the transformation in the Y-P framework yields the same result as outlined in Dubrova’s algorithm D2. In the following, we briefly state the relevant results in [8] and further extend the analysis.

Matching Initial States We begin by stating the two lemmas, proved in [17], and compare them with Dubrova’s Theorem 2. For facilitating this comparison, we use the following

Notation: Denote the k -th stage at time t of the transformed uniform Galois NFSR as r_k^t , and let s_k^t denote the corresponding entity for the Fibonacci NFSR. The complete states of the said NFSRs at time t are denoted with the boldface $\mathbf{r}^t$ and $\mathbf{s}^t$, respectively.

The lemmas establish that for the shift of a single monomial, using Dubrova's shift formula in [3], a relation between the initial states of the Fibonacci and the transformed uniform Galois NFSRs is preserved for all subsequent states.

Lemma 1 (Lemma 1, [17]). *Given an n -bit Fibonacci NFSR, a monomial m is shifted from f_{n-1} to f_b according to the shift operation in Definition 3 and the resulted feedback functions satisfy the conditions for uniform Galois NFSRs. If the initial state of the uniform Galois NFSR is computed according to (11), then the internal state of the uniform Galois NFSR $r_k^t, k \in \{0, \dots, n-1\}$ and the internal state of the Fibonacci NFSR $s_k^t, k \in \{0, \dots, n-1\}$ also satisfy this relation for $t > 0$.*

$$\begin{aligned} r_k &= s_k \oplus m|_{-(n-k)}(\mathbf{s}^t), \text{ for } k \in [b+1, n-1] \\ r_k &= s_k, \text{ for } k \in [0, b] \end{aligned} \quad (11)$$

The second lemma of the Y-P framework, which features iterative compensation, is stated as follows.

Lemma 2 (Lemma 2, [17]). *In the setting of Lemma 1,*

$$\begin{aligned} s_k &= r_k \oplus \overline{m|_{-(n-k)}}(\mathbf{r}^t), \text{ for } k \in [b+1, n-1] \\ s_k &= r_k, \text{ for } k \in [0, b] \end{aligned} \quad (12)$$

where $\overline{m|_{-(n-k)}}$ is m compensated by $C = [0, \dots, 0, m|_{-(a-b)}, \dots, m|_{-1}, 0, \dots, 0]$ iteratively in descending order.

The following result in [8] established the equivalence of matching the initial states in Y-P and Dubrova's formulations by proving the equivalence of Lemma 1 of [17] with Dubrova's theorem in [6].

Lemma 3. *For a Fibonacci-to-uniform Galois transformation, the formulation in [17] for the Galois NFSR stages r_k^t in terms of the Fibonacci stages $s_k^t, k \in \{0, \dots, n-1\}$, is simply obtained by a restriction of the formulation in Dubrova's Theorem in [6] to this particular case.*

It is noted that even though the results of matching the initial states are identical, when the transformed initial state is expressed in terms of the original state, there is no obvious counterpart of Lemma 2 in Dubrova's general framework. Further, the Y-P framework introduces the compensation operation to arrive at the results, which has no apparent equivalent in Dubrova's works. In the next subsection, we take a closer look at the actual analyses behind these results and reveal a stronger connection between the Dubrova and Y-P frameworks.

4.3 De-mystifying Compensation

In this subsection we first examine the two compensation operations defined in the Y-P framework in the context of Fibonacci-uniform Galois transformations, and show that they lead to the evaluation of the same matching initial states, albeit from different directions. Then we prove that the simple compensation operation is a re-formulation of a result from Dubrova's framework [4], which allows us to relate these two frameworks with greater clarity.

Simple vs. Iterative Compensation It is evident from the statements of Lemma 1 and Lemma 2 (cf. Lemma 1 and Lemma 2 of [17]) that the second lemma includes the first in its premise. We further make the following formal observation, which is obtained from the steps of the proof of the second lemma presented in [17].

Proposition 1. *For the Yao-Parampalli Fibonacci-to-uniform Galois transformation, the initial states before and after transformation can be matched in either of two ways, leading to identical results, as follows:*

1. *Start with the initial state of the original Fibonacci NFSR and perform simple compensation to obtain the matching initial state of the transformed Galois NFSR;*
2. *Start with the initial state of the transformed Galois NFSR and perform iterative compensation to obtain the matching initial state of the original Fibonacci NFSR.*

Proof. It follows from Equation 11 and Equation 12 that for indices $k \in [0, b]$, where $k = b + 1$ is the first possibly non-zero index in the compensation list, both the procedures return identical values by default.

For any $k \in [b + 1, n - 1]$, consider the expression for the k -th initial bit of the Fibonacci NFSR in terms of the corresponding bit of the transformed Galois NFSR obtained from Equation 12:

$$s_k^0 = r_k^0 \oplus \overline{m|_{-(n-k)}}(\mathbf{r}^0)$$

Here, the overline denotes iterative compensation using the initial state $\mathbf{r}^0$ of the Galois NFSR. But, as Lemma 2 pre-supposes Lemma 1, we substitute the value of r_k^0 from Equation 11 in the above to obtain:

$$s_k^0 = (s_k^0 \oplus m|_{-(n-k)}(\mathbf{s}^0)) \oplus \overline{m|_{-(n-k)}}(\mathbf{r}^0)$$

which readily implies

$$m|_{-(n-k)}(\mathbf{s}^0) = \overline{m|_{-(n-k)}}(\mathbf{r}^0) \quad (13)$$

This proves the assertion, as simple compensation using the initial state of the Fibonacci NFSR yields exactly the same values as iterative compensation using the state of the transformed Galois NFSR.

□

Remark 5. It is noted that the proof of Lemma 2 in [17] essentially consists of proving the relation in Equation 13 (cf. pp. 235-236 [17]).

Simple compensation and Dubrova's matching formula in [4] As discussed in Section 2.1, formulas for matching initial states in transformations between Fibonacci and uniform Galois NFSRs were presented in [4], [6]. Specifically, Theorem 1 in [4], gives the explicit formula for the initial state of a uniform Galois NFSR in terms of the initial state of an "equivalent" Fibonacci NFSR. In the following we show that, in this case, simple compensation in the Y-P framework is a re-formulation of the above formula in [4].

Theorem 5. *Let N_F and N_G be an equivalent pair, in the sense of matched outputs at stage 0, of an n -bit Fibonacci and a uniform Galois NFSR, respectively. Then matching the initial state $\mathbf{s} = (s_0, \dots, s_{n-1})$ of N_F to that of N_G , denoted $\mathbf{r} = (s_0, s_1, \dots, s_\tau, r_{\tau+1}, \dots, r_{n-1})$, through the Yao-Parampalli simple compensation operation yields the formula:*

$$r_k = s_k \oplus \bigoplus_{i=\tau}^{k-1} g_i|_{+(k-(i+1))}(\mathbf{s})$$

which matches exactly with the corresponding formula for r_k , $\tau + 1 \leq k \leq n - 1$, in (Dubrova's) Theorem 1.

Proof. Consider the case where the uniform Galois NFSR N_G has terminal bit τ , where $0 \leq \tau < n - 1$, and feedback functions of the form:

$$\begin{aligned} f_k(x) &= x_{(k+1) \bmod n} \oplus g_k(x_0, \dots, x_\tau); \quad k \in [\tau, n - 1] \\ f_k(x) &= x_{(k+1)}; \quad k \in [0, \tau - 1] \end{aligned}$$

Hence, the transformation entails the *shift* of all feedback functions, which themselves consist of XORed monomial terms, from the single Fibonacci feedback function at $k = n - 1$ of N_F . The terminal bit being τ , the first possibly non-zero entry in the compensation list corresponds to the shift from $n - 1$ to τ , with the *shifted form* being g_τ . We denote the original component of the feedback at the $(n - 1)$ -th stage of N_F as g_τ^* , which is related to the shifted feedback function g_τ of N_G as follows:

$$g_\tau = g_\tau^*|_{-(n-1-\tau)} \implies g_\tau^* = g_\tau|_{+(n-1-\tau)}$$

The compensation list corresponding to the shift $f_{n-1} \xrightarrow{g_\tau^*} f_\tau$ is given by:

$$C_\tau = [0, \dots, 0, \underbrace{g_\tau^*|_{-(n-1-\tau)}}_{\tau+1}, g_\tau^*|_{-(n-2-\tau)}, \dots, \underbrace{g_\tau^*|_{-1}}_{n-1}]$$

Substituting g_τ for g_τ^* in the above, we have:

$$C_\tau = [0, \dots, 0, \underbrace{g_\tau}_{\tau+1}, g_\tau|_{+1}, \dots, \underbrace{g_\tau|_{+(n-1-(\tau+1))}}_{n-1}]$$

which gives the simple compensation contribution involving g_τ in the combined compensation list:

$$C = C_\tau \oplus C_{\tau+1} \oplus \dots \oplus C_{n-1}$$

It can be similarly verified that the compensation contribution involving g_k , for all $\tau + 1 \leq k \leq n - 1$, is given by:

$$C_k = [0, \dots, 0, \underbrace{g_k}_{k+1}, g_{k|+1}, \dots, \underbrace{g_{k|+(n-1-(k+1))}}_{n-1}]$$

The first possibly non-zero entry of the combined compensation list is the entry in the position $\tau + 1$, given by the single term g_τ .

The entry in the position $\tau + 2$ is accordingly given by: $g_{\tau+1} \oplus g_{\tau|+1}$.

Collecting the terms in a similar manner, the k -th entry in the combined compensation list is given by:

$$g_{k-1} \oplus g_{k-2|+1} \oplus \dots \oplus g_{\tau|+(k-(\tau+1))} = \bigoplus_{i=\tau}^{k-1} g_{i|+(k-(i+1))}$$

Simple compensation using this combined compensation list yields the k -th bit of the initial state of N_G , say r_k , in terms of the initial state $\mathbf{s}$ of N_F as:

$$r_k = s_k \oplus \bigoplus_{i=\tau}^{k-1} g_{i|+(k-(i+1))}(\mathbf{s}).$$

Hence the theorem. □

We summarize our analysis of the transformation frameworks of Dubrova and Yao-Parampalli with the following

Observations:

1. For Fibonacci-to-uniform Galois NFSR transformation, the simple compensation operation of Yao-Parampalli yields the same result as Dubrova's formulation for matching initial states to achieve NFSR equivalence.
2. In the Y-P framework for Fibonacci-uniform Galois transformation, simple compensation and iterative compensation yield identical pairs of matched initial states. However, simple compensation computes the transformed state in terms of the original state while iterative compensation proceeds in the reverse direction.
3. Therefore, with regard to equivalent transformations between Fibonacci and uniform Galois NFSRs, Dubrova's general framework ([4], [6]) and the Y-P framework are essentially identical in computing the monomial shifts and matching the initial states.
4. However, as will be elaborated on in Section 5, the real efficacy of iterative/simple compensation is demonstrated in matching the outputs of non-linear functions, involving several state variables of the NFSRs, before and after the transformations. This contribution in the Y-P framework appears to represent a significant step beyond Dubrova's framework.

5 Resolving the Espresso Equivalence Problem

In this section we bring to bear our unifying analysis of Dubrova and Y-P transformation techniques on the Espresso Equivalence Problem (EEP) (cf. Subsection 4.1). As mentioned earlier, Yao and Parampalli have claimed in [16] that the application of *uniform Galois-to-Fibonacci* NFSR transformation on the Espresso NFSR G results in a Fibonacci LFSR with feedback function

$$f_L(\mathbf{x}) := f_{255}(x) = x_0 \oplus x_{12} \oplus x_{48} \oplus x_{115} \oplus x_{133} \oplus x_{213} \quad (14)$$

This also generates an output function, which we denote as $z_{YP}(\mathbf{x})$, consisting of 2289 monomials and having algebraic degree 12, such that the ensemble comprising LFSR $f_L(\mathbf{x})$ and $z_{YP}(\mathbf{x})$, with initial state matched with G , produces *the same keystream as Espresso*. This is indeed a fascinating result, and it showcases the power of Dubrova-Yao-Parampalli transformation framework by reducing a Galois NFSR with 14 non-linear taps to a Fibonacci LFSR whose single feedback consists of 6 linear terms. The Y-P LFSR-based Espresso-equivalent design is the springboard for our attempt to resolve EEP.

Thus far, we have shown that the NFSR transformation effected by the Y-P Fibonacci-to-uniform Galois method is essentially the same as Dubrova's algorithm D2. In [8], the Fibonacci-to-uniform Galois algorithm was applied on the Y-P LFSR ($f_L(\mathbf{x})$) to attempt a transformation to the Espresso alternate NFSR F . The rationale for this was given in [8] as follows:

If the NFSR F is recovered, then, given the equivalence claimed in [16], the output function obtained using the Y-P algorithm on the LFSR ensemble would complete the gap in the Espresso NFSR F design.

A concept schematic of the above strategy is given in Figure 2.

5.1 Recovery of Espresso NFSR F from Y-P LFSR

We now present, for the sake of completeness, the steps of the recovery of the Galois NFSR F by the application of the Fibonacci-to-uniform Galois algorithm on the Fibonacci LFSR $f_L(\mathbf{x})$ as given in [8].

We first note that the NFSR F has two feedback taps given by:

$$\begin{aligned} f_{255}(\mathbf{x}) &= x_0 \oplus x_{12} \oplus x_{48} \oplus x_{115} \oplus x_{133} \oplus x_{213} \oplus x_{41}x_{70} \oplus x_{46}x_{87} \\ &\quad \oplus x_{52}x_{110} \oplus x_{55}x_{130} \oplus x_{62}x_{157} \oplus x_{74}x_{183} \oplus x_{87}x_{110}x_{130}x_{157} \\ f_{217}(\mathbf{x}) &= x_{218} \oplus x_3x_{32} \oplus x_8x_{49} \oplus x_{14}x_{72} \oplus x_{17}x_{92} \oplus x_{24}x_{119} \oplus x_{36}x_{145} \\ &\quad \oplus x_{49}x_{72}x_{92}x_{119} \end{aligned} \quad (15)$$

It is remarked in [7] that

$$f_{255}(\mathbf{x}) = f_{Lin}(\mathbf{x}) \oplus f_{NL}(\mathbf{x}) \quad (16)$$

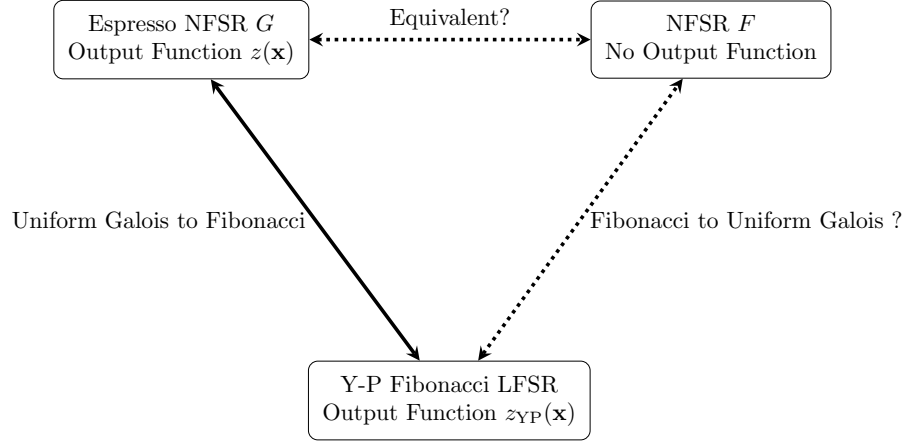


Fig. 2: Using Y-P/Dubrova Algorithms to Resolve Espresso Equivalence

where $f_{\text{Lin}}(\mathbf{x})$ is a linear polynomial component and $f_{\text{NL}}(\mathbf{x})$ is a non-linear component. Further the feedback function at stage 217 is given precisely by

$$f_{217}(\mathbf{x}) = x_{218} \oplus f_{\text{NL}}(\mathbf{x})_{(i-38)}$$

where the second term is given by the non-linear component $f_{\text{NL}}(\mathbf{x})$ of f_{255} , with every variable x_i changed to x_{i-38} .

The feedback function of the Y-P Fibonacci LFSR can be re-written as

$$f_L(\mathbf{x}) = (f_L(\mathbf{x}) \oplus f_{\text{NL}}(\mathbf{x})) \oplus f_{\text{NL}}(\mathbf{x})$$

Using Dubrova's shift formula to transfer all the monomials in $f_{\text{NL}}(\mathbf{x})$ from stage 255 to stage 217, we obtain the transformed feedback function at stage 217 as:

$$f_{217} = x_{218} \oplus f_{\text{NL}}(\mathbf{x})_{(i-255+217)} = x_{218} \oplus f_{\text{NL}}(\mathbf{x})_{(i-38)}$$

After the shift, the modified feedback function at stage 255 is

$$f_{255}(\mathbf{x}) = f_L(\mathbf{x}) \oplus f_{\text{NL}}(\mathbf{x})$$

This is precisely the Galois NFSR F with exactly two non-linear feedback taps at stages 255 and 217.

Remark 6. A general analysis and application of this 'trick' of extracting and shifting an 'invisible' feedback term by XORing twice is to be found in [5].

Proposed Resolution of Espresso Equivalence Problem As the NFSR F is a uniform Galois NFSR, with terminal bit 217, the Y-P Fibonacci-to-uniform

Galois algorithm will automatically generate a candidate for the non-linear output function. Hence, we can construct a non-linear filter generator cipher based on the Espresso alternate NFSR F , which would be *equivalent* to the Y-P LFSR ensemble in the sense discussed in [16], [17]. Our proposed procedure is formally stated in Algorithm 1. A theoretical validation and experimental results pertaining to this procedure are presented in the following subsections.

Algorithm 1 GENERATION OF ESPRESSO ALTERNATE NFSR CIPHER

- 1: **Input:** Y-P n -bit Fibonacci LFSR L and Non-linear function $z_{YP}(\mathbf{x})$; Initial state $\mathbf{s}$ of Fibonacci LFSR.
Output: Espresso alternate NFSR F and Non-linear function $z_F(\mathbf{x})$; Matching initial state $\mathbf{r}$ of NFSR F .
- 2: Use Dubrova's shift formula to generate NFSR F :

$$f_{255} \xrightarrow{f_{NL}(\mathbf{x})} f_{217}$$

- 3: Generate combined compensation list C for the shift.
 - 4: Obtain matching initial state for F by simple compensation using C (Dubrova's theorem/Y-P lemma).
 - 5: Obtain output function $z_F(\mathbf{x})$ from $z_{YP}(\mathbf{x})$ by iterative compensation in the descending order using C .
-

5.2 Validating the Solution to EEP

In this subsection we present theoretical analysis in support of our proposed procedure for resolving the Espresso Equivalence problem.

As stated in Algorithm 1, our procedure is based on using the Dubrova-Yao-Parampalli transformation framework to transform the Yao-Parampalli LFSR-based Espresso-equivalent design into a design based on the Espresso alternate NFSR F . It is evident from the analysis in Subsection 5.1, that we have achieved the LFSR-NFSR correspondence through appropriate use of Dubrova's shift formula. The final, more critical, challenge is to ensure that the output of the non-linear function so obtained is identical for both the Y-P LFSR-based design and the NFSR F -based design derived by us. To that end, we first examine the recipe(s) proposed by Yao-Parampalli in [17], to effect transformations between Fibonacci and uniform Galois NFSRs such that they preserve the output bitstreams of non-linear functions involving several stages of the NFSRs.

Issues with the Yao-Parampalli Algorithms As discussed in Section 2.2, there are three proven results in [17], Lemmas 1 and 2 along with the Theorem 6 (cf. p. 238, Section 3, [17]), all of which deal with the transformation of a Fibonacci to a uniform Galois NFSR. We have already discussed the lemmas,

and established their equivalence with Dubrova's works ([4], [6]), in computing matching initial states for the Fibonacci and Galois NFSRs. We now examine the following theorem of Yao-Parampalli which claims that their algorithm (cf. Section 2.2) achieves equivalence in the outputs of the respective non-linear functions.

Theorem 6 (Theorem 1, [17]). *Let an n -bit Fibonacci NLFSR [sic] transformed to a Type-II Galois NLFSR according to the Fibonacci-to-Type-II algorithm, then the internal states of the NLFSR before and after transformation denoted by x_i^t and $\hat{x}_i^t$ satisfy*

$$\begin{aligned}\hat{x}_i^t &= x_i^t \oplus c_i(X^t), \text{ for } k \in [\tau_1 + 1, n - 1] \\ \hat{x}_i^t &= x_i^t, \text{ for } k \in [0, \tau_1]\end{aligned}$$

and

$$\begin{aligned}x_i^t &= \hat{x}_i^t \oplus \overline{c_i}(\hat{X}^t), \text{ for } k \in [\tau_1 + 1, n - 1] \\ x_i^t &= \hat{x}_i^t, \text{ for } k \in [0, \tau_1]\end{aligned}$$

and the two NLFSRs are equivalent.

A comparison with Lemma 1 and Lemma 2 shows that the formulas in Theorem 6 have the same form with those presented in the lemmas, with single monomials in the former replaced by compensation entries in the latter. The main body of the proof of Theorem 6 consists of validating the stated formulas in simple instances. The crucial assertion in the theorem regarding “equivalence”, which appears to be the real contribution of the Y-P formulation beyond Dubrova's framework, is handled in the concluding paragraph of the proof in [17] as follows.

“If the output function f_z takes $x_i, i \in [\tau_1 + 1, n - 1]$ as input, the corresponding tap in $\overline{f_z}$ is compensated by C . Since the compensation is done iteratively, the x_i in c_i is also compensated. Therefore, at each clock t , the bit value of that input in $\overline{f_z}$ is equal to $\hat{x}_i^t \oplus \overline{c_i}(\hat{X}^t)$ which is equal to x_i^t according to (10) (cf. Theorem 6). Hence, the output function $\overline{f_z}$ generate [sic] same bit as the original function f_z does at each clock.”

We point out the following issues with the above statement and the steps of transformation from “Fibonacci-to-Type II” (uniform Galois) on p. 238 and “Type II-to-Fibonacci” on p. 241 in [17], respectively.

- It appears from the quoted statement that the equivalence of the outputs of the non-linear functions before and after the transformation is achieved via the following steps:
 1. Perform iterative compensation on the variables constituting the output function f_z to obtain $\overline{f_z}$.
 2. Perform iterative compensation on the original initial state to obtain $\hat{x}_i^t \oplus \overline{c_i}(\hat{X}^t)$ (following the notation in Lemma 2).

3. Evaluate the iteratively compensated output function $\overline{f_z}$ on the iteratively compensated initial state.

Issue: It is simply stated, without proof, that this evaluation is identical with the original output function evaluated on the original initial state.

- As shown in Subsection 2.2, the steps for “Fibonacci-to-Type II” transformation in [17] consist of the following sequence for producing identical function outputs: iterative compensation of output function followed by simple compensation of the initial state to be used as argument.

Issue: This is at variance with the suggested procedure in the “proof” of Theorem 6.

- The stated procedure for “Type II-to-Fibonacci” in [17] transformation suggests simple compensation of output function followed by iterative compensation of the initial state to be used as argument.

Issue: In this case, swapping the sequence of compensation operation that was suggested for the earlier case of “Fibonacci-to-Type II” adds to the ambiguity. More so, as no reason or validation is provided for either.

Remark 7. It is noted in this context that the proof of Theorem 1 as well as that of Lemma 2 in [17] refers to a “Lemma 3” for some sort of validation. Unfortunately, the said result does not exist in the entire paper, even though it is referred to elsewhere as well.

Remark 8. In [17], the authors present an illustration of their algorithm for Fibonacci-to-uniform Galois (Type-II) transformation (cf. Algorithm 2, Section 3, p. 240). They begin with a 4-bit Fibonacci NFSR with initial state $X^0 = \{x_0^0, x_1^0, x_2^0, x_3^0\} = \{1, 1, 0, 1\}$. However, in the later steps of the algorithm, specifically from Step 5, the initial state is specified as $\{0, 1, 1, 1\}$. No reason is given for this modification, whatsoever. In the following steps, simple compensation of the modified state $\{0, 1, 1, 1\}$ is used to generate $\{0, 1, 0, 0\}$, rather than the declared initial state.

Resolving the Issues and Proof of Concept In what follows we resolve the above issues in the Yao-Parampalli algorithms to establish the correct sequence of operations of simple and iterative transformations. Further, we formally prove that our prescribed sequence indeed achieves the following objective of Yao-Parampalli’s algorithms:

The modified non-linear function obtained after transformation, with a modified input state as the argument, will output exactly the same bitstream as the original function with a matched input state as argument.

We begin with a result on the composition of the dependence lists of the compensation entries corresponding to Fibonacci-to-uniform Galois NFSR transformations. This is an ingredient in the proof of the main theorem which will provide theoretical validation to the claimed equivalence.

Lemma 4. *For the D-Y-P equivalent transformation from an n -bit Fibonacci to a uniform Galois NFSR, with terminal bit $0 \leq \tau < n-1$, the entries c_i of the compensation list C satisfy the following condition:*

$$\max \text{dep}(c_i) \leq i-1, \forall i \in \{\tau+1, \tau+2, \dots, n-1\}$$

Proof. Consider the following shift from the feedback function at the $(n-1)$ -th stage of the Fibonacci NFSR:

$$f_{n-1} \xrightarrow{g_\tau^*} f_\tau$$

where we use the notation of Theorem 5, with the resulting shifted function at f_τ denoted by g_τ . As the transformed Galois NFSR, defined using the notation of Theorem 1, is uniform, we have:

$$\text{dep}(g_i) \subseteq [0, \tau], \forall i \in \{\tau+1, \tau+2, \dots, n-1\}$$

At some such i , it follows from Theorem 1 and Theorem 5 that the compensation expression is defined by the terms:

$$g_{i-1}, g_{i-2}|_{+1}, \dots, g_\tau|_{+(i-\tau-1)}$$

The first possibly non-zero compensation term is $c_{\tau+1}$, which is simply given by g_τ . Hence, it follows that, for $i = \tau+1$,

$$\max \text{dep}(c_i) = \max \text{dep}(c_{\tau+1}) = \tau \leq i-1$$

For $i = \tau+2$, we have the contributing terms $g_{\tau+1}$ and $g_\tau|_{+1}$ where

$$\text{dep}(g_{\tau+1}) \subseteq [0, \tau], \text{ while } \text{dep}(g_\tau|_{+1}) \subseteq [1, \tau+1]$$

Hence, the assertion holds for $i = \tau+2$ as well, since $\max \text{dep}(c_{\tau+2}) \leq \tau+1$. Further, it is readily observed that for any i , $\max \text{dep}(c_i)$ is determined by the term involving g_τ , and so, we have:

$$\max \text{dep}(c_i) \leq \text{dep}(g_\tau|_{+(i-\tau-1)}) \leq (\tau + i - \tau - 1) = i-1.$$

This proves the assertion. □

We now present the main theorem of this subsection, which explicitly confirms the design of the transformed non-linear output function as well as the choice of input which achieves equivalence of the output bitstreams before and after a Fibonacci-uniform Galois transformation. Some new notation is introduced for this purpose as follows.

Let a function F in, say, n Boolean variables $x_0, \dots, x_{n-1}$, be denoted as: $F(X) = F(x_0, \dots, x_{n-1})$. If these variables are iteratively compensated (in descending order of the indices) by some array $C = [c_0, c_1, \dots, c_{n-1}]$, where each c_i is a Boolean function in X , the form obtained from F is termed the iteratively

compensated F . We denote the iteratively compensated function with the superscript ι as F^ι . Similarly, if the variables are compensated by direct XORing with the entries in C - what has been termed simple compensation - the compensated function form has the superscript σ : F^σ .

If the function F is evaluated at some modified value of X , where each x_i component has been iteratively compensated as described above, the evaluation is denoted as $F(X^\iota)$, with the analogous form for simple compensation of X denoted by $F(X^\sigma)$.

A combination of these compensation operations, where both the function and the argument have been compensated, will carry both the superscripts. For instance, an iteratively compensated function F evaluated on simple compensated argument X will be denoted by $F^\iota(X^\sigma)$.

Using the above notation, we now state a result which settles the question of the correct choice of the compensation operations for the output function and initial state to achieve matching output bitstreams for Fibonacci to uniform Galois NFSR transformation.

Theorem 7. *For Fibonacci-to-Uniform Galois NFSR transformations according to the D-Y-P framework, any choice of compensation for the output function and the initial state will produce matching output bitstreams before and after transformation. In particular, denoting the Fibonacci output function as f and the Fibonacci initial state as X , we have, using the notation defined above, the following equalities:*

$$f^\iota(X^\sigma) = f^\sigma(X^\iota) = f^\iota(X^\iota) = f^\sigma(X^\sigma) = f(X)$$

Proof. We prove the theorem in two parts as follows:

1. $f^\iota(X^\sigma) = f^\sigma(X^\iota) = f(X)$;
2. $f^\iota(X^\iota) = f(X)$ and $f^\sigma(X^\sigma) = f(X)$.

Part 1.: Consider $X_1 = (x_0, x_1)$, where the subscript corresponds to the maximum index in the list of variables. By Lemma 4, the corresponding entries of the compensation list are specified as follows: c_0 is a constant and $c_1 = c_1(x_0)$. Therefore, we have:

$$\begin{aligned} f^\sigma(X_1) &= f(x_0 \oplus c_0, x_1 \oplus c_1(x_0)) \\ f^\sigma(X_1^\iota) &= f((x_0 \oplus c_0) \oplus c_0, (x_1 \oplus c_1(x_0 \oplus c_0)) \oplus c_1(x_0 \oplus c_0)) \\ &= f(x_0, x_1). \end{aligned}$$

Similarly,

$$\begin{aligned} f^\iota(X_1) &= f(x_0 \oplus c_0, x_1 \oplus c_1(x_0 \oplus c_0)) \\ f^\iota(X_1^\sigma) &= f((x_0 \oplus c_0) \oplus c_0, (x_1 \oplus c_1(x_0)) \oplus c_1((x_0 \oplus c_0) \oplus c_0)) \\ &= f(x_0, x_1 \oplus c_1(x_0) \oplus c_1(x_0)) \\ &= f(x_0, x_1). \end{aligned}$$

Hence, we have: $f^\sigma(X_1^\iota) = f^\iota(X_1^\sigma) = f(X_1)$. We prove the result by induction on i , where $X_i = (x_0, x_1, \dots, x_i)$.

Assume, as induction hypothesis, that the result holds for all $i \leq k-1$. We will prove the result holds for $i = k$, i.e.

$$f^\sigma(X_k^\iota) = f^\iota(X_k^\sigma) = f(X_k). \quad (17)$$

The form of the function is compensated first and then the argument is compensated prior to evaluation. Hence, it can be verified that

$$f^\iota(X_{k-1}^\sigma) = f((x_0^\iota)^\sigma, \dots, (x_{k-1}^\iota)^\sigma)$$

with an analogous expansion for $f^\sigma(X_{k-1}^\iota)$.

Either way, the result in Equation 17 is proved if the following holds, making use of the induction hypothesis:

$$(x_k^\sigma)^\iota = (x_k^\iota)^\sigma = x_k \text{ given } (X_i^\sigma)^\iota = (X_i^\iota)^\sigma = X_i, \forall i \leq k-1.$$

We have:

$$\begin{aligned} (x_k^\sigma)^\iota &= (x_k \oplus c_k(X_{k-1}))^\iota \\ &= x_k^\iota \oplus c_k(X_{k-1}^\iota) \\ &= x_k \oplus c_k(X_{k-1}^\iota) \oplus c_k(X_{k-1}^\iota) \\ &= x_k. \end{aligned}$$

On the other hand,

$$\begin{aligned} (x_k^\iota)^\sigma &= (x_k \oplus c_k(X_{k-1}^\iota))^\sigma \\ &= (x_k \oplus c_k(X_{k-1})) \oplus c_k((X_{k-1}^\iota)^\sigma) \\ &= x_k \oplus c_k(X_{k-1}) \oplus c_k(X_{k-1}) \text{ (by induction)} \\ &= x_k. \end{aligned}$$

Hence the equalities in Equation 17 hold.

Part 2.: Consider evaluation of $f^\iota(X_1^\iota)$ in two stages as follows. First we compute the form of the iteratively compensated function (in descending order):

$$f^\iota(X_1) = f(x_0^\iota, x_1^\iota) = f(x_0 \oplus c_0, x_1 \oplus c_1(x_0 \oplus c_0))$$

Next we evaluate the above compensated function on an iteratively compensated argument, i.e. replacing all the variables in the above expression with iteratively compensated variable forms *in descending order*. Therefore, we have:

$$\begin{aligned} f^\iota(X_1) &\xrightarrow{x_1 \rightarrow x_1^\iota} f(x_0 \oplus c_0, (x_1 \oplus c_1(x_0 \oplus c_0)) \oplus c_1(x_0 \oplus c_0)) = f(x_0 \oplus c_0, x_1) \\ &\xrightarrow{x_0 \rightarrow x_0^\iota} f((x_0 \oplus c_0) \oplus c_0, x_1) = f(x_0, x_1) \\ &= f^\iota(X_1^\iota). \end{aligned}$$

In the same manner as computed above for $k = 1$, the first pass involving the iterative compensation of the argument in $f^\iota(X_k)$, for any positive integer $k \leq n - 1$, will be:

$$f^\iota(X_k) \xrightarrow{x_k \rightarrow x_k^\iota} f(X_{k-1}^\iota, (x_k \oplus c_k(X_{k-1}^\iota)) \oplus c_k(X_{k-1}^\iota)) = f(X_{k-1}^\iota, x_k).$$

Therefore, the successive passes for the iterative compensation of the argument will clearly return

$$f^\iota(X_k^\iota) = f(X_k).$$

Finally, we have:

$$\begin{aligned} f^\sigma(X_1^\sigma) &= f(x_0^\sigma \oplus c_0, x_1^\sigma \oplus c_1) \\ &= f((x_0 \oplus c_0) \oplus c_0, (x_1 \oplus c_1) \oplus c_1) \\ &= f(x_0, x_1) \end{aligned}$$

which is readily extended from $k = 1$ to any positive integer $k \leq n - 1$:

$$f^\sigma(X_k^\sigma) = f(X_k).$$

Hence the theorem. □

In view of the results in Theorem 7, we now have sufficient clarity as regards the D-Y-P Fibonacci-to-uniform Galois NFSR transformation for choosing either iterative or simple compensation for the original output function as well as the original initial state to produce matched output bitstreams post transformation.

Our experimental results in the next subsection further corroborate the statement of Theorem 7, which removes all ambiguity regarding the steps of this particular NFSR transformation.

5.3 Experimental Results

In [16], a 256-bit Fibonacci LFSR was derived from the Espresso design NFSR G using (Dubrova-Yao-Parampalli) uniform Galois-to-Fibonacci transformation. The transformation generated a counterpart to the non-linear output function defined for Espresso which had 2289 monomials in 104 variables and degree 12. It was claimed that this LFSR/non-linear function ensemble, following the recipe of simple compensation of the output function and iterative compensation of the initial state ([16], [17]), generated identical bitstreams with Espresso. We have experimentally verified this claim, which is depicted in the left halves of the schematics in Figure 2 and Figure 3.

Building on the verified claim in [16], we have successfully used the D-Y-P Fibonacci-to-uniform Galois transformation on the LFSR-based ensemble to generate a non-linear output function corresponding to the Espresso alternate Galois NFSR F (cf. Algorithm 1). Our non-linear function consists of 3656 monomials

in 88 variables and has algebraic degree 18. This NFSR F /non-linear function ensemble generates identical output bitstream, using iterative compensation of the output function and simple compensation of the initial state (cf. [17]), with the Fibonacci LFSR/non-linear function ensemble in [16].

Further, in accordance with the results in Theorem 7, we have verified that the output bitstream of our NFSR F -based ensemble matches the Yao-Parampalli LFSR-based ensemble using both simple and iterative compensation separately on their output function and the LFSR initial state.

We have, therefore, positively resolved the Espresso Equivalence Problem as stated in Section 4 (cf. Figure 1), by generating a non-linear output function corresponding to the Espresso alternate NFSR F which ‘inherits’ the equivalence with the original Espresso design through the following steps:

1. Espresso Galois NFSR G ensemble is output-equivalent to Yao-Parampalli Fibonacci LFSR ensemble
2. Yao-Parampalli Fibonacci LFSR ensemble is output-equivalent to Espresso NFSR F ensemble (this paper).
3. Therefore, Espresso Galois NFSR G ensemble is output-equivalent to Espresso NFSR F ensemble.

This is graphically depicted in Figure 3.

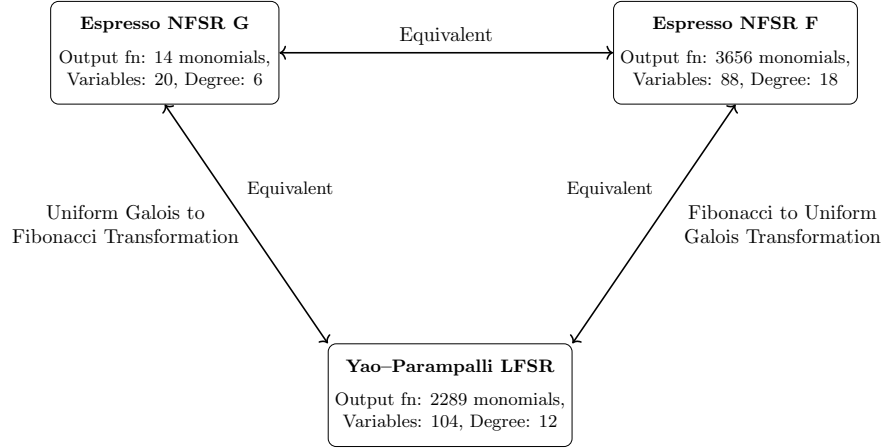


Fig. 3: Resolving the Espresso Equivalence Problem

Conclusion and Future Work

The design methodology of the stream cipher Espresso is theoretically significant as well as of practical interest as it attempts to harness the benefits of a Ga-

lois NFSR-based implementation while promising simpler analysis through an equivalent design. But the validity of its security claims against several standard cryptanalytic techniques rested on the equivalence of two designs discussed in [7]. We have termed this the Espresso Equivalence Problem and have demonstrated the first instance for its positive resolution.

In order to demonstrate the said equivalence, we have attempted a partial unification of E. Dubrova and Yao-Parampalli’s transformation techniques. The insights obtained from the above analysis led us to recover the Espresso alternate NFSR F from the Y-P LFSR, and, finally, a matching non-linear function for F . This has opened up the possibility of revisiting the security of Espresso through a proper analysis of the now truly output-equivalent alternate NFSR F and non-linear function ensemble. However, it remains an open question whether the two output-equivalent versions will provide any significant advantage in analysis, as evident from the following list of properties:

- **Espresso:** [7] Galois NFSR G with 14 non-linear taps; Output function: 14 monomials, 20 variables, algebraic degree 6.
- **Y-P Design:** [16] Fibonacci LFSR; Output function: 2289 monomials, 104 variables, algebraic degree 12.
- **Espresso Equivalent:** (this paper) Galois NFSR F with 2 non-linear taps; Output function: 3656 monomials, 88 variables, algebraic degree 18.

In course of establishing the Espresso equivalence, we have demonstrated that (cf. Theorem 7), for the Fibonacci-to-uniform Galois transformation, either compensation procedure applied to the output function-initial state would suffice to match the output. This assertion has also been verified by our experiments. However, this begs the obvious question:

Why is the sequence of compensation operations applied in [17], with respect to Fibonacci-to-uniform Galois transformation, to the output function and the initial state, *interchanged* for the uniform Galois-to-Fibonacci case?

The answer to the above question may lie in formulating and proving analogous results of Lemma 1, Lemma 2, and Theorem 6 for the case of uniform Galois-to-Fibonacci case. Further, in addition to the Fibonacci-uniform Galois transformation, three other NFSR designs have been discussed in [17]. However, as stated earlier, rigorous proof of concept has only been provided for Fibonacci-to-uniform Galois case. Proving (or disproving) the (output) equivalence in all these cases would be a worthwhile research direction.

Recently, there is a growing body of research on NFSR equivalence, for instance, [18], [19] and [13]. In [19], the authors claim to establish that the 288-stage Galois NFSR of Trivium cannot be shown to be equivalent to a 288-stage Fibonacci NFSR. We hope that further investigation would lead to the creation of a general template for equivalence-preserving shift register transformations. Such a template, if rigorously formulated, would open up the exciting possibility of obtaining equivalent circuits for existing designs of proven security like Grain and Trivium, and possibly identify variants with optimized implementation.

Acknowledgement The assistance of Debendranath Das with initial experimental verifications and the inputs from Subhamoy Maitra are gratefully acknowledged. The work of Anirban Ghatak is supported by SAG, DRDO, Delhi, and R. C. Bose Centre for Cryptology and Security, ISI, Kolkata.

References

1. Carlet, C., Sarkar, P.: The nonlinear filter model of stream cipher redivivus. Cryptology ePrint Archive, Paper 2025/160 (2025), <https://eprint.iacr.org/2025/160>
2. De Cannière, C.: Trivium: A stream cipher construction inspired by block cipher design principles. In: Katsikas, S.K., López, J., Backes, M., Gritzalis, S., Preneel, B. (eds.) *Information Security*. pp. 171–186. Springer Berlin Heidelberg, Berlin, Heidelberg (2006). https://doi.org/10.1007/11836810_13
3. Dubrova, E.: A Transformation From the Fibonacci to the Galois NLFSRs. *IEEE Transactions on Information Theory* **55**(11), 5263–5271 (2009). <https://doi.org/10.1109/TIT.2009.2030467>
4. Dubrova, E.: Finding matching initial states for equivalent NLFSRs in the Fibonacci and the Galois configurations. *IEEE transactions on information theory* **56**(6), 2961–2966 (2010). <https://doi.org/10.1109/TIT.2010.2046250>
5. Dubrova, E.: A Scalable Method for Constructing Galois NLFSRs With Period 2^{n-1} Using Cross-Join Pairs. *IEEE Transactions on Information Theory* **59**(1), 703–709 (2012). <https://doi.org/10.1109/TIT.2012.2214204>
6. Dubrova, E.: An equivalence-preserving transformation of shift registers. In: *International Conference on Sequences and Their Applications*. pp. 187–199. Springer (2014). https://doi.org/10.1007/978-3-319-12325-7_16
7. Dubrova, E., Hell, M.: Espresso: A stream cipher for 5G wireless communication systems. *Cryptography and Communications* **9**, 273–289 (2017). <https://doi.org/10.1007/s12095-015-0173-2>
8. Ghatak, A., Chattopadhyay, A., Awasthi, A., Gupta, I.: Analyzing non-linear shift register transformations in the design and cryptanalysis of espresso. In: Karfa, C., Asadi, N., Chattopadhyay, A. (eds.) *Security, Privacy, and Applied Cryptography Engineering*. pp. 171–190. Springer Nature Switzerland, Cham (2026). https://doi.org/10.1007/978-3-032-16342-4_10
9. Golomb, S.W.: *Shift register sequences: secure and limited-access code generators, efficiency code generators, prescribed property generators, mathematical models*. World Scientific (2017). <https://doi.org/10.1142/9361>
10. Hamann, M., Krause, M., Meier, W.: LIZARD - A Lightweight Stream Cipher for Power-constrained Devices. *IACR Transactions on Symmetric Cryptology* **2017**(1), 45–79 (Mar 2017). <https://doi.org/10.13154/tosc.v2017.i1.45-79>
11. Hell, M., Johansson, T., Maximov, A., Meier, W.: The Grain family of stream ciphers, pp. 179–190. Springer, Germany (2008). https://doi.org/10.1007/978-3-540-68351-3_14
12. Mansouri, S.S., Dubrova, E.: An Improved Hardware Implementation of the Grain Stream Cipher. In: *13th Euromicro Conference on Digital System Design: Architectures, Methods and Tools*. pp. 433–440 (2010). <https://doi.org/10.1109/DSD.2010.49>
13. Pan, Y., Zhong, J., Lin, D.: The equivalence between Galois and Fibonacci NF-SRs. *Theoretical Computer Science* **1003** (2024). <https://doi.org/10.1016/j.tcs.2024.114620>

14. Shi, Z., Chen, C., Yang, G., Xiong, H., Li, F., Hu, H., Wan, Z.: Design Space Exploration of Galois and Fibonacci Configuration Based on Espresso Stream Cipher. *ACM Trans. Reconfigurable Technol. Syst.* **16**(3) (Jun 2023). <https://doi.org/10.1145/3567428>
15. Yao, G., Paramalli, U.: Improved Transformation Algorithms for Generalized Galois NLFSRs. In: *Sequences and Their Applications SETA 2020* (2020)
16. Yao, G., Paramalli, U.: Cryptanalysis of the class of maximum period Galois NLFSR-based stream ciphers. *Cryptography and Communications* **13**, 847–864 (2021). <https://doi.org/10.1007/s12095-021-00511-0>
17. Yao, G., Paramalli, U.: Improved transformation algorithms for generalized Galois NLFSRs. *Cryptography and Communications* **14**(2), 229–258 (2022). <https://doi.org/10.1007/s12095-021-00500-3>
18. Zhao, X.X., Qi, W.F., Zhang, J.M.: Further results on the equivalence between Galois NFSRs and Fibonacci NFSRs. *Designs, Codes and Cryptography* **88**, 153–171 (2020). <https://doi.org/10.1007/s10623-019-00677-y>
19. Zhong, J., Pan, Y., Kong, W., Lin, D.: Necessary and Sufficient Conditions for Galois NFSRs Equivalent to Fibonacci Ones and Their Application to the Stream Cipher Trivium. *Cryptology ePrint Archive* (2021), <https://eprint.iacr.org/2021/928>