AKIŞ ŞİFRELEME ALGORİTMALARI KULLANILARAK RASGELE SAYI ÜRETİLMESİ VE FPGA ORTAMINDA GERÇEKLEŞTİRİLMESİ

Esra ERKEK

Yüksek Lisans Tezi
Bilgisayar Mühendisliği Anabilim Dalı

Danışman: Yrd. Doç. Dr. Taner TUNCER

HAZİRAN-2015

T.C
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

AKIŞ ŞİFRELEME ALGORİTMALARI KULLANILARAK RASGELE SAYI ÜRETİLMESİ VE FPGA ORTAMINDA GERÇEKLEŞTİRİLMESİ

YÜKSEK LİSANS TEZİ

Esra ERKEK

(121129119)

Anabilim Dalı: Bilgisayar Mühendisliği

Programı: Donanım

Danışman: Yrd. Doç. Dr. Taner TUNCER

Tezin Enstitüye Verildiği Tarih: 01.06.2015

HAZİRAN-2015

T.C
FIRAT ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

AKIŞ ŞİFRELEME ALGORİTMALARI KULLANILARAK RASGELE SAYI ÜRETİLMESİ VE FPGA ORTAMINDA GERÇEKLEŞTİRİLMESİ

YÜKSEK LİSANS TEZİ

Esra ERKEK

(121129119)

Tezin Enstitüye Verildiği Tarih : 01.06.2015
Tezin Savunulduğu Tarih : 22.06.2015

Tez Danışmanı : **Yrd. Doç. Dr. Taner TUNCER (Fırat Üniversitesi)**

Diğer Jüri Üyeleri : **Prof. Dr. Ali KARCI(İnönü Üniversitesi)**

Prof. Dr. Mehmet KAYA(Fırat Üniversitesi)

HAZİRAN-2015

TEŞEKKÜR

Bu çalışmada değerli vaktini bana ayırarak çalışmamın bitirilmesinde her türlü desteğini esirgemeyen sayın danışman hocam Yrd. Doç. Dr. Taner TUNCER'e teşekkürlerimi sunmak istiyorum. Danışmanım olarak her konuda benim için harcadığı zaman ve çabalardan ötürü tekrar teşekkür ediyorum.

Ayrıca hayatımın her anında ilgi, anlayış ve her türlü desteğini esirgemeyen aileme çok teşekkür ediyorum.

Esra ERKEK
ELAZIĞ - 2015

İÇİNDEKİLER

Sayfa No

ÖZET

AKIŞ ŞİFRELEME ALGORİTMALARI KULLANILARAK RASGELE SAYI ÜRETİLMESİ VE FPGA ORTAMINDA GERÇEKLEŞTİRİLMESİ

Bu tez, simetrik şifreleme algoritmalarından akış şifreler ve blok şifreler ile ilgilidir. Akış şifrelerin tasarım yapıları, rassal sayı üretimi, rassallığın test edilmesi ve test kriterleri incelenmiş, akış şifreleme algoritmaları FPGA(Field Programmable Gate Array)'da donanımsal olarak gerçekleştirilmiş ve test sonuçlarına yer verilmiştir

Tezin ikinci bölümünde temel şifreleme yapıları olan blok ve akış şifrelerinin yapısının tanımı yapılmıştır. Akış şifrelerin tasarım mimarilerini ve matematiksel alt yapısını inceleyen şifreleme kavramları anlatılmıştır. 3. bölümünde akış şifreleme algoritmalarında kullanılan rassal sayı üreteçleri incelenmiştir. 4. Bölümde ise kriptografik uygulamalarda kullanılan rassal sayıların ve bu sayılar kullanılarak geliştirilen anahtarların güvenilirliğini sağlamada önemli kriter oluşturan istatistiksel testler incelenmiştir. Rassallık için günümüzde pek çok uygulamada kullanılan NIST (National Institute of Standards and Technology - Ulusal Standartlar ve Teknoloji Enstitüsü) test paketinden faydalanılmış ve bu bölümde NIST test paketinde bulunan testlerin matematiksel alt yapıları incelenmiştir. 5. bölümde, seçilen akış şifreleme algoritmalarının çalıştırılmasıyla elde edilen anahtar değerlerinin NIST test paketi programına uygulanmasına ve sonuçlarının değerlendirilmesine yer verilmiştir Son olarak 6. Bölümde ise tezde elde edilen sonuçların değerlendirilmesi yapılmıştır.

Anahtar Sözcükler: Akış Şifreler, Blok Şifreler, Rassal Sayı Üreteçleri, NIST (National Institute of Standards) Testleri

ABSTRACT

RANDOM NUMBER GENERATION USING STREAM CIPHER ALGORİTHMS AND IMPLEMENTATION ON THE FPGA ENVIRONMENT

This thesis is about the flow through symmetrical encryption algorithms, ciphers and block ciphers. The design structure of stream ciphers, testing of randomness and the testing criteria were analyzed, stream encryption algorithms were carried in the FPGA (Field Programmable Gate Array) as hardware and testing results were included.

In the second chapter of the thesis, fundamental encryption patterns that are block and stream ciphers were described. The concepts of encryption which analyze the design architecture and mathematical infrastructure of flow encryption were described. In the 3rd section, the random number generators used in stream encryption algorithms were analyzed. As for the 4th section, statistical tests that form an important criteria in the provision of the reliability of random numbers and ciphers that are developed by using these numbers. For randomness, NIST (National Institute of Standards and Technology) test pack, which is being used in many applications in the present, was benefited from and in this chapter, mathematical infrastructures that are present in the NIST test pack were analyzed. In the 5th chapter, the application of ciphers that are obtained by the operation of the stream encryption algorithms to the NIST test pack and the evaluation of the results were included. Finally, in the 6th chapter, the results obtained from the thesis were evaluated.

Key Words: Stream Ciphers, Block Ciphers, Random Number Generators, NIST (National Institute of Standards) Tests

ŞEKİLLER LİSTESİ

Sayfa No

TABLOLAR LİSTESİ

Sayfa No

KISALTMALAR LİSTESİ

RSÜ : Rasgele Sayı Üreteçleri

GRSÜ : Gerçek Rasgele Sayı Üreteçleri

SRSÜ : Sözde Rasgele Sayı Üreteçleri

NIST : National Institute of Standards and Technology

FIPS : Federal Information Processing Standards

FPGA : Field-Programmable Gate Array

DES : Data Encryption Standards

AES : Advanced Encryption Standards

RSA : Rivest , Shamir, Adlemon

SHA : Secure Hash Algorithm

SPN : Substitution Permutation Networks

LFSR : Linear FeedBack Shift Register

IV :Initialization Vector

ASG :Alternating Step Generators

SG :Shrinking Generators

PRNG :Psuedo Random Number Generators

XOR :Exclusive Or

1. GİRİŞ

Günümüzde teknolojinin sürekli geliştiği ve çok hızlı bir şekilde gelişmeye devam edeceği bilinen bir gerçektir. Gelişen teknoloji ile birlikte veri iletimi ve iletilen verinin güvenliği önemli bir unsur haline gelmiştir. Şifreleme algoritmaları, iletilen verinin güvenliği için kullanılan yöntemlerdir. Bu algoritmaların teknolojinin gerektirdiği şekilde olacağı, teknolojiye ayak uyduramayanların kullanımının terk edileceği, yeni algoritmaların bulunacağı bilinen bir gerçektir.

Rastgele sayılara, dolayısıyla da rastgele sayı üreteçlerine olan ihtiyaç gelişen bu teknoloji sürecinde güvenli veri iletimi için çok önemli bir yere sahip olmaya başlamaktadır. Bu sayı üreteçlerine ihtiyaç günden güne artmaktadır. Rasgele sayılara özellikle bilgisayar derleyicileri, şifreleme sistemleri gibi bilgisayar biliminde ihtiyaç duyulmaktadır. Genellikle çekirdek adı verilen bir başlangıç değerine bazı sayısal işlemler uygulanmasıyla rastgele sayılar üretilmektedir. Rasgele sayılar; belli bir algoritma, matematiksel bir formül, önceden hesaplanmış tablolar kullanarak ya da deterministik karaktere sahip olmayan doğal fiziksel olaylar kullanılarak çeşitli şekillerde üretilmesi sağlanabilmektedir.

Rastgele sayı üretme işlemi aynı çekirdeği kullanıldığında tekrarlanabilir bir hal alabilir. Dolayısı ile rasgele sayı üretecinin çıkışı gerçek anlamda rastgele olmayabilir. Rastgele sayı dizileri, istatistiksel olarak birbirinden bağımsız ve aralarından korelasyon ilişkisi bulunmayan sayılardan oluştuğu için birçok alanda kullanılmaktadır. Rastgele sayı üreteçleri bilgisayar benzetimleri, sayısal analiz uygulamaları, istatistiksel analiz, Monte Carlo metodunun kullanıldığı uygulamalar ve özellikle şifreleme gibi alanlarda sıkça kullanılmaktadır. Örneğin, şifreleme algoritmalarının güvenilirliği, rastgele sayı üreteçlerinin ürettiği sayılara bağlıdır. Bu sayılar istatistiksel olarak rastgele ise, yani önceki çıkışlara bakarak daha sonrakiler tahmin edilemezse üretecin istatistiksel özelliği iyi demektir. Başka bir deyişle iyi bir şifreleme iyi bir Rasgele Sayı Üreteci (RSÜ) gerektirir. RSÜ'leri kendi aralarında gerçek ve sözde RSÜ'ler şeklinde ikiye ayırmak mümkündür. Uygulamanın amacına göre bu iki yapıdan biri tercih edilmektedir. Gerçek RSÜ'lerin çalışması gürültü gibi doğal süreçlerin ölçümüne dayanırken, sözde RSÜ'ler ise sayısal algoritmalar gibi deterministik süreçleri kullanmaktadır. Şifreleme gibi güvenliğin önemli olduğu uygulamalarda gerçek RSÜ'lerin kullanılması zorunlu iken, bilgisayar benzetimlerinde kullanılmak üzere sözde RSÜ'lerin başarımları yeterli olmaktadır.

Şifreleme teknikleri her türlü iletişim ve veri depolamada önemli bilgilerin güvenliğini sağlamak için kullanılır. En yaygın ve önemli uygulamalardan biri de kullanılan algoritmalar üzerinde aktarılan bilginin güvenliğini sağlamak için kullanılan şifreleme işlemleridir. Simetrik şifreler de bilgi güvenliğinin sağlanmasında önemli rol oynarlar. Bu güvenliğin sağlanmasında dolayısıyla rasgele sayı üreteçlerine ihtiyaç duyulmaktadır. Bu şifreleri blok ve akış şifreler olmak üzere iki ana kategoriye ayırabiliriz. Buna ek olarak güvenli şifreler tasarlamak da kriptolojinin en önemli konusudur. Akış şifreler ile ilgili olarak güvenliğin daha iyi anlaşıldığı gözlenmektedir. Diğer yandan, akış şifrelerin yanısıra giderek blok şifreler daha ön plana çıkmaktadır ve bu yüzden akış şifrelemelerin eski popülerliği azalmaktadır. Bunun sonucu olarak 2004 yılında daha güçlü akış şifre geliştirmek ve akış şifrelere eski popülaritesini tekrar kazandırmak amacıyla eSTREAM projesi başlatılmıştır. Bir kriptosistem, şifreleme algoritması, açık metin, şifreli metin ve anahtardan oluşmaktadır. Şifreleme algoritmaları kriptosistemin en önemli parçasıdır. Diğer yandan kriptografide blok şifreleme ve akış (stream) şifreleme olmak üzere iki temel simetrik algoritma tipi vardır. Bunlardan blok şifreleme, orijinal metni veya şifreli metni bloklara bölerek şifreleme/deşifreleme işlemini yapar. Akış şifrelemede ise bir bit veya byte üzerinde şifreleme ve deşifreleme işlemleri yapılır.

Bu tez çalışmasında birinci bölümde tez çalışmasıyla ilgili genel bir bilgi verilmiştir. İkinci bölümde ise şifreleme kavramlarının tanımları yer almaktadır. Günümüzdeki kullanım alanları belirtilmiş ve veri şifreleme algoritmalarından olan blok şifreleme ve akış şifreleme tanımlamalarından bahsedilmiştir. 3. bölümde rasgele sayıların tanımı ve sınıflandırılmaları üzerine çalışmalara yer verilmiş ve literatürde rasgele sayılar ile ilgili yapılan çalışmalar belirtilmiştir. Ayrıca rasgeleliğin tam olarak sağlanması yani önceki değerlere bakılarak bir sonraki değerin tahmin edilememesinde kullanılan çekirdek başlangıç değeri için günümüzde kullanılan üretme metotları ve literatürde kullanılan sözde rasgele sayı üreteçleri ve gerçek rasgele sayı üreteçleri açıklanmıştır. Bu çalışmanın 4. Bölümde ise söz konusu olan rasgeleliğin tam olarak gerçeklediğini gösteren istatistiksel testlerden National Institute of Standards and Technology *(*NIST), test süitine ve bu süit içerisine dahil edilmiş olan algoritmalara yer verilmiştir. 5. bölümde ise akış şifreleme algoritmalarının genel yapısı incelenerek rasgele sayı üreteci olarak kullanılması konusunda bazı değerlendirme çalışmaları yapılmıştır. Bu çalışmalar sırasında incelenen algoritmaların donanımsal gerçeklenmeleri FPGA ortamında yapılarak sonuçları incelenmiştir. Donanım çıkışları sonucu elde edilen rasgele sayı verileri alınarak testlere tabi tutulmuş ve rasgelelik değerleri ölçülmüştür. Bu değerler ise tablo

içerisinde belirtilmiştir. 6. bölümde yer alan sonuçlar kısmında ise yapılan çalışmalarla ilgili kısa bir değerlendirme verilmiştir.

1.1. Tezin Amacı ve Kapsamı

Bu tez simetrik şifreleme tekniklerinden akış şifreler ile bu şifreleme teknikleriyle üretilebilen rasgele sayı üreteçleri üzerinedir. Ayrıca kriptografik uygulamalarda kullanılan rassal sayıların ve bu sayılar kullanılarak geliştirilen anahtarların güvenilirliğini sağlamada önemli kriter oluşturan istatistiksel testler incelenmiştir. Rasgele sayı üreteçlerinden incelenen algoritmalar sonucu gerçekleştirilen donanımlardan elde edilen rasgele sayı dizileri bu testlere tabi tutularak değerlendirmelere yer verilmiştir. NIST test paketinde bulunan çeşitli test teknikleri incelenerek seçilen bazı akış şifrelere bu testler uygulanmış ve sonuçlar değerlendirilmiştir.

2. ŞİFRELEME KAVRAMLARI

2.1. Kriptografi

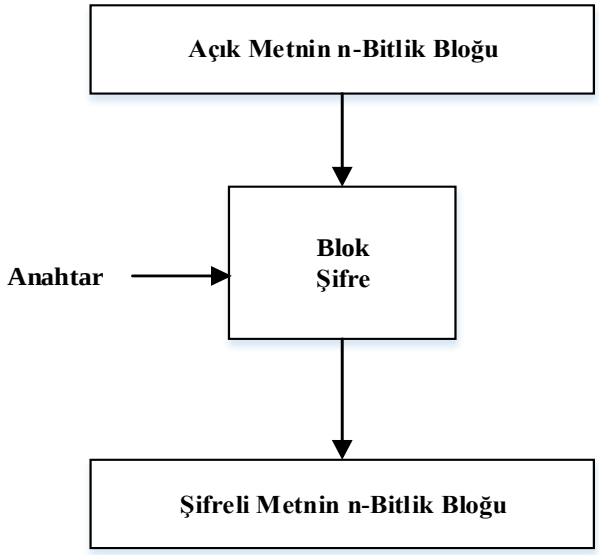
Kriptoloji bilgi güvenliğinin sağlanması ile uğraşmaktadır ve sayısal verinin korunmasında ya da güvenli bir şekilde iletilmesinde kullanılan şifreleme algoritmalarının tasarımı ve bu algoritmaların güvenlikleri ile ilişkilidir. Bu açıdan bakıldığında bilgi güvenliği, günümüzde sayısal verinin güvenli bir şekilde iletilmesinde çok önemli yer tuttuğu için giderek dikkat çekmektedir. Şifreleme algoritmaları sayısal verinin anlaşılmaz hale, bir anahtar yardımıyla, dönüştürülmesi işlemini yapmaktadır. Şifreleme işlemi sonucunda meydana gelen şifreli metin saldırgan tarafından anahtar bilinmeden deşifre edilememelidir.

Günümüzde kullanılan modern şifreleme algoritmaları üç ana kategoriye ayrılmaktadır. Bunlardan ilki simetrik şifreleme algoritmalarıdır. Blok şifreleme algoritmaları ve stream (akış) şifreler bu kategoriye girer. Bu tür algoritmalarda şifreleme ve deşifreleme işlemleri aynı anahtarı kullanır. Kullanılan anahtara gizli anahtar denir. İkinci ana kategori asimetrik şifreleme algoritmalarıdır ve şifreleme için gizli anahtarı kullanırken deşifreleme için açık anahtarı, yani herkesin erişebileceği anahtarı, kullanır. Son kategoriye ait şifreleme algoritmaları ise, hash algoritmalarıdır. Bunlar verinin sıkı bir temsilini oluşturmak için kullanılırlar ve kimlik denetiminin sağlanmasında büyük rol oynarlar.

2.1. 1. Blok Şifreler

Bugünün modern şifreleme teknikleri 0 ve 1' ler üzerinde yani ikili kodda şifreleme yapan algoritmalardan oluşmaktadır. Blok şifreler, sabit uzunluktaki blokları bir anahtar yardımıyla şifreleme işlemine tabi tutarlar. Yani blok şifreler n bit uzunluğundaki açık metni k bit anahtar yardımıyla n bit şifreli metne dönüştürürler. Şekil 2.1 bir blok şifrenin genel yapısını göstermektedir.

Simetrik şifreleme tekniklerinden blok şifreler ürün şifrelerdir ve köklerini Shannon'un ortaya koyduğu karıştırma ve yayılma tekniklerinden almıştır.[20] Karıştırma şifreli metin ve açık metin arasındaki ilişkiyi gizlemeyi amaçlarken, yayılma açık metindeki izlerin şifreli metinde sezilmemesini sağlamak için kullanılır. Karıştırma ve yayılma, sırasıyla yer değiştirme ve doğrusal dönüşüm işlemleri ile gerçeklenir. Blok şifrelerde yer değiştirme S kutuları ile sağlanırken yayılma byte veya bit bazında gerçekleştirilen doğrusal dönüşümler vasıtasıyla sağlanmaktadır. S kutuları ise bunun yanında doğrusal olmayan yapılardır ve doğrusal olmama (nonlinearity) bu tip şifrelerin tasarımındaki en önemli ölçütlerden biridir.



Şekil.2.1 Bir Blok Şifrenin Genel Yapısı

Blok şifrelerin tasarımında Feistel ağları ve yer değiştirme-Permütasyon ağları (SPN-Substitution-Permutation Networks) olmak üzere iki ana blok şifreleme mimarisi vardır.[21] Her iki mimari ürün şifrelerinin örneklerindendir. Yani birden fazla şifreleme işleminin birleşmesi ile oluşturulurlar. Tekrarlanan şifreler yine ürün şifreleridir ve aynı şifreleme adımının tekrarlanan uygulamasını içerirler. Her şifreleme adımına da döngü denir. Bir döngü birden fazla şifreleme adımı içerebilir. Genellikle her döngüde farklı anahtar materyali kullanılır.

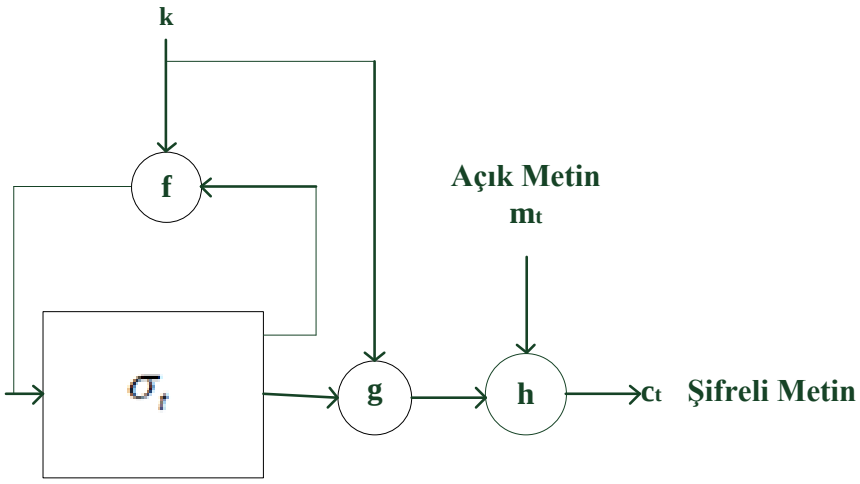
2.1.2. Akış Şifreleri

Kriptolojide açık veriyi rastgele bir şifreleme verisiyle, genelde dar veya işlemiyle, karıştıran simetrik anahtar şifreleyicisine akan veri şifreleyicisi denir. Buna karşın blok şifreleyiciler büyük bloklar üzerinde sabit ve değişmeyen dönüşümler yapar. Akan veri şifreleyicisi, blok şifreleyicisinden daha hızlıdır ve daha düşük donanıma ihtiyaç duyar. Öte yandan yanlış kullanıldığında, özellikle aynı başlangıç durumu 2. kez kullanıldığında, büyük güvenlik açıkları verebilir.

Vigenere şifresinin mesaj uzunluğu anahtar uzunluğuna eşit formuna tek kullanımlık şerit (one time pad) denmektedir.[22] Mesaj bitleri M = m_1, m_2 ,... m_s ve anahtar bitleri K = k_1, k_2 ,..... k_s olmak üzere şifreli metin, (2.1) ifadesinde gösterildiği gibi anahtar bitleri ve açık metin bitlerinin (mod 2)' de toplamı ya da XOR işlemi sonucu elde edilir.

$$c_i = m_i \oplus k_i \ , \quad i = 1.....s \tag{2.1}$$

Tek kullanımlık şeritlerde anahtar tamamen rastlantısal olmalı ve bir kereliğine kullanılmalıdır. 1949 da Shannon bu şifreleme sisteminin koşulsuz güvenli olduğunu göstermiştir. Bu şifrelerdeki en önemli kısıtlama anahtar uzunluğunun mesaj uzunluğuna eşit olması gerekliliğidir. İşte, akış şifreler *k* bit anahtarla bir üreteci besleyerek mümkün olduğu kadar uzun periyotlu ve rastlantısal gözüken anahtar dizilerini üretmeyi amaç edinir ve elde ettiği anahtarı açık metinle şifreleme fonksiyonuna sokarak şifreli metni elde eder.[23][24] Akış şifreler zamanla değişen bir fonksiyon kullanarak tek karakterler üzerinde işlem yaparlar. Şekil 2.3 de senkron bir akış şifresinin genel yapısı görülmektedir. Şifreli metin (c_t), açık metin (m_t) ve anahtar dizisi (s_t)'nin h fonksiyonuna girişinin sonucunda elde edilir.



Şekil.2.2. Senkron Bir Akış Şifresinin Genel Yapısı

3. RASGELE SAYILAR VE RASGELE SAYI ÜRETEÇLERİ

Rassallık, elemanları arasında kolay ilişki bulunmayan, belirli bir taslağı olmayan kısaca tahmin edilemeyen bir özellik olarak karşımıza çıkar. Rassal olarak üretilen sayılar, şans oyunlarında, istatistiksel örneklemelerde ve simülasyon uygulamalarında sıkça kullanılır. Rassallık kriptografide gizliliği, çözülemezliği sağlayabilmek amacıyla kullanılan en temel özelliklerden biridir. Saldırganın gerçek verileri elde edememesi için şifreleme sonucunun olabildiğince tahmin edilemez olması gerekir. Rassal sayılar birçok kriptografik uygulamanın temelini oluşturur. Kriptografik uygulamalarda kullanılmak üzere rassal sayı üreteçleri bulunmaktadır. Rastgele sayı üreteçleri, çıkışındaki sayılar istatistiksel olarak birbirinden bağımsız olan sistemlerdir. Rastgele sayı üreteçleri bilgisayar benzetimleri, sayısal analiz uygulamaları, istatistiksel analiz, Monte Carlo metodunun kullanıldığı uygulamalar ve özellikle şifreleme gibi alanlarda sıkça kullanılmaktadır. Rasgele sayı üreteçleri ile giriş ve çıkışlardaki sayıların istatiksel olarak birbirinden bağımsız olması sağlanır. Rasgelelik, gizliliği yani çözülmezliği sağlayan en önemli unsurların başında gelir. Örneğin, şifreleme algoritmalarının güvenilirliği, rastgele sayı üreteçlerinin ürettiği sayılara bağlıdır. Bu sayılar istatistiksel olarak rastgele ise, yani önceki çıkışlara bakarak daha sonrakiler tahmin edilemezse üretecin istatistiksel özelliği iyi demektir. Başka bir deyişle iyi bir şifreleme iyi bir RSÜ gerektirir.

Rasgele bir sayı her basamağının aynı olma olasılığına sahip olduğu ve ardışık basamakların birbirinden tamamen bağımsız olduğu bir basamaklar serisi olarak da tanımlanabilir. Bir kümenin veya dizinin elemanlarından bir kısmının, istatistiksel olarak rasgele seçilmesi yoluyla üretilmiş olan sayılar bilgisayar biliminin birçok alanında ihtiyaç duyulmaktadır.

Bilgisayar bilimlerinin birçok alanında ihtiyaç duyulan rasgele sayılar; şifreleme algoritmalarında da önemli bir role sahiptir. Şifreleme işleminin gizliliği ve güvenilirliği açısından rasgele sayılar çok önemlidir. Anahtarlar, kriptografik protokoller hiçbir düşmanın tahmin edemeyeceği rastgele bit kaynaklarına ihtiyaç duyarlar. Benzer şekilde bir oyun programlanırken veya bir simülasyon sırasında rastgele meydana gelen olaylar modellenirken rastgele sayılara ihtiyaç duyulur. Bu sayıların da gerçekten rastgele olması beklenir. Bundan dolayı üreteçlerin her zaman için aynı rastgele sayıyı üretmesi istenmeyen bir durumdur. Çünkü bu durumda sisteme bir kere saldırarak anahtarı ele geçiren saldırganın bundan sonraki saldırılarda anahtarı bulmak için vakit kaybetmesine gerek kalmaz. Benzer şekilde farklı sayılar üreten ama sayıların tahmin edilmesi mümkün olduğu durumlarda da sistemin karmaşıklığı azaltılarak sistemin getirdiği karmaşıklık sonucu beklenenden oldukça kısa sürelerde sisteme saldırı gerçekleşebilir.

Bilgisayarlar tasarımları ve yapıları itibariyle rastgeleliğe yer bırakmayan belirli ve her adımında olacak şeylerin önceden tasarlandığı makinelerdir. Bu anlamda bilgisayarlarda rastgele bir bilgi üretmek oldukça zordur ve istenilen rasgele sayı dizilerini elde etmek için belirli koşullara uymak gerekir. Gerekli koşullar aşağıdaki gibi özetlenebilir;

- Rastgele sayı üretecinin tekrarlanma periyodunun uzun olması gerekir. Bir üretecin n tane sayı üretimi için kendini yineleme periyodunun çok uzun olması gerekir. Rastgele sayı üreteçleri bir matematiksel fonksiyona dayanarak sayı üretimini sağlarlar. Bu nedenle belirli bir periyodda fonksiyonun kendisini yinelemesi söz konusudur. Böylece oluşturulan algoritmanın yineleme periyodunun çok uzun olması gerekir.
- Rastgele sayı üretecinin n tane sayı üretiminde elde edilen dizinin sayılarının ardışık olarak birbirinden bağımsız olmaları gerekir
- Rastgele sayı üretecinde t zamanda elde edilen n tane sayının elemanları t_i , t_{i+k} periyodlarında bir kümeleme göstermemelidir.
- Rastgele sayı üretecinde sayı üretimleri tekrarlanabilir, yeniden elde edilebilir olmalıdır. Oluşturulan bir algoritmanın her *t* periyod için çalıştırılması halinde üretilen diziler birbirine eşit olmalıdır.
- Rastgele sayı üreteci çalıştırıldığı bilgisayar türüne bağımlılık göstermemelidir. Genellik prensibine uygun üretimler yapılabilmelidir.
- Rastgele sayı üreteçleri sistemdeki her hangi bir *X* değişkeninin asimtotik dağılışına kolayca uyabilen bir esneklik içinde olmalıdır. Türetilen diziler kolayca amaca uygun biçime geçebilmelidir.
- Rastgele sayı üreteçleri ile üretilen sayı dizilerinde sayıların önceki ve sonraki değerlerine bağımlılığı olmamalıdır.
- Üretim, $n \rightarrow \infty$ için istenilen büyüklükte kısa sürede elde edilebilir rastgele sayı üretim algoritmaları olmalıdır.

3.1. Rasgele Sayı Üreteçlerinin Sınıflandırılması

Doğa ve mühendislik sistemleri kesin olarak tahmin edilebilir bir tarzda değildirler. Sistemler genelde gürültü içerir bu yüzden bir sistemi gerçekçi modellemek için rastgeleliğin bir derecesi modele eklenmelidir. Böylelikle gerçeğe yakın sistemler tasarlanabilir. Bilgisayar ve derleyici sistemler üzerinde de aynı şekilde gerçek ve doğru bir sisteme tasarlayabilmek için

rasgele sayılara gereksinim duyulmaktadır. Bu sayıları elde edebilmek için ise rastgele sayı üreteçlerine ihtiyaç duyulmaktadır.

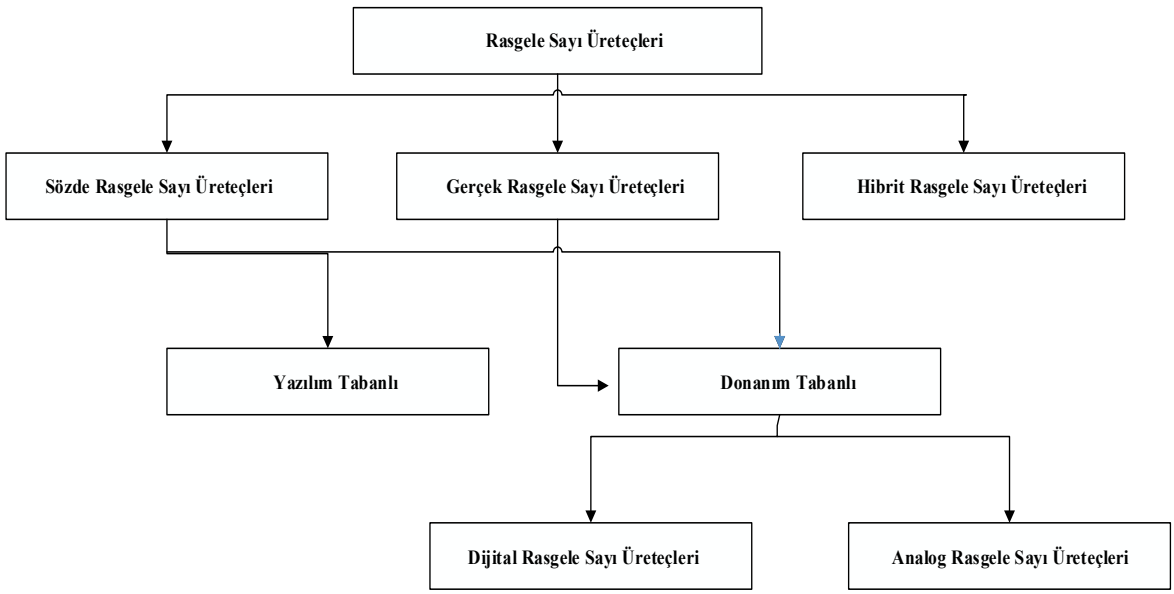
Rasgele sayıların elde edilebilmesi amacıyla çeşitli rasgele sayı üreteçleri geliştirilmiştir. Bu rasgele sayı üreteçleri

- Gerçek Rasgele Sayı Üreteçleri (GRSÜ)
- Sözde Rasgele Sayı Üreteçleri (SRSÜ)
- Hibrit Rasgele Sayı Üreteçleri

olmak üzere3 farklı şekilde sınıflandırılmaktadır.

Gerçek Rasgele Sayı Üreteçleri birçok uygulama için pahalı ve yavaş iken daha basit yapıya sahip Sözde Rasgele Sayı Üreteçleri birçok uygulama için yeterli ve etkilidir. Kriptografik sistemlerde rasgele sayı üreteçleri kullanıldığında, rasgele sayı üretecindeki zayıflık veya eksiklik sistemin başarısız olmasına yol açar. Bu yüzden etkili rasgele sayı üretme ihtiyacı şimdilerde gittikçe artmaktadır.

Rasgele sayıların bilgisayar biliminde kullanılması için Şekil.3.1'deki gibi sınıflandırılmaktadır.



Şekil.3.1. Rasgele Sayı Üreteçlerinin Sınıflandırılması

SRSÜ'leri Şekil.3.1'e göre bir alt sınıflandırma olan yazılım ve donanım tabanlı rasgele sayı üreteçleri olarak gerçekleştirilebilmesine rağmen GRSÜ'leri sadece Donanım tabanlı olarak üretilebilmektedir. Donanım tabanlı rasgele sayı üretiminde rasgelelik kaynağı olarak fiziksel olaylar kullanılmaktadır. Bu rasgelelik kaynakları aşağıdaki gibi özetlenebilir.

- Radyoaktif bozulma sırasında parçacıkların emisyon arasında geçen süre
- Bir direnç ya da diyot elemanından elde edilecek termal gürültü
- Birbirinden bağımsız olarak çalışan ossilatörler arasındaki frekans istikrarsızlığı
- Yarı iletken kondansatörün belirli bir süre boyunca şarj zamanı
- Bir hard disk içerisindeki hava türbülansı
- Bir mikrofon elde edilen ses veya kameradan elde edilen görüntü

Yazılım tabanlı rasgele sayı üreteçlerinde ise rasgelelik kaynakları deterministik olup aşağıdaki gibi özetlenebilir.

- Sistem saat frekansı
- Fare hareketleri
- Giriş/Çıkış tampon bellek içerikleri
- Kullanıcı giriş değerleri
- İşletim sistemi ile ilgili olaylar, sistem yükü veya ağ kullanım istatistiği

3.1.1. Sözde Rasgele Sayı Üreteçleri

SRSÜ'ler, sonlu durum makinalarıyla gerçeklenen deterministik bir algoritmayla, sayı üreten sistemlerdir. GRSÜ'lerle karşılaştırıldığında, kolay gerçeklenme ve düşük maliyetle üretilme gibi avantajları vardır. Fakat kullanılan algoritmalar determinstiktir ve bu nedenle çıkışlar istenen şekilde tam rastgele değildir. Girişteki algoritma bilindiğinde, herhangi bir andaki değerine bakarak sonraki çıkışlar tahmin edilebilmektedir. Bu da gizlilik isteyen şifreleme algoritmalarında kullanımını kısıtlar. Kısaca GRSÜ'lere kıyasla istatistiksel olarak başarımı düşük RSÜ'lerdir. Bunun yanı sıra sayısal analiz ya da fiziksel süreç modelleme gibi daha düşük istatistiksel kalitede rastgeleliğin yeterli olduğu durumlarda tercih edilmektedirler.Bu algoritmalar kendi içlerinde herhangi bir rasgelelik barındırmazlar, algoritmalar da genelde açıktır. Buradaki rasgelelik algoritmaların girdileri ile sağlanır. Bu yüzden algoritmaların girdileri gizli tutulmalıdır ve kolay tahmin edilemez olmalıdır. Algoritma ve girdi bilinirse, dizinin tümü elde edilebilir. Bu üreteçler verimlidir ve uzun diziler üretmenin maliyeti düşüktür. Kriptografik olarak kullanılabilecek sözde-rasgele (pseudo-random) sayı üreteçleri ile üretilen bir dizinin bir kısmı biliniyorsa, bu dizinin diğer kısımları ile ilgili bir bilgi vermemelidir. Aynı üreteçle üretilen farklı diziler birbirleri ile ilişkileri olmamalıdır. Dizilerin periyotları mümkün olduğunca uzun olmalıdır.

Uygulamada, pek çok sözde-rasgele sayı üreteci istatistiksel olarak önemli testleri geçmelerini engelleyen bazı durumlar sergiler. Bunlar;

- Bazı başlangıç durumları için beklenenden daha kısa periyodlar
- Kötü boyutsal dağılım
- Birbirini takip eden değerlerin bağımsız olmaması
- Bazı bitlerin diğerlerinden 'daha rasgele' olabilmesi
- Tek biçimlilik eksikliği Hatalı sözde rasgele sayı üreteçlerinin problemleri kolay kolay tespit edilemeyecek türde olabileceği gibi saçma denecek kadar açık da olabilir.
- Şifre bilimsel olarak uygun olan bir sözde rasgele sayı üreteci rasgelelik testlerini geçmeye ek olarak bazı ek şifre bilimsel koşulları da sağlamak zorundadır. Bazı şifre bilimsel olarak güvenli sözde rasgele sayı üretici algoritmalar şunlardır:
- Counter modda veya çıktı besleme modunda çalışan akış veya blok şifreleri.
- Güvenlik kanıtı olan özel tasarımlar. Örneğin; Blum Blum Shub algoritmasının güçlü bir koşullu güvenlik kanıtı vardır ancak yavaş çalışmaktadır.

Bu sayı üreteçlerinin verdiği çıkış değerlerinin ne kadar rasgele olduğu, yani giriş değerlerinden elde edilen çıkış değerleri arasındaki bağımsızlığın yani karmaşanın değerlendirmesi işleminde, yapılan matematiksel işlemlerle kesin bir sonuca varılamadığından, bu üreteçler için hazırlanmış istatiksel testler uygulanarak sonuç hakkında bir yorum getirilebilmiştir.

Sözde rasgele sayı üreteçleri herhangi bir başlangıç (tohum) değeri olmadan başlayamaz. Tohum değeri rasgele seçilmiş olmalıdır. Belirlenen tohum değeri belirli bir algoritmaya tabi tutularak uzun rasgele sayı dizileri üretilmiştir. SRSÜ'nün avantajlı yanı diğer uygulamalara oranla ucuz olması, kolay gerçeklenebilir olması, hızlı olması ve donanım ihtiyacına gerek duymamasıdır. Ancak SRSÜ'ler ile üretilen sayılar tohum değeri tespit edildiğinde veya sistemde kullanılan fonksiyonlar yeterince karmaşık olmadığı taktirde tahmin edilebilmiştir. Ayrıca belli bir süre sonra üretilen dizi kendini tekrar etmeye (periyodiklik) başlamıştır. Belirtilen bu eksiklikler nedeniyle SRSÜ'ler kriptografik uygulamalar için uygun değildir.

3.1.2. Gerçek Rasgele Sayı Üreteçleri

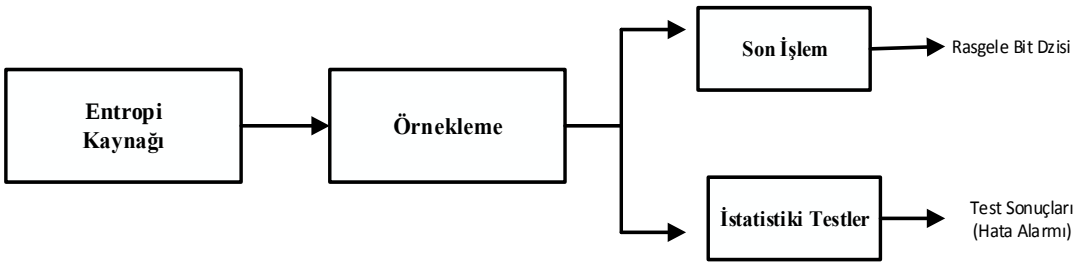
GRSÜ'ler, girişleri deterministik olmayan, doğal süreçlerin rasgeleliğini kullanan bir algoritmayla sayı üreten sistemlerdir. Çıkıştaki sayılar rastgele olduğundan şifrelemede çokça kullanılırlar. İçinde rassal bir yapı bulunduran fiziksel sinyal kaynakları kullanarak dizi üreten üreteçlerdir. Bu üreteçlerin en önemli avantajları:

- Dizinin bir kısmına sahipken, farklı bir kısmını elde etmenin mümkün olmaması;
- Üretilen diziler kendi içinde herhangi bir gizli bağıntının bulunmaması;
- Periyodik olmamalarıdır.

Bu avantajların yanı sıra, gerçek rassal sayı üreteçlerinin önemli dezavantajları da bulunur. Bu üreteçler çoğunlukla verimsizdir, uzun sayı dizileri elde etmenin maliyeti yüksektir. Deneyi tekrarlayıp bir sayı dizisini yeniden elde etmek mümkün değildir.

Donanım ve yazılım tabanlı olmak üzere iki farklı teknikle gerçeklenebilirler. Donanım tabanlı üreteçler, yarı iletken bir diyodun ya da bir direncin ısıl gürültüsü, bir ossilatörün faz gürültüsü, radyoaktif bir bozulma esnasında parçacıkların yayılmaları arasında geçen süre gibi fiziksel olayların rasgeleliğini kullanır. Bu süreçler sonucu oluşan işaretler de kendi aralarında ilintili olabileceğinden dolayı, tam rasgeleliği sağlamak için çıkış yeniden basit bir algoritmaya tabi tutulabilir. Yazılım tabanlı üreteçler, sistemin saati, mouse hareketleri arasındaki süre, sabit diske erişim gibi bilgisayar tabanlı olayları temel alır. Yazılım tabanlı üreteçleri gerçeklemek, donanım tabanlıları gerçeklemekten daha zahmetli ve daha az güvenilirdir. Örneğin, Netscape'in rastgele sayı üretecinin temel aldığı verinin, günün saati ve süreç numarası olduğu tespit edilebilmiştir. Bu nedenle yazılım tabanlı üreteçler ikinci planda kalmıştır.

Gerçek rasgele sayı üreteçleri gürültü kaynağı olarak kontrol edilemeyen ve kestirilemeyen gerçek fiziksel süreçleri kullanarak rasgele sayılar üretmektedir. GRSÜ tarafından üretilen sayıların özellikleri ve rasgeleliği fiziksel süreçlerin rasgeleliğine bağlıdır. Kontrol edilemeyen fiziksel süreçler olduğu taktirde üretilen sayılarda kestirilemez ve kontrol edilemez. Ancak bazı üretilen bit dizileri istatistiki zayıflıklar göstermiştir. Bu zayıflıkların giderilmesi amacıyla üretilen bit dizisi son işleme tabi tutulmuştur. Şekil.3.2. de bu durum gösterilmiştir.



Şekil.3.2. Gerçek Rasgele Sayı Üreteçlerinin Genel Yapısı

GRSÜ'ler kriptografik uygulamalar için zorunlu olan kestirilememe, tekrar üretilememe ve iyi istatistiki özellikleri sağlaması sebebiyle kriptolojide birçok uygulamada kullanılmıştır.

4. RASGELE SAYI ÜRETEÇLERİ İÇİN İSTATİSTİKSEL TESTLER

Bu testler, üretecin çıkışının gerçek bir rastgele diziden beklenenleri karşılayıp karşılamadığını söyler. Ayrıca testlerin sonuçlarına bakılarak rastgele sayı üretecinin kalitesi hakkında yorum yapılabilir. Bir sayı dizisinin rastgele olduğunu söylemek için, tüm testlerden geçmesi gerekir. Sadece bir tane test başarısız olsa bile dizi rastgele kabul edilemez.

Üretilen rasgele sayıların uygun koşullarda üretildiğini kontrol etmek ve rasgeleliğe uygunluğunu ölçmek için bir istatistiksel test yeterli değildir. Bu konuda birçok test paketi üretilmiştir (FIBS 140, DieHard, NIST). Özellikle rasgele sayıların kriptografik uygulamalarda kullanılabilmesi için bu rasgelellikleri belirleyen testlerden geçmesi gerekmektedir. Bu tez çalışmasında hipotez test tabanlı NIST istatistiksel test süiti açıklanacaktır. Bu hipotez testler üretilen 0 veya 1 sayısının rasgele olup olmadığını belirler. Bu amaç için NIST test süitinde iki önemli parametre vardır bunlar sırasıyla α ve P-Değeri değeridir. Önem seviyesi olarak bilinen α 0.01 olarak seçilmesi test edilecek sayıların rasgeleliğinin 99% güven değerine sahip olduğunu belirtir. Diğer parametre P-degeri rasgeleliğin ölçüsü olarak bilinir. Eğer P-değeri 1'e eşit olursa sayılar mükemmel rasgeleliğe sahiptir denir. P-değeri sıfıra eşit olursa sayıların rasgeliğinden söz edilemez. Kriptografik uygulamalarda kullanılmak üzere üretilen sayıların önem seviyesi α, uygun bir değer seçilmelidir. Her bir test için eğer P-değeri, α değerinden büyük ve eşit olursa test başarılıdır. Aksi durumda test başarısız yani üretilen sayılar rasgele değildir. Tipik olarak önem seviyesi [0.001, 0.01] aralığında seçilir.

4.1. FIPS 140-1 Testleri

FIPS 140-1 testi dört tane ayrı testten oluşur. RSÜ'nün çıkışından alınan ve 20000 tane bit içeren bir bit dizisi dört teste birden tabi tutulur ve dizinin rassal olabilmesi için tüm testlerden geçmesi gerekir. Bu dört test aşağıda açıklanmıştır.

4.1.1. Monobit Testi

Bu testin amacı, bit dizisindeki 0 ve 1 sayısının rassal bir diziden beklendiği gibi olup olmadığını tespit etmektir. Testin başarılı olabilmesi için 20000 bitlik bir dizideki '1' sayısının $9654 < n < 10346$ aralığında olması gerekir.

4.1.2. Poker Testi

Bu testte, 'k' bit içeren dizi, $m\,k > 5.2^m$ olacak şekilde, üst üste çakışmayan m bitlik parçalara ayrılır ve $i.$ parça n_i diye adlandırılır. Rassal bir diziden beklenen, tüm m bitlik blokların k uzunluklu bir dizide aynı sayıda birbirini tekrar etmesidir. Testin başarılı olabilmesi için, $X = \frac{2^m}{k}\left[\sum_{i=0}^{2^m} n_i^2\right] - k$ formülüyle hesaplanan X değerinin, $k=20000$ ve $m=4$ için, $1.03 < X < 57.4$ aralığında olması gerekir.

4.1.3. Blok Testi

Elimizdeki bit dizisinin bu testten başarıyla geçmesi için, dizide ardarda gelen '1' ve '0'lardan oluşan çeşitli uzunluktaki blokların (runs'ların) sayısının tabloda belirtildiği gibi olması beklenir. 6 bitten daha uzun bloklar 6 bitlik olarak kabul edilmektedir

Tablo 4.1. Run Testi Koşulları

Blok Uzunluğu	Blok Sayısı Aralığı
1	2267-2733
2	1079-1421
3	502-748
4	223-402
5	90-223
6+	90-223

4.2. NIST Testleri

NIST testi, FIPS 140-1 testine göre çok daha güçlü bir testtir. FIPS 140-1 testini geçen bir bit dizisi NIST 800-22 testinden kalabilir. Bu nedenle ciddi uygulamalarda NIST 800-22 tercih edilmektedir. Sistem uzun bloklardan oluşan verileri test etmek amacıyla kullanılır. Daha önceki testlere oranla daha güçlü yapı içerir. Yani daha önceki testlerden geçmiş ve güvenilir sayılan bir sistem bu testten geçemeyebilir. Bu sebeple bu sistem genelde ciddi işlemlerde uygulanabilecek bir yapıdır. NIST 800-22 kendi içinde 15 tane ayrı testten oluşur. Teste tabi tutulan bit dizisinin başarılı olabilmesi için tüm testleri başarıyla geçmesi gerekmektedir. Aşağıda bu testlerin hepsi kısa açıklamasıyla birlikte verilmiştir.

- **Frekans (Frequency) Testi:** Bit dizisindeki 1 ve 0 dengesini inceler.
- **Blok Frekans (Block Frequency) Testi:** *m* bitlik bit bloklarının 0 ve 1 dengesini inceler.
- **Akış (Runs) Testi:** Dizideki 0 ve 1 bloklarının (runs) sayısını inceler.
- **Bloktaki En Uzun Birler (Longest run of Ones in a Block) Testi:** Dizideki 0 ve 1 bloklarının (runs) uzunluklarını inceler.
- **Rank Testi:** Sabit uzunluklu bit blokları kullanılarak, her biri bir satırı belirtecek şekilde, bir matris oluşturulur ve matrisin rankı hesaplanarak bloklar arasındaki lineer bağımlılık incelenir.
- **Ayrık Fourier Dönüşümü (Discrete Fourier Transform) Testi:** Mevcut bit dizisinin ayrık Fourier dönüşümünü alır ve periyodikliği inceler.
- **Örtüşmeyen Şablon Eşleşme (Non-Overlapping Template Matching) Testi:** *m* bitlik bir bloğun dizi içinde tekrarını inceler. Tekrar edilmesi halinde, tekrar edilen bloktan itibaren yeni bir *m* bitlik blok oluşturulur.
- **Örtüşen Şablon Eşleşme (Overlapping Template Matching) Testi:** *m* bitlik bir bloğun dizi içinde tekrarını inceler. Tekrar edilmesi halinde, blok 1 bit ötelenerek yenisi oluşturulur.
- **Maurer's Evrensel (Universal) Testi:** Dizinin veri kaybı olmadan ne kadar sıkıştırılabileceğini inceler.
- **Doğrusal Karmaşıklık (Linear Complexity) Testi:** Bit dizisinin LFRS (linear feedback shift register) uzunluğuna bakarak kompleksliğini inceler.
- **Seri (Serial) Testi:** Tekrar eden m bitlik *2m* tane bloğun tekrar sayısının dağılımını inceler. *m=1* için, birinci teste denktir.
- **Yaklaşık Entropi (Approximate Entropy) Testi:** Tekrar eden *m* ve *(m+1)* bitlik blokların entropisini inceler.
- **Birikimli Toplamlar (Cumulative Sums) Testi:** Bit dizisini ardışık uzunluklu bloklara ayırıp blokların 1 ve 0 dengesini belirler ve bloklar arasındaki dengesizlik farkına bakar.
- **Rasgele Gezinim (Random Excursion) Testi:** Bit dizisini ardışık uzunluklu bloklara ayırıp blokların 1 ve 0 dengesini belirler ve daha sonra blokların dengesinin dağılımını inceler.

- **Rasgele Gezinim (Random Excursion) Değişken Testi:** Bit dizisini ardışık uzunluklu bloklara ayırıp blokların 1 ve 0 dengesini belirleyip ortalama değerden sapma miktarını belirler.

4.2.1. Frekans Testi

Verilen bir dizide bulunan 0 ve 1'lerin oranını kontrol eder. Testin herhangi bir parametresi yoktur. Testte kullanılan referans dağılım yarım normal dağılımdır. Testin sonunda elde edilen p-değerinin çok küçük çıkması dizideki 1' lerin ya da 0' ların sayısının beklenenden fazla olduğunu gösterir. Testin geçerli olabilmesi için dizi uzunluğunun en az 100 bit olması gerekir. Test denklemleri kullanılarak üretilen değer olan $p > 0.01$ ise dizi rastgele olarak kabul edilir.

n: Bit dizisinin boyutu

ε: RNG veya PRNG ile üretilen bit dizisi

S_n**:** Bit dizisinin toplam değeri (Bit dizisindeki 0'lar (-1), 1'ler ise kendi değeri kabul edilerek toplama işlemi gerçekleştirilerek elde edilen sonuçtur).

$$S_{obs} = \frac{|S_n|}{\sqrt{n}} \tag{4.1}$$

erfc: Hata fonksiyonu $\quad \text{erfc}(z) = \frac{2}{\sqrt{n}} \int_2^{\infty} e^{u^2} du$ (4.2)

P-Değeri=$\text{erfc}(\frac{S_{obs}}{\sqrt{2}})$ (4.3)

Örnek:

ε=10110110101101100010101010101011100111000110110001010001100100110101010011010100101011001100111001 10

n = 100 $\quad S_{100} = 2 \quad S_{obs}=2.0$

P-değeri = 0.843053 >0.01 olduğundan dizi rastgele kabul edilir.

4.2.2. Blok Frekans Testi

Bu test bloğunda verilen bir dizide bulunan 0 ve 1'lerin oranını *m* bitlik bloklar içinde kontrol eder. Testin tek parametresi elde edilen blok uzunluğudur. Blok uzunluğu 1 bit olursa bu test

frekans testine dönüşür. Her bir bloktaki 1'lerin beklenen oranı *m/2* dir. Kullanılan referans dağılımı ki-kare dağılımıdır. Testin iyi sonuç vermesi için dizi uzunluğu en az 100 bit blok uzunluğu da 20 bit olmalıdır.

m: blok uzunluğu

n:dizi uzunluğu

ε: RSÜ veya SRSÜ ile üretilen bit dizisi

X^2(obs) : Beklenen oran (1/2) ile karşılaştırılan verilmiş m bit blok içerisindeki 1'lerin gözlemlenen oranının nasıl olduğunun ölçüsüdür.

- $\mathrm{N} = \frac{\mathrm{n}}{\mathrm{m}}$ örtüşmeyen bloklar halinde girilen dizi bölünür. Kullanılmayan bitler atılır.

  n=10 m=3 ε=0110011010 $\mathrm{N} = \frac{10}{3} = 3$

  011, 001 ve 101 bloklarından oluşur.

- $\pi_i = \frac{\sum_{j=1}^{m} \varepsilon_{(i-1)m+1}}{\mathrm{m}}$ denklemi kullanılarak her bir *m* bit bloktaki 1'lerin oranı π_i'ye karar verilir.

  $1 \leq \mathrm{i} \leq \mathrm{N}$ $\pi_1 = \frac{2}{3}, \pi_2 = \frac{1}{3}, \pi_3 = \frac{2}{3}$

- X^2 istatistiği hesaplanır. $X^2(\mathrm{obs}) = 4\ \mathrm{m}\ \sum_{\mathrm{i}=1}^{\mathrm{N}} \left(\pi_i - \frac{1}{2}\right)^2$

- $X^2(\mathrm{obs}) = 4 * 3 * \left(\left(\frac{2}{3} - \frac{1}{2}\right)^2 + \left(\frac{1}{3} - \frac{1}{2}\right)^2 + \left(\frac{2}{3} - \frac{1}{2}\right)^2\right) = 1$

P-Değeri= $\mathrm{igamc}\left(\frac{\mathrm{N}}{2}, \frac{X^2(\mathrm{obs})}{2}\right)$ (4.4)

- **P-Değeri**= $\mathrm{igamc}\left(\frac{3}{2}, \frac{1}{2}\right) = 0.801252 \geq 0.01$ ise dizi rastgeledir.

Örnek:

ε=10110110101101100010101010101011100111000110110001010001100100110101010011010100101011001100111001 10

π_1=$\frac{3}{5}$ π_2=$\frac{1}{2}$ π_3=$\frac{1}{2}$ π_4=$\frac{3}{5}$ π_5=$\frac{1}{2}$

π_6=$\frac{2}{5}$ π_7=$\frac{1}{2}$ π_8=$\frac{2}{5}$ π_9=$\frac{3}{5}$ π_{10}=$\frac{1}{2}$

$\chi^{2(obs)}$=2

P-Değeri=igamc($\frac{N}{2}, \frac{x^2}{2}$)

P-Değeri=igamc(5,1)=0,99634015

P-Değeri ≥0.01 olduğundan dizi rastgeledir.

4.2.3. Akış Testi

Akış testi bit dizisindeki akışların toplam sayısıyla ilgili olan testtir. Akış ifadesi dizideki ardışık aynı bit sıralamasını ifade eder. Böylece 0'lar ve 1'ler arasındaki dalgalanmaların kontrolü sağlayıp üretilen bit dizisinin yavaş ya da hızlı olacağını söyler. Bu test için referans dağılımı X^2 dağılımıdır. Bu testin işleyişi aşağıdaki gibidir;

n: bit dizisinin uzunluğu

ε: RSÜ veya SRSÜ ile üretilen bit dizisi

$\mathbf{V_n(obs)}$: Tüm n bitlerin arasında toplam tekrar sayısı (yani, sıfırların toplam tekrarı + birlerin toplam tekrarı(Akışların toplam sayısı))

- $\pi = \frac{\sum_j \varepsilon_j}{n}$ dizideki 1'lerin sayısını hesaplamak için kullanılır.

  Örneğin: ε=0001101001 n=10 ve $\pi = \frac{4}{10} = \frac{2}{5}$

- Eğer ön şart olarak frekans testi geçildi ise karar verilme aşamasına gidilir $\left|\pi - \frac{1}{2}\right| \geq \tau$ gösterilir. Eğer bu durum gerçekleşmezse akış testi uygulanmayabilir. Test uygulanmaz ise p değeri 0.000 olur. Bu test için, $\tau = \frac{2}{\sqrt{n}}$ test kodu önceden tanımlanmıştır. Bu bölümdeki örnek için: $\tau = \frac{2}{\sqrt{10}} = 0.63246$ olduğundan dolayı, $\left|\pi - \frac{1}{2}\right| = \left|\frac{3}{5} - \frac{1}{2}\right| = 0.1 < \tau$ olur ve test çalışmaz.

- $V_n(obs) = \sum_{k=1}^{n-1} r(k) + 1$, burada $\varepsilon_k = \varepsilon_{k+1}$ ise r(k)=0 değilse r(k)=1 olur.

- Örneğin:ε=100110101$V_{10}(obs)(1+0+1+0+1+1+1+1+0)+1=7$

P-Değeri= $erfc\left(\frac{|V_n(obs)-2n\pi(1-\pi)|}{2\sqrt{2n}\pi(1-\pi)}\right)$ (4.5)

= $erfc\left(\frac{\left|7-2\,10\,\frac{3}{5}(1-\frac{3}{5})\right|}{2\sqrt{210}\,\frac{3}{5}(1-\frac{3}{5})}\right) \geq 0.01$ ise dizi rasgeledir.

Bu testin uygulanabilmesi için dizi uzunluğunun en az 100 bit olması gereklidir. $V_n(obs)$'un büyük değerleri için dizide osilasyon hızlı, küçük değerleri için yavaştır(bir osilasyon 1'den 0'a veya tersine bir değişimdir). Değişim ne kadar çok gerçekleşiyor ise osilasyon o kadar hızlı gerçekleşiyordur.

4.2.4. Bloktaki En Uzun Birler Testi

Bloktaki en uzun birler testi, *m*-bitlik bloklarda bulunan en uzun birler grubu üzerinde odaklanarak rasgeleliğin testini gerçekleştirir. Testin tek parametresi blok uzunluğudur. Dizi *m*-bitlik *n* tane bloğa bölünür ve her blok içerisindeki en uzun birler öbeğinin uzunluğuna bakılır ve bu değerlerin frekansları beklenen değerlerle kıyaslanır. Ciddi bir sapma olup olmadığı kontrol edildikten sonra testte kullanılmak üzere referans dağılımı olan ki-kare test dağılımı uygulanır. Dizi uzunluğuna göre blok uzunluğu ve blok sayısına karar verilir.

n: bit dizisinin uzunluğu
ε: RSÜ veya SRSÜ ile üretilen bit dizisi
m:Her bir bloğun uzunluğu
N: Örtüşmeyen blokların sayısı

Tablo.4.2 Bloktaki En Uzun Birler Test Parametreleri

Minumum n	M
128	8
6272	128
750000	10^2

Dizi M bitlik bloklara bölünür ve kategori halinde her bir blok için en uzun birlerin akışının frekansı v_i hesaplanır. Bu değerler aşağıdaki tabloda gösterilmiştir. Tablolardaki her bir hücre belirli uzunluktaki birlerin akışının sayısını içerir.

Tablo.4.3. Belirli Uzunluktaki Birlerin Akış Sayıları

$\mathbf{v_i}$	M=8	M=128	M=10^4
$\mathbf{v_0}$	≤1	≤4	≤10
$\mathbf{v_1}$	2	5	11
$\mathbf{v_2}$	3	6	12
$\mathbf{v_3}$	≥4	7	13
$\mathbf{v_4}$		8	14
$\mathbf{v_5}$		≥9	15
$\mathbf{v_6}$			≥16

$$X^2(\text{obs}) = \sum_{i=0}^{k} \frac{(v_i - N\pi_i)^2}{N\pi_i}$$

K ve N değerleri aşağıda verilen tablodaki uygun olan m değerine göre karar verilir.

Tablo.4.4. Blok Uzunluğuna Göre Kullanılması Gereken K ve N Değerleri

M	K	N
8	3	16
128	5	49
10^4	6	75

N değeri toplam dizi sayısının blok sayısına bölümünden bulunmaktadır.

$$X^2(\text{obs}) = \frac{(4-16(0.2148))^2}{16(0.2148)} + \frac{(9-16(0.3672))^2}{16(0.3672)} + \frac{(3-16(0.2305))^2}{16(0.2305)} + \frac{(0-16(0.1875))^2}{16(0.1875)} = 4.882605$$

P-Değeri = igamc $\left(\frac{K}{2}, \frac{X^2(\text{obs})}{2}\right)$ (4.6)

$= \text{igamc}\left(\frac{3}{2}, \frac{4.882605}{2}\right) = 0.180 \geq 0.01$ ise dizi rastgeledir.

4.2.5. Rank Testi

Bu testte sabit uzunluklu bit blokları kullanılarak, her biri bir satırı belirtecek şekilde, bir matris oluşturulur ve matrisin rankı hesaplanarak bloklar arasındaki lineer bağımlılık incelenir.

n: Bit dizisinin boyutu

M: Her bir matristeki satır sayısı. Test için bu sayı 32 kabul edilir.

Q: Her bir matristeki sütun sayısı. Test için bu sayı 32 kabul edilir.

N: Matris sayısı $N = \left[\frac{\mathrm{n}}{\mathrm{MQ}}\right]$

$\mathbf{R_1}$**:**Her bir matrisin rankı

$\mathbf{F_M}$**:** R_1=M olan matris sayısı

$\mathbf{F_{M-1}}$**:** R_1=M-1 olan matris sayısı

N-$\mathbf{F_M}$**-**$\mathbf{F_{M-1}}$**:** Arta kalan matrislerin sayısı

$$X^2(\mathrm{obs})=\frac{(\mathrm{F}_M-0.2888N)^2}{0.2888N}+\frac{(\mathrm{F}_{M-1}-0.5776N)^2}{0.5776N}+\frac{(\mathrm{N}-\mathrm{F}_M-\mathrm{F}_{M-1}-0.1336N)^2}{0.1336N}$$

P-Değeri: $\mathrm{e}^{-x^2(obs)/2}$ (4.7)

P-Değeri ≥ 0.01 olduğundan dizi rassal kabul edilir.

4.2.6. Ayrık Fourier Dönüşüm Testi

Bu test dizinin hızlı Fourier dönüşümündeki tepeciklerin yüksekliklerini sınamaktadır. Bu test dizide rastsallığı engelleyici herhangi bir baskın harmoninin olup olmadığını sınamaktır. Bu testin amacı ise rasgelelik varsayımından bir sapma gösteren, test edilen sıradaki periyodik özellikleri (yani, birbirine yakın olan tekrarlı kalıpları) tespit etmektir. Tepe yüksekliklerinin %95 i aştığı durumlar rasgelelik için iyi sayılmaktadır.

n: Bit dizisinin boyutu

d: %95 eşik değerinden fazla olan elemanların beklenen ve gerçekleşen sayısı arasındaki farkın normalize değeri.

T: %95 zayıflıktaki eşik değeri $\mathrm{T}=\sqrt{\left(\log\frac{1}{0.05}\right)^2}$

M: Modül(S´) ≡ |S'|, S´bit dizisinin ilk n/2 elemanı içindeki alt dizileri temsil eder ve modül fonksiyonu zayıf eşik değerlerinin serisini üretir.

$\mathbf{N_0}$**:** Teorik olarak T değerinden daha az sayıda beklenen eşik değeri sayısı

$\mathbf{N_0}=0.95\mathrm{n}/2$

$\mathbf{N_1}$**:**Gözlenen ve *M*'deki T değerinden daha az sayıda beklenen eşik değeri sayısı

$$d = \frac{(N_1 - N_0)}{\sqrt{n\,(0.95)(0.05)/4}}$$

P-Değeri: $\text{erfc}\left(\frac{|d|}{\sqrt{2}}\right)$ (4.8)

P-Değeri ≥ 0.01 olduğunda dizi rassal kabul edilir.

4.2.7. Örtüşmeyen Şablon Eşleştirme Testi

Bu test birbirinden farklı olup birbiriyle kesişmeyen m uzunluğundaki alt dizilerin sayısını hesaplamaktadır. Bu sayının rastsallık için en büyük olması gerekir. Önceden belirlenmiş hedef dizisinin bulunma sıklığının gözlenmesine dayanır. Bu testin amacı, üretecin oluşturduğu periyodik olmayan örneklerin tespit edilmesidir. Bu test ve bir sonraki test olan örtüşen şablon eşleştirme testlerinde, *m* bitlik bir örneği aramak için *m* bitlik bir pencere kullanılır. Eğer bu örnek bulunmaz ise pencere bir bit kaydırılarak arama işlemine devam edilir. Eğer örnek bulunur ise, pencere bulunan örnekten sonraki ilk bite yeniden konumlanır ve arama işlemi devem eder. Yani *m* bitlik bir bloğun dizi içinde tekrarını inceler. Eşleşme bulunamazsa blok 1 bit kaydırılır. Tekrar edilmesi halinde, tekrar edilen bloktan itibaren m bit öteleme yapılarak yeni bir m bitlik blok oluşturulur.

m : Her bir şablonun bit uzunluğu. Şablon hedef dizidir.

n: Test edilen bütün bit dizisi uzunluğu

ε: RSÜ veya SRSÜ ile üretilen bit dizisi

B:Eşleştirilecek m bit şablon; B test kodu içerisinde bulunan periyodik olmayan örnekler şablon kütüphanesinde tanımlı 0 ve 1'lerden oluşan dizidir.

M: test edilecek ε altdizilerinin bit uzunluğu

N: Bağımsız blokların sayısı

Wj (j = 1, …, N): j. blok içinde oluşan *B*'lerin sayısı.

Örnek:

ε = 10100100101110010110 için m=3 ve B=001 alınırsa W_1=2 ve W_2=1 olur. İlgili adımlar aşağıdaki tabloda gösterilmektedir. 20 bitlik dizi 10 bitlik 1010010010 ve 1110010110 iki bloğa ayrılır.

Tablo.4.5 Blok İçerisindeki *B* Şablonlarının Bulunma Sayısı

Bit Pozisyonları	**Blok 1**		**Blok 2**	
	Bitler	W_1	**Bitler**	W_2
1-3	101	0	111	0
2-4	010	0	110	0
3-5	100	0	100	0
4-6	001 (bulundu)	1 arttırma	001 (bulundu)	1arttırma
5-7	Test edilmedi		Test edilmedi	
6-8	Test edilmedi		Test edilmedi	
7-9	001	2 arttıma	011	1
8-10	010(bulundu)	2	110	1

$$\mu = {(M - m + 1)}/{2^m} \qquad \sigma^2 = M\left(\frac{1}{2^m} - \frac{2m-1}{2^{2m}}\right) \tag{4.9}$$

$$X^2(obs) = \sum_{j=1}^{N} \frac{(W_j - \mu)^2}{\sigma^2} \tag{4.10}$$

P-Değeri: $\text{igamc}\left(\frac{N}{2}, \frac{X^2(obs)}{2}\right)$ (4.11)

P-Değeri ≥ 0.01 olduğunda dizi rassal kabul edilir.

4.2.8. Örtüşen Şablon Eşleştirme Testi

Bu test örtüşmeyen şablon eşleştirme testine benzer olarak farklı alt dizileri kontrol etmektedir. Örtüşmeyen şablon eşleştirme testinden farklı olarak burada alt diziler birbiriyle kesişebilmektedir. Yani *m* bitlik bir bloğun dizi içinde tekrarını inceler. Tekrar edilmesi halinde, blok 1 bit ötelenerek yenisi oluşturulur. Non-overlapping Template Matching testten tek farkı eşleşme bulunduğunda bloğun sadece 1 bit ötelenmesidir

m : Her bir şablonun bit uzunluğu. Şablon hedef dizidir.

n: Test edilen bütün bit dizisi uzunluğu

ε: RSÜ veya SRSÜ ile üretilen bit dizisi

B: Eşleştirilecek m bit şablon; B test kodu içerisinde bulunan periyodik olmayan örnekler şablon kütüphanesinde tanımlı 0 ve 1'lerden oluşan dizidir.

K:Bağımsızlık derecesi sayısı burada K 5 olarak sabitlenmiştir.

M: Test edilecek ε alt dizilerinin bit uzunluğu M test kodunda 1032 alınmıştır.

N: Bağımsız blokların sayısı. N test kodunda 968 alınmıştır.

πi : Teorik olasılıklar

vi (i = 0, …, 5): i. blok içinde oluşan *B*'lerin sayısı

Örnek:

ε = 10111011110010110100011100101110111110000101101001 *n*=50 *K*=2 *M*=10 ve *N*=5 alınmıştır. Sonra dizi 1011101111, 0010110100, 0111001011, 1011111000 ve 0101101001 şeklinde 5 bloğa bölünmüştür. Her bir *N* bloktaki *B* şablonunun sayısı hesaplanır. *m*=2 için *B*=11 alınırsa ve ilk blokta 1011101111 aranırsa;

Tablo.4.6. Blok İçerisindeki *B* Şablonlarının Bulunma Sayısı

Bit Pozisyonu	**Bitler**	***B* = 11 in bulunma sayısı**
1-2	10	0
2-3	01	0
3-4	11 (bulundu)	1 arttırma
4-5	11 (bulundu)	2 arttırma
5-6	10	2
6-7	01	2
7-8	11 (bulundu)	3 arttırma
8-9	11 (bulundu)	4 arttırma
9-10	11 (bulundu)	5 arttıma

İlk blokta arama yapılır. 11'e 5 defa rastlanmıştır. v_5 arttırılır ve v_5=0, v_5=0, v_5=0, v_5 =0 ve v_5=1.

ʎ ve η değerleri hesaplanır.

$$ʎ=(M-m+1)/2^m \qquad \eta= ʎ/2 \tag{4.12}$$

Bu örnek için $ʎ=(10-2+1)/2^2=2.25$ ve $\eta=2.25/2=1.125$ değerleri bulunur.

$$X^2(\text{obs})= \sum_{i=0}^{5} \frac{(v_i - N\pi_i)^2}{N\pi_i} \tag{4.13}$$

$$X^2(\text{obs}) = \frac{(0-5*0.324652)^2}{5*0.324652} + \ldots + \frac{(0-5*0.166269)^2}{5*0.166269} = 3.167729$$

P-Değeri = igamc$\left(\frac{\mathrm{K},}{2}\ \frac{\mathrm{X}^2(\mathrm{obs})}{2}\right) = \left(\frac{5}{2}, \frac{3.167729}{2}\right)$= 0.274932 (4.14)

P-Değeri ≥ 0.01 olduğunda dizi rassal kabul edilir.

4.2.9. Maurer's Evrensel Testi

Verilen dizinin yeterince sıkıştırılıp sıkıştırılamayacağını kontrol eder. Dizinin fazlasıyla sıkıştırılması, dizinin rassallıktan uzak olduğunu gösterir. Testte, dizi *L* bitlik bloklara ayrılır. Bu blokların bir kısmı testin başlangıç kısmında uygulanır. Testte kullanılan referans dağılımı yarım normal dağılımdır. *L*-bitlik kalıpların birbirlerini ne kadar sıklıkla tekrar ettiği hesaplanır ve bu değerler beklenen değerler ile karşılaştırılır. Blok uzunluğu 6 seçildiğinde, dizi uzunluğu en az 387,840 olmalıdır.

L :Her bir bloğun büyüklüğü.

Q : Başlangıç dizisindeki blokların sayısı.

n : Bit dizisinin uzunluğu.

fn : Eşleşen L-bitlik şablonlar arasındaki mesafelerin log2 tabanında toplamı

ε : Teste tabi tutulan bit dizisi

$\varepsilon = \varepsilon_1\ \varepsilon_2 \ldots\ldots \varepsilon_n$

K : Test edilen blokların sayısı

T_j : Her bir L-bitlik bloğun blok sayısını tutan j'ye bağlı tablo değeri

sum : K bloklarında tespit edilen farklılıkların log2 tabanında toplamı

σ : Standart sapma.

c : Sezgisel yaklaşım.

$n \geq (Q + K) \times L$

$6 \leq L \leq 16$

$Q=10.\ 2^L$

$K=\left(\frac{N}{L}\right) - Q\ \approx 1000 \times 2^L$ (4.15)

L, Q, n değerleri aşağıdaki tabloya göre seçilir;

Tablo.4.7. Blok İçerisindeki L, Q, n Değerleri

n	L	Q=10.$\mathbf{2^L}$
≥387.840	6	640
≥904.960	7	1.280
≥2.068.480	8	2.560
≥4.654.080	9	5.120
≥1.342.400	10	10.240
≥22.753.280	11	20.480
≥49.643.520	12	40.960
≥107.560.960	13	81.920
≥231.669.760	14	163.840
≥496.435.200	15	327.680
≥1.059.061.760	16	655.360

$$f_n = \frac{1}{K} \sum_{i=Q+1}^{Q+K} \log_2(i - T_j) = \frac{sum}{K} \tag{4.16}$$

$$\textbf{P-Değeri} = \text{erfc}\left(\left|\frac{\mathrm{f_n - BeklenenDeğer(L)}}{\sqrt{2}\,\sigma}\right|\right) \tag{4.17}$$

BeklenenDeğer(L) aşağıdaki tablodan elde edilir:

Tablo.4.8. BeklenenDeğer Sonuçları

L	BeklenenDeğer	Varyans
6	5.2177052	2.954
7	601962507	3.125
8	7.1836656	3.238
9	8.1764248	3.311
10	9.1723243	3.356
11	10.170032	3.384
12	11.168765	3.401
13	12.168070	3.410
14	13.167693	3.416
15	14.167488	3.419
16	15.167379	3.421

$$\sigma = c\sqrt{\frac{\text{varyans(L)}}{K}} \tag{4.18}$$

$$c = 0.7 - \frac{0.8}{L} + \left(4 + \frac{32}{L}\right) x \frac{K^{\frac{3}{L}}}{15} \tag{4.19}$$

P-Değeri ≥ 0.01 olduğunda dizi rassal kabul edilir.

4.2.10. Doğrusal Karmaşıklık Testi

Bit dizisinin LFRS (linear feedback shift register) uzunluğuna bakarak kompleksliğini inceler. Test dizinin rassallık için yeterince karmaşık olup olmadığını kontrol eder. Diziler LFSR çıktıları olarak kabul edilir ve diziyi oluşturabilecek en küçük LFSR'ın boyu küçükse, dizinin rassal olmak için yeterince karmaşık olmadığına karar verilir. Testte dizi *M* bitlik bloklara ayrılır ve bloktaki bitlerin lineer karmaşıklıkları Berlekamp-Massey algoritması kullanılarak hesaplanır. Hesaplanan lineer karmaşıklıkların beklenen dağılıma uygun olup olmadıklarına bakılır. Testte kullanılan referans dağılım ki-kare dağılımıdır. Testin geçerli olabilmesi için dizinin boyu en az 1,000,000; blok uzunluğu da 500 ve 5000 arasında olmalıdır.

M:Bloktaki bit uzunluğu

n: Bit dizisinin uzunluğu. n=M.N

N: M bitlik bağımsız blok sayısı

K: Serbestlik derecesi. Test kodunda *K = 6* olarak alınmıştır.

T_i: Alt dizi sayısı

$$\mu = \frac{M}{2} + \frac{(9 + (-1)^{M+1})}{36} - \frac{(M/3 + 2/9)}{2^M} \tag{4.20}$$

$$T_i = (-1)^M . (L_i - \mu) + 2/9 \tag{4.21}$$

T_i değerleri için v_0, …..v_6 değerleri şu şekilde hesaplanır:

$T_i \leq$ -2.5	v_0 1 arttırılır
-2.5 < $T_i \leq$ -1.5	v_1 1 arttırılır
-1.5 < $T_i \leq$ -0.5	v_2 1 arttırılır
-0.5 < $T_i \leq$ 0.5	v_3 1 arttırılır
0.5 < $T_i \leq$ 1.5	v_4 1 arttırılır
1.5 < $T_i \leq$ 2.5	v_5 1 arttırılır
$T_i >$ 2.5	v_6 1 arttırılır

$$X^2(\text{obs}) = \sum_{i=0}^{K} \frac{(v_i - N\pi_i)}{N\pi_i}$$

P-Değeri= igamc $\left(\frac{K}{2}, \frac{X^2(\text{obs})}{2}\right)$ (4.22)

P-Değeri ≥ 0.01 olduğunda dizi rassal kabul edilir.

4.2.11. Seri Testi

Tekrar eden *m* bitlik *2m* tane bloğun tekrar sayısının dağılımını inceler. *m*=1 için, birinci teste denktir. $\nabla\psi^2(obs)$ ve $\nabla^2\psi^2(obs)$ değerleri *m*-bitlik öbeklerin frekansını hesaplamak için kullanılır.

$$\psi_m^2 == \frac{2^m}{n}\sum_{i_1..,i_m} v_{i_1..,i_m}^2 - n$$

$$\psi_{m-2}^2 = \frac{2^{m-2}}{n}\sum_{i_1..,i_{m-2}} v_{i_1..,i_{m-2}}^2 - n$$

$$\psi_{m-1}^2 = \frac{2^{m-1}}{n}\sum_{i_1..,i_{m-1}} v_{i_1..,i_{m-1}}^2 - n$$

$$\nabla\psi_m^2 = \psi_m^2 - \psi_{m-1}^2 \qquad \nabla^2\psi_m^2 = \psi_m^2 - 2\psi_{m-1}^2 + \psi_{m-2}^2 \quad (4.23)$$

P-Değeri.1 = igamc$(2^{m-2}, \nabla\psi_m^2/2)$ ve

P-Değeri.2 = igamc$(2^{m-3}, \nabla^2\psi_m^2/2)$ (4.24)

Örnek:

ε=10110110101101100010101010101011100111000110110000101000110010011010101001

10101001010110011001110011 0

v_{000}= 4 v_{001}= 12 v_{010}=18 v_{011} =15

v_{100}= 12 v_{101}= 20 v_{110}=15 v_{111} = 2

v_{00}= 16 v_{01}= 32 v_{10}=33 v_{11} = 18

v_0= 49 v_1=51

$$\psi_3^2 = \frac{2^3}{100}(16 + 144 + 324 + 225 + 144 + 400 + 225 + 4) - 100 = 18.56$$

$$\psi_2^2 = \frac{2^2}{100}(256 + 1024 + 1089 + 324) - 100 = 7.72$$

$$\psi_1^2 = \frac{2^1}{100}(2401 + 2601) - 100 = 0.04$$

$\nabla\psi_3^2$ = 10.84

$\nabla^2\psi_3^2$ = 3.16

P-Değeri.1 = igamc(2, 10.84) ve

P-Değeri.2 = igamc(1, 3.16)

P-Değeri $<$ 0.01 olduğundan dizi rassal kabul edilmez.

P-Değeri $\geq$ 0.01 olduğundan dizi rassal kabul edilir.

4.2.12. Yaklaşık Entropi Testi

Tekrar eden *m* ve *(m+1)* bitlik blokların entropisini inceler.Bu test, *m* bitlik kesişen blokların frekansları üzerine odaklaşır ve bu frekansları *m* ve *(m+1)* bitlik bloklar için beklenen değerler ile karşılaştırır. Testte kullanılan referans dağılımı ki-kare dağılımıdır. Diziden kesişen *n* tane *m*-bitlik blok üretilir. Bu blokların frekansları ve entropisi hesaplanır. Aynı işlemler blok uzunluğu *m+14* için tekrarlanır. *m* ve *m*+ 1 bit için hesaplanan değerlerin farkına bağlı test istatistiği hesaplanır. Bu farkın düşük olması rassallıktan uzaklığı gösterir.

m: Her bir blok uzunluğu. Bu durumda teste kullanılan ilk blok uzunluğudur. *m+1* ise kullanılan 2. blok uzunluğudur.

n: Tüm bit dizisinin uzunluğu

$\boldsymbol{C_i^m}$**:** Her bir i değeri için hesaplanan m bitlik blok sayısı

$$ApEn(m)=\varphi^{(m)} - \varphi^{(m+1)} \quad (4.25)$$

$$C_i^m = \frac{\#i}{n}$$

$\varphi^{(m)} = \sum_{i=0}^{2^m-1} \pi_i \log\pi_i$ değeri hesaplanır. $\pi_i = C_j^3$ $j=\log_2 i$

(m+1) bitlik bloklar için de bir önceki denklem değerleri hesaplanır.

$$X^2 = 2n[\log 2 - ApEn(m)]$$

***P*-Değeri**= igamc(2^{m-1}, $\frac{X^2}{2}$). (4.26)

P-Değeri $\geq$ 0.01 olduğunda dizi rassal kabul edilir.

4.2.13. Birikimli Toplamlar Testi

Bit dizisini ardışık uzunluklu bloklara ayırıp blokların 1 ve 0 dengesini belirler ve bloklar arasındaki dengesizlik farkına bakar.

n: Bit dizisinin uzunluğu

mod: Test uygulaması dizinin başından sonuna doğru yapılırsa mod=0, sondan başa doğru yapılırsa mod=1'dir.

$\mathbf{S_i}$ = Artarak büyüyen alt dizilerin toplamı

Tablo.4.9. Blok İçerisindeki Kullanılması Gereken mod Formülleri

Mod=0	Mod=1
$S_1=X_1$	$S_1=X_n$
$S_2=X_1+X_2$	$S_2=X_n+X_{n-1}$
$S_3=X_1+X_2+X_3$	$S_3=X_n+X_{n-1}+X_{n-2}$
…	…
…	…
$S_k=X_1+X_2+\ldots.X_k$	$S_k=X_n+X_{n-1}+\ldots.X_{n-k+1}$
….	….
…	…
$S_n=X_1+X_2+\ldots.X_n$	$S_n=X_n+X_{n-1}+\ldots.X_{k-1}+\ldots..X_1$

$z=max_{1\le k\le n}\,|S_k|$

$\mathbf{\Phi}$ = Normal Birikimli Olasılık Dağılım Fonksiyonu = $\frac{1}{\sqrt{2\pi}}\int_{-\infty}^{z} e^{-u^2}/2\ du$

$$\textbf{P-Değeri}=1-\sum_{k=\left[-\frac{n}{z}+1\right)/4}^{\left[\frac{n}{z}-1\right)/4}\left[\Phi\left(\frac{(4k+1)z}{\sqrt{n}}\right)-\Phi\left(\frac{(4k-1)z}{\sqrt{n}}\right)\right]+\sum_{k=\left[-\frac{n}{z}-3\right)/4}^{\left[\frac{n}{z}-1\right)/4}\left[\Phi\left(\frac{(4k+3)z}{\sqrt{n}}\right)-\Phi\left(\frac{(4k+1)z}{\sqrt{n}}\right)\right] \quad (4.27)$$

P-Değeri ≥ 0.01 olduğunda dizi rassal kabul edilir.

4.2.14. Rasgele Gezinim Testi

Bit dizisini ardışık uzunluklu bloklara ayırıp blokların 1 ve 0 dengesini belirler ve daha sonra blokların dengesinin dağılımını inceler. Bu test rastsal gezinimler içindeki durum yürüyüşlerinin sayısının beklenen değeri aşıp aşmadığını kontrol etmektedir. Eğer beklenilen

değer aşılıyorsa, dizinin rastsal olmadığı kabul edilir. Bu testin amacı, bu döngü sırasında rasgele dizide beklenen sapmadan kaynaklanan belirli bir durumun ziyaretlerinin sayısının belirlenmesidir. Bu test aslında 8 testten ve çıkarımlarından oluşan bir seridir. Burada kullanılan referans dağılımı ki kare testidir.

n: Bit dizisinin uzunluğu

X: Bit dizisindeki 0 ve 1'lerin toplamı (0= -1, 1=1 alınarak yapılan aritmetik toplama sonucu).

S_i: Her defasında X_1'den başlanarak adım adım arttırılan kısmi toplamlar

$S_1 = X_1$

$S_2 = X_1 + X_2$

..

..

$S_k = X_1 + X_2 + \ldots\ldots\ldots\ldots + X_k$

...

$S_n = X_1 + X_2 + \ldots\ldots\ldots\ldots + X_k + \ldots\ldots\ldots X_n$

S′ : Oluşturulan S dizisinin başına ve sonuna 0 eklenerek oluşturulan yeni alt dizi

***J*:** *S*′ dizisinde geçen sıfırların sayısıdır. Aynı zamanda *S′*'deki döngü sayısıdır. Döngü sayısı sıfır ile başlayan ve biten dizilerin sayısına bağlıdır.

x: Her bir döngü ve sıfır içermeyen durum sayısı. $-4 \le x \le -1$ ve $1 \le x \le 4$

v_k (x): *k*'ya bağlı *x* koşulunun tüm döngülerdeki toplam sayısı. $k = 0,1,..,5$ için

$$\sum_{k=0}^{5} v_k\,(x) = J$$

$$X^2(obs) = \sum_{k=0}^{5} \frac{(v_k\,(x) - J\pi_k\,(x))^2}{J\pi_k\,(x)}$$

$\pi_k\,(x)$: x koşulunun rassal bir dağılımda k kez oluşma durumudur.

P-Değeri $= igamc\left(\frac{5}{2}, \frac{X^2}{2}\right)$ (4.28)

P-Değeri ≥ 0.01 olduğunda dizi rassal kabul edilir.

ε =0110110101

X= -1,1,1 -1,1,1 -1,1 -1,1

S= {-1,0,1,0,1,2,1,2,1,2}

S' = 0, -1,0,1,0,1,2,1,2,1,2,0

J = 3 {0,-1,0}, {0,1,0} , {0,1,2,1,2,1,2,0}

Tablo.4.10. Blok İçerisindeki Döngülerin Bulunma Sayısı

x	Döngü Sayıları					
	0	1	2	3	4	5
-4	3	0	0	0	0	0
-3	3	0	0	0	0	0
-2	3	0	0	0	0	0
-1	2	1	0	0	0	0
1	1	1	0	1	0	0
2	2	0	0	1	0	0
3	3	0	0	0	0	0
4	3	0	0	0	0	0

$v_0(-1) = 2$ (-1 durumu 0. döngüde 2 kez meydana geldiği için)

$v_1(-1) = 1$ (-1 durumu 0. döngüde de 1 kez meydana geldiği için)

$v_2(-1) = v_3(-1) = v_4(-1) = v_5(-1) = 0$

4.2.15. Rasgele Gezinim Değişken Testi

Bit dizisini ardışık uzunluklu bloklara ayırıp blokların 1 ve 0 dengesini belirleyip ortalama değerden sapma miktarını belirler. Bir önceki Rassal Farklılık Testi ile aynı stratejiyi kullanır. Bu testin amacı farklı durum sayılarının beklenen değerden sapma gösterip göstermediğini anlamaktır.

ξ(x): Tüm *J* döngülerindeki *x* durumlarının meydana gelme sayısı

***P*-Değeri** = $\text{erfc}\left(\frac{|\xi(x) - J|}{\sqrt{2J(4|x|-2)}}\right)$ şeklinde hesaplanır. (4.29)

P-Değeri ≥ 0.01 olduğunda dizi rassal kabul edilir.

5. AKIŞ ŞİFRELEME ALGORİTMALARI VE RASGELE SAYI ÜRETECİ OLARAK KULLANILMASI

5.1. Akış Şifrelerinin Genel Yapısı

Akış şifreler açık metnin bir seferde bir karakterine (genellikle ikili sayı) zamanla değişen bir şifreleme fonksiyonu uygulayarak açık metin karakterlerini ayrı ayrı şifreler**.** Blok şifreler ise açık metnin bir bloğunu sabit bir şifreleme fonksiyonu kullanarak şifreleme işlemini gerçekleştirir.[17][18][19]

Akış şifreleri genellikle blok şifrelerden daha verimli bir şekilde üretilebilirler. Bu yüzden, akış şifreleri şifreleme dünyasında uzun süre hüküm sürmüşlerdir. Hızlı blok şifrelerinin belirmesi ve blok şifrelerin CM, OFB veya CBC modlarında akış-benzeri davranabilmeleri ilginin blok şifrelere kaymasına sebep olmuştur. Üstelik, genellik güvenli bir akış şifresi tasarlamak güvenli bir blok şifre tasarlamaktan çok daha zordur**.** 2000-2003 yılları arasında NESSIE yarışması ve 2004 te başlayan eSTREAM projesi güvenli akış şifresi seçmeyi amaç edinmiştir. Ancak güvenli bir akış şifresi tasarlamanın zor bir iş olduğu da NESSIE yarışmasında ortaya çıkmıştır. Bununla beraber eSTREAM projesinde 35 farklı akış şifresi yarışmakta ve bu şifrelerin değerlendirilmesi devam etmektedir.

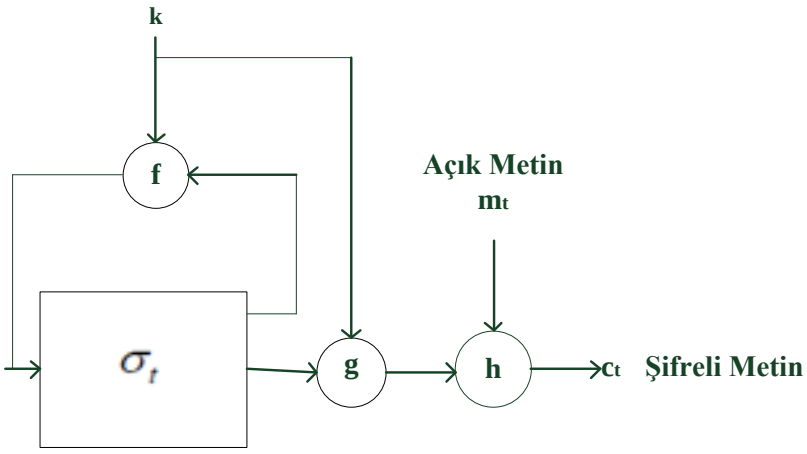
Vernam şifreleme olarak ta bilinen akış şifreleri Shannon"un tek kullanımlık şerit (one time pad) kavramından yola çıkılarak ortaya çıkarılmıştır. Tek kullanımlık şerit uzun bir anahtar akış kümesi kullanır ve bu küme rasgele seçilmiş birçok bit grubu içerir. Bu anahtar akışı açık metnin bütün bitleri ile tek tek birleştirilir. Akış şifreler girdi olarak alınan bir anahtar *(K)* ve başlangıç vektörü (*IV*-Initialization Vector) ile mümkün olduğu kadar uzun periyotlu ve rassal gözüken anahtar dizilerini üretir ve elde ettiği anahtarı bir fonksiyona (genellikle XOR işlemi) sokarak şifreli metni elde eder. Şekil 5.1'de bir akış şifrenin şifreli metin üretme safhası ile beraber örnek gösterimi verilmiştir.

Şekil 5.1. XOR Fonksiyonu ile Akış Şifre Gösterimi

Akış şifreler (Stream Cipher), açık metnin bir seferde bir karakterine (genellikle ikili sayı) zamanla değişen bir şifreleme fonksiyonu uygulayarak açık metin karakterlerini ayrı ayrı şifreler.

Blok şifreler ise açık metnin bir bloğunu sabit bir şifreleme fonksiyonu kullanarak şifreleme işlemini gerçekleştirir. Genellikle donanımsal yönden bakıldığında akış şifreler blok şifrelerden daha hızlıdır. Bazı durumlarda da daha uygundurlar. Uygun olduğu durumlara örnek olarak şifreleme işleminin tek karakterler üzerinde ayrı ayrı gerçekleştirilmesi istenen ya da arabellek (buffer) işleminin sınırlı olduğu yerler verilebilir. Ayrıca iletim hatalarının çok yüksek olasılıklı olduğu yerlerde avantajları vardır. Akış şifreler, eşzamanlı (senkron) ve asenkron akış şifreler olmak üzere ikiye ayrılabilir. Eşzamanlı akış şifreler sonlu durum makinesidir (finite state machine). Anahtar dizisi (keystream), açık metin ve şifreli metinden bağımsız olarak gizli anahtardan üretilir ve anahtar dizisi ve şifreli metnin üretimi (5.1) deki eşitliklerle $t \geq 0$ olmak üzere tanımlanabilir.

$$\sigma_{t+1} = \mathrm{f}\,(\alpha_t, \mathrm{k}) \qquad (5.1)$$
$$s_t = \mathrm{g}\,(\sigma_t, \mathrm{k})$$
$$c_t = \mathrm{h}(s_t, m_k)$$



Şekil.5.2. Senkron Bir Şifrenin Yapısı

Şekil 5.2, eşzamanlı bir şifrenin yapısını göstermektedir. Burada σ_0 başlangıç durumunu (initial state) (anahtar k 'ya bağlı olabilir), k anahtarı, f diğer durum fonksiyonunu, g anahtar dizisi s_t 'yi üreten fonksiyonu, h ise açık metin ile anahtar dizisini (keystream) birleştirerek şifreli metin c_t'yi üreten çıkış fonksiyonunu temsil etmektedir. h çıkış fonksiyonu yerine XOR fonksiyonu kullanılırsa bu tür şifrelere toplamsal akış şifreler (additive stream cipher) adı

verilir. Bu tür şifrelere yalancı random üreteçleri (pseudo random number generator) ya da anahtar dizisi üreteçleri (keystream generator) de denmektedir. Bilindiği gibi XOR işleminin tersi kendisidir ($h = h^{-1}$). Bundan dolayı bu tür şifrelerde şifreleme ile deşifreleme aynıdır ve bu işlemin kullanılması karşımıza kullanışlı bir özellik olarak çıkar.

Bu tür şifreler iletim hatalarına karşı zayıf değillerdir çünkü her karakter bağımsız olarak şifrelenmektedir. Ancak bir saldırgan şifreli bir metni silebilir ya da değiştirebilir. Dolayısıyla gönderilen mesajın kimlik denetiminin (authentication) yapılmasını sağlayan mekanizmalara gereksinim vardır. Aynı nedenden dolayı senkronizasyon bozulabilir. Gönderen ve alıcı arasında iyi bir senkronizasyon sağlanmalı ve senkronizasyon bozulmasını sezecek mekanizmalar kullanılmalıdır. Toplamsal akış şifreler (additive stream cipher), One Time Pad (Tek Kullanımlık Şifreler) şifrelere anlayış olarak çok benzemektedir. Bu yöntemden farklı olarak gizli anahtar başlangıç durumu ya da üreteci beslemek için kullanılır ve yalancı rastlantısal (pseudo-random) bitler üretilir.

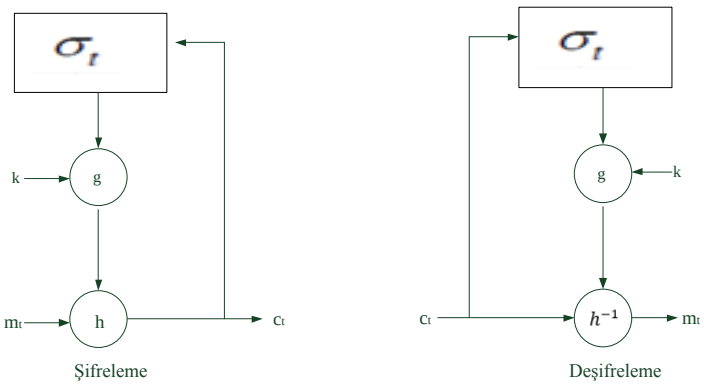
Diğer akış şifre tipi olan asenkron akış şifreler de sonlu durum makinesidir. Ancak anahtar dizisi, sabit uzunluktaki bir önceki şifreli metinlerin ve anahtarın bir fonksiyonu ile elde edilir. Asenkron şifreler de şifreli metin üretme işlemi (5.2) deki eşitliklerle $t \geq 0$ olmak üzere tanımlanabilir.

$$\sigma_t = (c_{t-v}, c_{t-v+1}, \ldots\ldots\ldots\ldots c_{t-1}) \qquad (5.2)$$

$$s_t = \mathrm{g}(\sigma_t, \mathrm{k})$$

$$c_t = \mathrm{h}(s_t, m_k)$$

Burada σ_0 başlangıç durumunu (initial state), k anahtarı, g anahtar dizisi s_t 'yi üreten fonksiyonu, h ise açık metin ile anahtar dizisini (keystream) birleştirerek şifreli metin c_t 'yi üreten çıkış fonksiyonunu temsil etmektedir. Başlangıç durumu $\sigma_0 = (c_{t-v}, c_{t-v+1}, \ldots\ldots\ldots c_{t-1})$ herkes tarafından bilinebilir. Asenkron akış şifrenin yapısı Şekil 5.3'de görülmektedir.

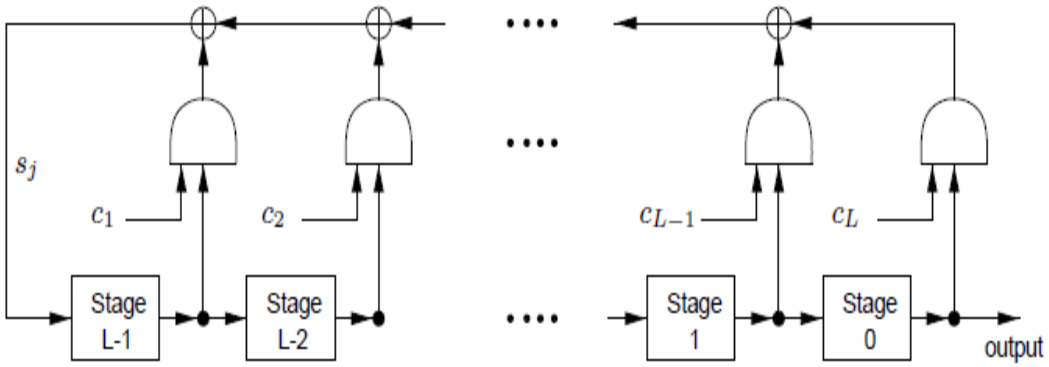


Şekil.5.3. Asenkron Şifrenin Genel Yapısı

Şifreleme ve deşifreleme senkron şifrelere göre şekilde görüldüğü üzere farklılık göstermektedir. Asenkron şifrelerde şifreleme *v* şifreli metin sembolüne bağlı olduğu için bir iletim hatası durumunda *v* sembol sonra şifrenin tekrar senkronizasyonu mümkün olacaktır. Böyle bir durum söz konusu olduğunda öteki *v* sembol hatalı olacaktır. Yani hata yayılması (error propagation) senkron olan şifrelere göre kötüdür. Ancak senkronizasyon düşünüldüğünde asenkron şifreler senkron olanlara göre daha iyidir. Çünkü bu tür şifrelerde *v* doğru şifreli metin sembolü elde edildikten sonra senkronizasyon kendiliğinden sağlanacaktır. Senkron şifreler ise senkronizasyonu tekrar sağlayamazlar.

5.1.1. Doğrusal Geri Beslemeli Öteleyici Saklayıcılar

Doğrusal Geri Beslemeli Öteleyici Saklayıcılar birçok anahtar dizisi üretecinde kullanılmaktadır.[25] Bunun nedeni olarak donanımsal uygulamalarda uygunlukları, geniş periyoda sahip olmaları, üretilen serinin iyi istatistiksel özellikler göstermesi ve cebirsel tekniklerle kolayca analiz edilebilmeleri gösterilebilir. *L* uzunluğunda bir LFSR (Linear Feedback Shift Register- Doğrusal Geri Beslemeli Öteleyici Saklayıcılar) F_q üzerine bir sonlu durum otomatıdır (finite state automation) ve F_q elemanlarının yarı-sonsuz bir serisini üretir. Şekil.5.4'te görüldüğü gibi *L* uzunluğunda bir LFSR, 0'dan $L-1$'e kadar numaralanmış her biri bir bit depolayabilme yeteneği olan *L* tane gecikme ünitesi içerir. Ayrıca her gecikme hücresi bir giriş ve bir çıkışa ve verinin hareketini kontrol eden bir saate sahiptir.

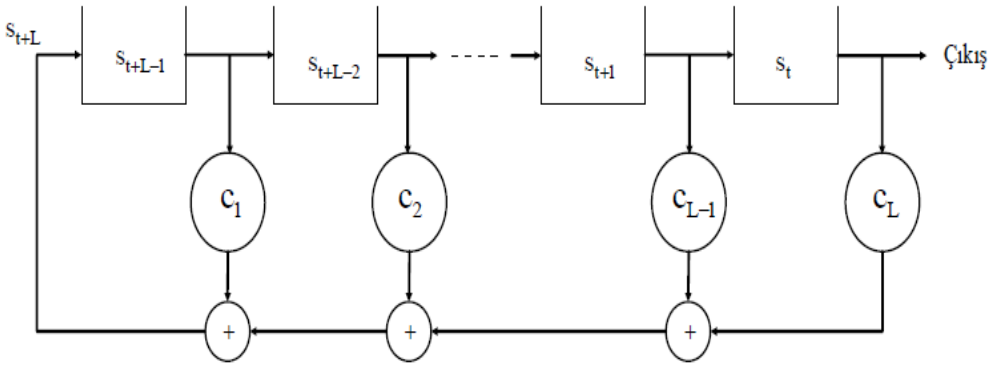


Şekil.5.4. *L* Uzunluğundaki Bir Doğrusal Geri Beslemeli Saklayıcı

Bir LFSR, $s = (s_t) = s_0, s_1, \ldots\ldots$ olmak üzere F_q üzerine derece *L*'ye sahip doğrusal tekrarlayan bir ilişkiye sahiptir ve bu ilişki (5.3) eşitliğinde gösterilmiştir.

$$s_{t+L} = \sum_{i=1}^{L} c_i \, s_{t+L-i} \,, \quad \forall\ t \geq 0 \qquad (5.3)$$

L 'nin katsayıları olan c_1, c_2 ,...., c_L F_q'nun elemanlarıdır ve LFSR'ın geri besleme katsayıları olarak isimlendirilir. F_q üzerine L uzunluğunda bir LFSR Şekil 5.5 formundadır.



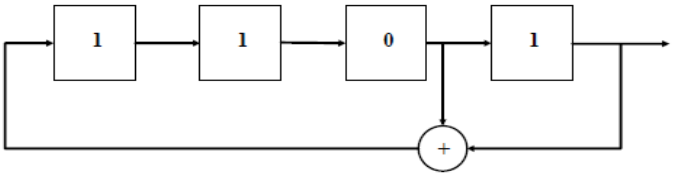
Şekil.5.5. Doğrusal Geri Beslemeli Saklayıcının Genel Yapısı

Şekilde de görüldüğü gibi $1 \leq i \leq L-1$ olmak üzere durum i 'nin içeriği her i için $i-1$ durumuna kaydırılır. Daha sonra $L-1$ durumunun yeni içeriği geri besleme s_{t+L} (s_{t+L} bazı sabit sayıdaki durumların modulo 2 de toplanma) ile elde edilir.

Daha önce de bahsedildiği gibi LFSR'ı oluşturan her biri F_q 'nun bir elemanını içeren L gecikme hücresine evreler (stages) denir. L evrenin içeriği s_t ,..... s_{t+L-1}ise LFSR'ın durumunu (stage) oluşturur. L durum başlangıçta keyfi olarak seçilmiş keyfi F_q'da L eleman ile yüklenir ve başlangıç durumu olan s_0 ,..... s_{L-1} 'i oluşturur. Örnek 5.1'de 4 bit uzunluğunda bir LFSR'ın çıkış değerlerinin elde edilişini göstermektedir.

Örnek 5.1 :

Şekil 5.6'da 4. dereceden ya da 4 bit uzunluğunda bir LFSR'ın geri besleme katsayıları $c_1 = c_2 = 0$, $c_3 = c_4 = 1$ ve başlangıç durumu (s_0 , s_1, s_2, s_3) = (1011) olmak üzere diğer durumları gösterilmektedir. Bu LFSR tarafından üretilen çıkış serisi s_0 , s_1,..... ise 1011100 şeklindedir. Bu LFSR için doğrusal tekrarlayan ilişki $s_{t+4} = s_{t+1} + s_t$ şeklinde verilebilir.



t	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
s_t	1	0	1	1	1	1	0	0	0	1	0	0	1	1	0	1
s_{t+1}	0	1	1	1	1	0	0	0	1	0	0	1	1	0	1	0
s_{t+2}	1	1	1	1	0	0	0	1	0	0	1	1	0	1	0	1
s_{t+3}	1	1	1	0	0	0	1	0	0	1	1	0	1	0	1	1

Şekil.5.6. LFSR ve Ardışık Durumları

Bir LFSR'ın çıkış dizisi onun geri besleme katsayıları ve başlangıç durumundan elde edilir. L uzunluğundaki LFSR'ın c_1,...., c_L katsayıları (4.4) eşitliğinde gösterilen genellikle LFSR geri besleme polinomu (feedback polynomial, connection polynomial) ile temsil edilir.

$$P(X) = 1 - \sum_{i=1}^{L} c_i \ X^i \tag{5.4}$$

Geri Besleme polinomunun karakteristik polinomu da (5.5) denklemindeki gibi verilebilir.

$$P^*(X) = X^L \ P\left(\frac{1}{X}\right) = X^L - \sum_{i=1}^{L} c_i \ X^{L-i} \tag{5.5}$$

Örnek 5.2

İkili bir LFSR'ın geri besleme polinomu P(X) = X^4+ X^3+1 ise onun karakteristik polinomu P^*(X) = X^4 + X +1 olur.

Bir LFSR'ın geri besleme polinomunun derecesi LFSR uzunluğuna eşitse, nonsingular olarak isimlendirilir (Geri besleme katsayısı c_L , 0'dan farklı ise). Bu tür bir LFSR ile üretilen dizinin periyodu q^L −1'i (ikili LFSR'lar için 2^L −1'i) geçemez. LFSR en fazla q^L duruma sahiptir. Bunun ötesinde eğer LFSR singularsa tüm diziler eninde sonunda periyodiktir.

Bir LFSR q^L farklı seri (sequence) üretir ve bunlar F_q üzerinde bir vektör uzayı oluştururlar. Tüm serilerin seti için geri besleme polinomu P , bir seri (s_t) t≥0 olmak üzere aşağıdaki şekilde tanımlanır.

Tanım 5.1 : P geri besleme polinomu ile F_q üzerine L uzunluğunda bir LFSR tarafından ancak ve ancak *deg(V)* < L olmak üzere V∈ F_q *[X]* şeklinde (5.6) ifadesinde gösterildiği gibi

bir polinom varsa bu LFSR tarafından (s_t) t≥0 (aynı şekilde (s_t) t≥0 'nin üreteç fonksiyonu (5.6) daki denklemi sağlar) üretilebilir.

$$\sum_{t\geq 0} s_t X^t = \frac{V(X)}{P(X)} \tag{5.6}$$

Buna ek olarak LFSR'ın başlangıç durumu ve P 'nin katsayıları kullanılarak (5.7) de gösterildiği gibi V(X) elde edilebilir.

$$V(X)= - \sum_{i=0}^{L-1} X^i \left(\sum_{j=0}^{i} c_{i-j}\, s_j\right), \quad P(X)= \sum_{i=0}^{L} c_i x_i \tag{5.7}$$

Yukarıdaki ifade göstermektedir ki *L* uzunluğunda *P* geri besleme polinomuna sahip bir LFSR tarafından üretilen serilerle *deg(V) < L* olmak üzere $\frac{V(X)}{P(X)}$ kesirleri arasında birebir bir ilişki vardır. Bunun sonucu olarak;

- *P* geri besleme polinomuna sahip bir LFSR tarafında üretilen herhangi bir seri *P* 'nin katı olan herhangi bir geri besleme polinomuna sahip bir LFSR tarafından üretilebilir. Bu özellik hızlı ilinti (correlation) saldırılarında (fast correlation attacks) kullanılabilir.
- Diğer yandan *P* geri beslemeli bir LFSR tarafından üretilen herhangi bir dizi geri besleme polinomu P′ olan, eğer $\frac{V(X)}{P(X)}$kesri için OBEB(V,P) ≠1 gerçekleşiyorsa, daha küçük bir LFSR tarafından üretilebilir. Böylece P , F_q üzerine indirgenemez polinom değilse *P* geri besleme polinomuna sahip bir LFSR'ın ürettiği tüm serilerin içerisinden biri daha kısa bir LFSR tarafından üretilebilir.

Bunun ötesinde, doğrusal tekrarlayan seri (linear recurring sequence) (s_t) t≥0 için P_0 ve V_0 birbirlerine göre asal olmak üzere seri (s_t) t≥0 'in üreteç fonksiyonu $\frac{V_0(X)}{P_0(X}$' tir ve bu seri için sabit terimi 1 olan tek bir P_0 polinomu mevcuttur. Dolayısıyla (s_t) t≥0 'ı üreten en kısa LFSR L = max (deg(P_0) , deg(V_0) +1) uzunluğuna sahiptir. P_0'ın reciprocal polinomu, (s_t) t≥0 'ı üreten en kısa LFSR'ın karakteristik polinomudur ve serinin minimal polinomu olarak isimlendirilir. Dolayısıyla o da seri tarafından sağlanan en düşük dereceli tekrarlayan ilişkiyi elde eder. Bir doğrusal tekrarlayan dizinin minimal polinomunun derecesi serinin doğrusal karmaşıklığıdır. Doğrusal karmaşıklık seriyi üreten en kısa uzunlıktaki LFSR'a karşılık gelmektedir.

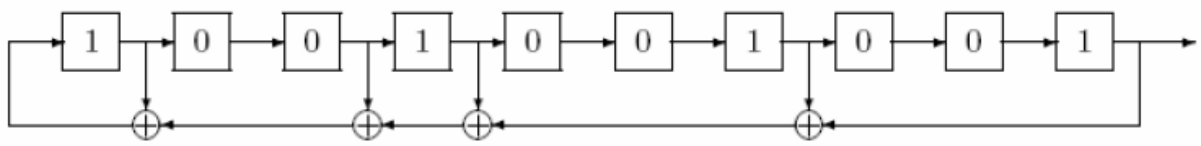
Bir s = (s_t) t≥0 serisinin minimal polinomunun doğrusal karmaşıklıgı *λ(s) s*'in en az iki *2λ(s)* sıralı bit bilgisi ile Berlekamp –Massey algoritması kullanılarak elde edilebilir.

Örnek 5.3

Geri besleme polinomu P(X) = $X^{10}+ X^7 + X^4 + X^3 + X +1$ ve başlangıç durumu s_0, ... s_9 = 1001001001 olan Şekil 5.6 de gösterilen LFSR'ı düşünelim. Bu LFSR tarafından üretilen üreteç fonksiyonu aşağıda verilmiştir.

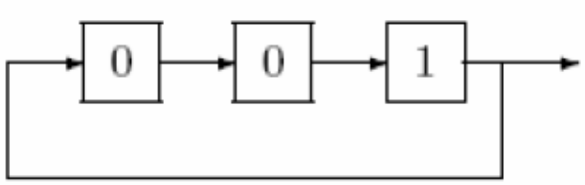
$$\sum_{t\geq 0} s_t X^t = \frac{V(X)}{P(X)}$$

V , P 'nin katsayıları ve başlangıç durumundan faydalanılarak bulunabilir. Sonuç olarak V(X) = X^7 + X +1 olarak bulunur.



Şekil 5.7. 10 Bit Uzunluğa Sahip LFSR

Bundan dolayı $\sum_{t\geq 0} s_t X^t = \frac{X^7 + X + 1}{X^{10} + X^7 + X^4 + X^3 + X + 1} = \frac{1}{X^3 + 1}$ olarak bulunur. Bu da göstermektedir ki (s_t) t≥0, Şekil 5.6 da gösterilen P_0 $(X) = X^3 + 1$ geri beslemeli bir polinom tarafından da üretilecektir. Serinin minimal polinomu $X^3 + 1$ ve karmaşıklığı da 3'e eşittir. Örnek 5.4, V(X) polinomunun X^7 teriminin katsayısının bulunuşunu göstermektedir.



Şekil 5.8. 10 Bit Uzunluğundaki LFSR ile Aynı Seriyi Üreten 3 Bit Uzunluğundaki LFSR

Doğrusal tekrarlayan dizinin minimal polinomu o dizinin doğrusal karmaşıklığını ve en düşük periyodunu bulmada çok önemli rol oynar. Gerçekte doğrusal tekrarlayan dizinin en düşük periyodu onun minimal polinomunun periyoduna eşittir. $F_q[X]$'te bir *P* polinomunun periyodu, *P(0)* ≠ 0 olmak üzere, X^e-1'i bölen P(X) polinomu için en küçük *e* değeridir. Dolayısıyla s serisinin minimal polinomu asal bir polinom ise s serisinin maksimum periyodu $q^{\lambda(s)} -1$ (ikili seriler için $2^{\lambda(s)} -1$) dir.

L uzunluğunda bir LFSR tarafından üretilen herhangi bir s = (s_t) t≥0 serisi asal bir geri besleme polinomuna sahipse serinin olası en yüksek doğrusal karmaşıklığı λ(s) = L ve q^L-1 olası en büyük periyot değeridir. Bu tür serilere en uzun (maximum-length) seriler denir. Bir LFSR'ın geri besleme polinomu daima asal polinom olmalıdır.

En uzun LFSR'lar ile üretilen seriler anahtar dizisi üreteçleri tasarlama da iyi istatistiksel özellikler ortaya koyar. En azından Golomb'un önerilerini yerine getirirler.

Tanım 4.2: N periyotlu bir *s* dizisini düşünelim. Golomb'un rastlantısallık (randomness) ile ilgili öneri aşağıdaki gibidir.

G1: Her periyot boyunca 0'ların sayısı ve 1'lerin sayısı olabildiğince eşit olmalıdır.

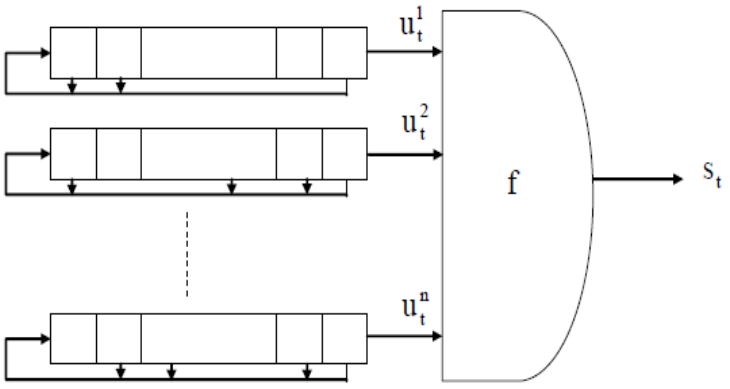
G2: Run değerlerinin sayısının yarısı 1 uzunluğunda, $\frac{1}{4}$ 'ü 2 uzunluğunda, $\frac{1}{8}$'i 3 uzunluğunda vb. olmalıdır. Bunun ötesinde bu uzunlukların her biri için eşit sayıda boşluklar (gap) ve bloklar içermelidir.

G3: Otokorelasyon fonksiyonu r_f (*d*) *K* bir tam sayı olmak üzere iki değerlidir. *N.* r_f değeri (5.8) ifadesinde gösterilmiştir.

$$\text{N. } r_f(d) = \sum_{i=0}^{N-1} 2s_i - 1)\,(2s_{i+d} - 1) = \begin{cases} N, d = 0 \\ K, 1 \le d \le N-1 \end{cases} \qquad (5.8)$$

Örnek 5.5, bir asal geri besleme polinomuna sahip bir LFSR tarafından üretilen serinin Golomb'un önerilerini yerine getirdiğini göstermektedir.

Seri kullanılmadan önce LFSR tarafından ortaya konan doğrusallık özelliğini yok etmemiz ve doğrusal karmaşıklık değerini arttırmamız gerekmektedir. Bunun için klasik yaklaşımlardan biri olan birden fazla LFSR kullanarak Şekil 5.9 da görüldüğü gibi bunları bir boolean fonksiyonu ile birleştirme yolu örnek olarak verilebilir.



Şekil 5.9 LFSR Serilerinde Doğrusallığı Yok Etmek İçin Kullanılan Örnek Bir Doğrusal Olmayan Birleştirici

LFSR tabanlı akış şifreleri;

- Doğrusal Olmayan Bileşim Üreteçleri (Nonlinear Combination Generators),
- Doğrusal Olmayan Filtre Üreteçleri (Nonlinear Filter Generators)
- Saat Kontrollü Üreteçler (Clock-Controlled Generators)

olarak üç genel kategori altında toplanabilir. Bununla beraber çeşitli tasarım teknikleri bir arada kullanılabileceği için kesin bir sınıflandırma yapmakta mümkün değildir.

5.1.2 Doğrusal Olmayan Bileşim Üreteçleri

Şekil 5.9 de görülen doğrusal olmayan bileşim üreteci birden fazla LFSR ile bu LFSR'ları bir boolean fonksiyonu ile birleştirme yoluyla elde edilir. f , n değişkenli bir fonksiyondur ve n LFSR'ın F_q üzerine oluşturduğu bir bileşim (combination) üreteç için $f : F_q^n \rightarrow F_q$ şeklindedir. Buna ek olarak f boolean fonksiyonu uniform çıkış dağılımı elde edilebilmesi için dengeli olmak zorundadır. Çıkış serilerinin iyi istatistiksel özellikler gösterebilmesi için arka arkaya gelen LFSR'ların asal (primitive) geri besleme polinomuna sahip olması gerekmektedir. Genellikle birleştirici fonksiyon ve LFSR'ların özellikleri herkes tarafından bilinir. Gizli paremetreler LFSR'ların başlangıç durumlarıdır ve bir anahtar yükleme algoritması yardımıyla şifrenin gizli anahtarından elde edilir. Dolayısıyla saldırıların çoğu bir üreteç tarafından üretilen serinin bazı bitlerinin bilgisinden yola çıkılarak LFSR'ların başlangıç durumlarını elde etmeyi amaç edinir (Bilinen Açık metin Saldırısı). Eğer LFSR'ların geri besleme polinomları ve birleştirici fonksiyonu bilinmiyorsa tekrar inşa saldırısı (reconstruction attack) geniş bir şifreli metin segment bilgisinden yola çıkarak üretecin toplam tanımının elde edilmesini sağlar.

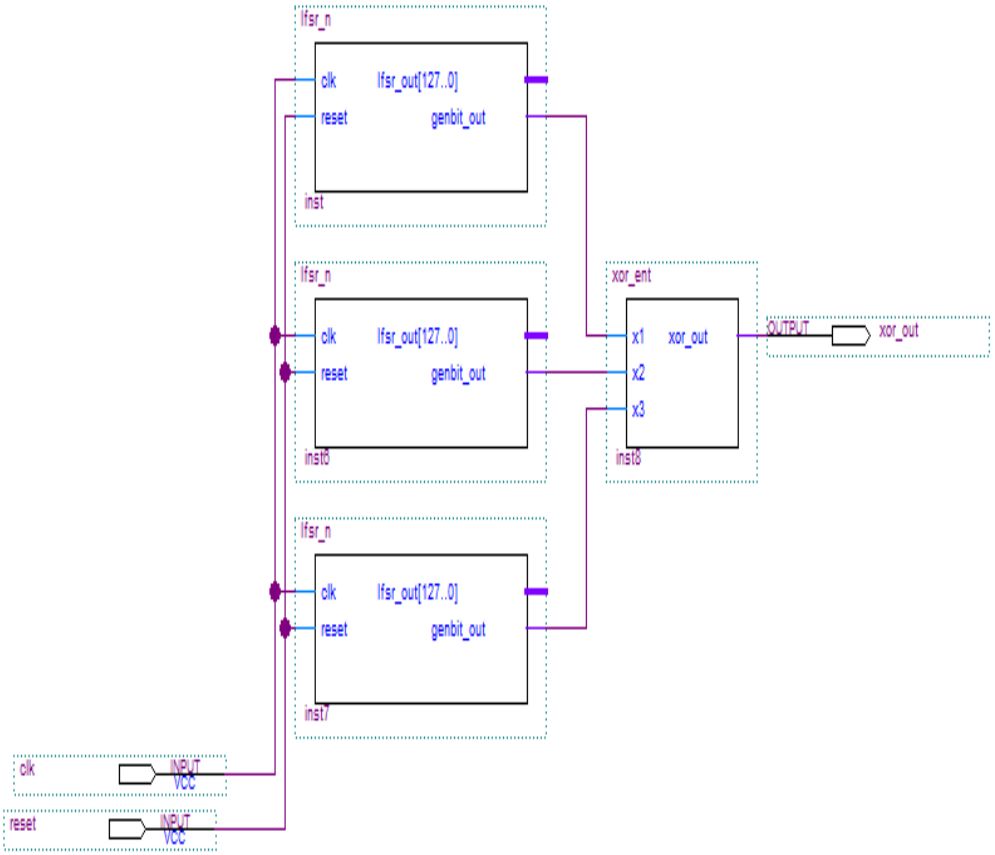
Birleştirici fonksiyon tarafından üretilen seri doğrusal tekrarlayan seridir. Periyodu ve karmaşıklığı seriyi üreten sıralı LFSR'lar ile birleştirici fonksiyonun cebirsel gösterim biçiminden (ANF) yararlanılarak elde edilebilir. F_q^n üzerine iki doğrusal tekrarlayan seri u ve v olsun. Yine bu serilerin karmaşıklıkları $\lambda(u)$ ve $\lambda(v)$ ise o zaman $u+v=(u_t + v_t)_{t\geq 0}$ dizisinin karmaşıklığı $\lambda(u + v) \leq \lambda(u) + \lambda(v)$ ' dir. Eğer u ve v 'nin minimal polinomları birbirlerine göre asal ise $\lambda(u + v) = \lambda(u) + \lambda(v)$ ' dir. Aynı şekilde uv serisi $uv = (u_t\, v_t)_{t\geq 0}$ olmak üzere uv serisinin karmaşıklığı $\lambda(uv) \leq \lambda(u)\, \lambda(v)$ ' dir. Eğer u ve v 'nin minimal polinomları birbirlerine göre asal ise $\lambda(uv) = \lambda(u)\lambda(v)$ ' dir. Böylece f boolean fonksiyonu ile birleştirilmiş asal geri beslemeli n tane ikili LFSR'dan oluşan bir bileşim üreteci tarafından üretilen bir anahtar dizisi (keystream) serisi ispatlı takip eden özelliği sağlar.

- Eğer tüm LFSR uzunlukları L_1, L_2 ……..,L_n birbirlerinden farklı ve 2 den büyükse çıkış serisinin doğrusal karmaşıklığı $f\,(L_1, L_2$ …….. $L_n)$ ’ e eşittir. Burada cebirsel gösterim biçimi (ANF) kullanılır ve bu gösterim biçimi tamsayılar ile değerlendirilir.

5.1.2.1. Doğrusal Olmayan Bileşim Üretecinin FPGA Ortamında Gerçekleştirilmesi

LFSR tabanlı doğrusal olmayan bileşim üreteci VHDL dili ile davranışsal olarak modellenmiş ve Cyclone IV FPGA board’ında gerçekleştirilmiştir. Gerçekleştirilen sistemlerde 128 bitlik LFSR yapıları kullanılmıştır. Şekil.5.10’da doğrusal olmayan bir bileşim üretecinin şematik gerçekleştirimini gerçek zamanda elde edilerek gösterilrmektedir.

Doğrusal olmayan bileşim üreteçleri tarafından üretilen sayıların rasgeleliğini kontrol etmek için üretilen sayılar NIST test süitine tabi tutulmuştur. Tablo.5.1 NIST test süitine göre elde edilen elde edilen *P*-Değer sonuç değerlerini göstermektedir. *P*-Değer ifadeleri ve NIST’teki her bir testin detaylı açıklamaları bir önceki bölümde anlatılmıştır.



Şekil 5.10. Doğrusal Olmayan Bir Bileşim Üretecinin FPGA Ortamında Gerçekleştirimi

Tablo.5.1 Doğrusal Olmayan Bileşim Üreteci İçin Test sonuçları

Testler	Doğrusal Olmayan Bileşim Üreteçi *P*-Değer Sonuçları
Frekans Testi	0.870
Blok Frekans Testi	0.979
Akış Testi	0.625
Bloktaki En Uzun Birler Testi	0.791
İkili Matris Rankı Testi	0.298
Ayrık Fourier Dönüşüm (Spectral)	0.843
Örtüşmeyen Şablon Eşleştirme Testi	0.065
Örtüşen Şablon Eşleştirme Testi Test	0.755
Maurer's "Universal Statistical" Testi	0.376
Doğrusal Karmaşıklık Testi	0.708
Seri Testi	0.821
Yaklaşık Entropi Testi	0.546
Kümülatif Toplamlar (Cusums) Testi	0.799
Rasgele Gezinim Testi	--
Rasgele Gezinim Değişken Testi	--

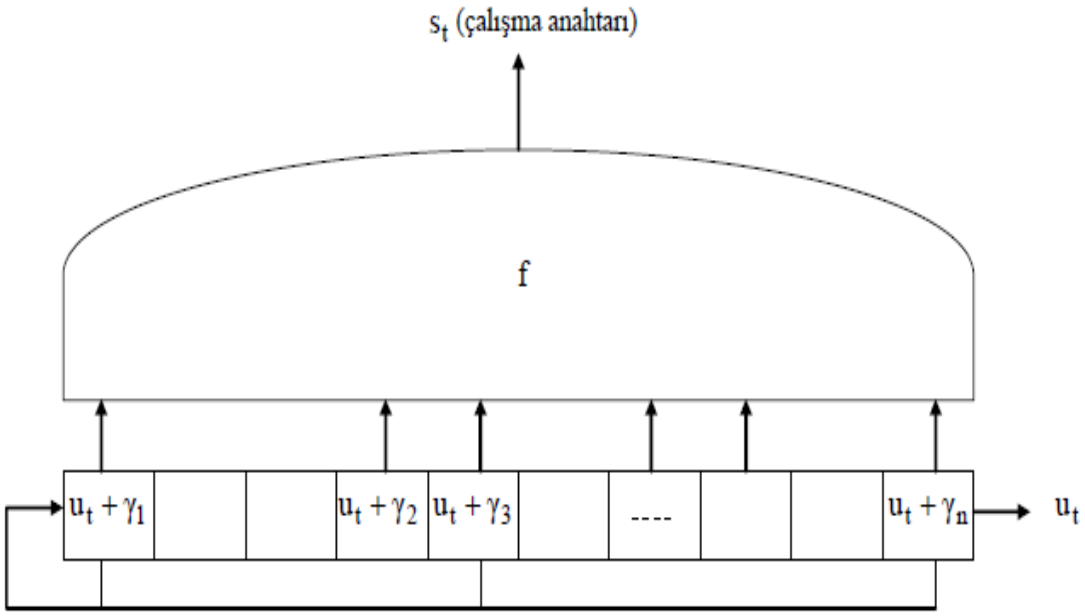
5.1.3. Doğrusal Olmayan Filtre Üreteçleri

Şekil 5.11'de görüldüğü gibi doğrusal olmayan filtre üreteçleri tek bir LFSR kullanırlar ve bu LFSR boolean fonksiyonuna girişler sağlar. Çıkış dizisi s_t = f ($u_{t+\gamma 1}$, $u_{t+\gamma 2}$,...., $u_{t+\gamma n}$), $\forall$ $t \geq 0$ olmak üzere n değeri LFSR uzunluğuna eşit ya da küçük ve f ise n değişkenli bir fonksiyondur. Buna ek olarak γ_i , $1 \leq i \leq n$ olmak üzere tapping serisi olarak isimlendirilir ve negatif olmayan tamsayıların azalan bir sırasıdır. İyi istatistiksel özelliklere sahip bir anahtar dizi serisi elde edebilmek için filtre fonksiyonu f dengeli olmalı ve LFSR geri besleme polinomu asal olarak seçilmelidir. Bir filtre üretecinde, LFSR geri besleme polinomu, tapping sırası herkes tarafından bilinir. Gizli parametre bir anahtar yükleme algoritması yoluyla şifrenin gizli anahtarı kullanılarak elde edilen LFSR'ın başlangıç durumudur. Buna ek olarak herhangi bir filtre üreteci, özel bir birleştirici üretece denktir. Doğrusal olmayan filtre yaklaşımı, F_{2^w} genişletilmiş cisimini kullanan ve yazılım yoluyla tasarlanan akış şifrelerinde tasarım için etkin bir yoldur. Bunun nedeni olarak F_{2^w} üzerine tanımlanan maksimum uzunluklu LFSR'ların ötelenmesinin yazılımda oldukça maliyetli olması gösterilebilir.

Bir filtre üretecin *s* çıkış sırası doğrusal tekrarlayan bir seridir ve karmaşıklığı *λ(s)* LFSR uzunluğuna ve filtre fonksiyonu *f* 'in cebirsel derecesine bağlıdır. Geri besleme fonksiyonu ile ikili bir LFSR'ın karmaşıklığı, LFSR'ın uzunluğu *L* ve *f* 'in cebirsel derecesi *d* olmak üzere (5.9) ifadesindeki gibi verilebilir.

$$\lambda(\mathrm{s}) \leq \sum_{i=0}^{d} \binom{L}{i} \tag{5.9}$$

Üretilen seri *s* 'in periyodu $2^{\mathrm{L}} - 1$'i böler ve L büyük bir asalsa $\lambda(\mathrm{s})$ çoğu filtreleme fonksiyonları için en azından $\binom{\mathrm{L}}{\mathrm{d}}$ değerine eşittir. Yüksek doğrusal karmaşıklığı elde edebilmek için LFSR uzunluğu *L* ve filtreleme fonksiyonunun derecesi yeterince büyük olmalıdır. Daha kesin bir ifadeyle saldırganın elindeki anahtar dizisi uzunluğu $\binom{L}{deg(f)}$ değerinden küçük olmalıdır.



Şekil 5.11. Doğrusal Olmayan Bir Filtre Üretecinin Genel Yapısı

Doğrusal olmayan filtre üreteçleri tarafından üretilen sayıların rasgeleliğini kontrol etmek için üretilen sayılar NIST test süitine tabi tutulmuştur. Tablo.5.2 NIST test süitine göre elde edilen elde edilen *P*-Değer sonuç değerlerini göstermektedir.

Tablo.5.2. Doğrusal Olmayan Filtre Üreteci İçin Test sonuçları

Testler	Doğrusal Olmayan Filtre Üreteçi *P*-Değer Sonuçları
Frekans Testi	0.287
Blok Frekans Testi	0.376
Akış Testi	0.310
Bloktaki En Uzun Birler Testi	0.019
İkili Matris Rankı Testi	0.405
Ayrık Fourier Dönüşüm (Spectral)	0.517
Örtüşmeyen Şablon Eşleştirme Testi	0.999
Örtüşen Şablon Eşleştirme Testi Test	0.321
Maurer's "Universal Statistical" Testi	0.219
Doğrusal Karmaşıklık Testi	0.255
Seri Testi	0.231
Yaklaşık Entropi Testi	0.149
Kümülatif Toplamlar (Cusums) Testi	0.420
Rasgele Gezinim Testi	--
Rasgele Gezinim Değişken Testi	--

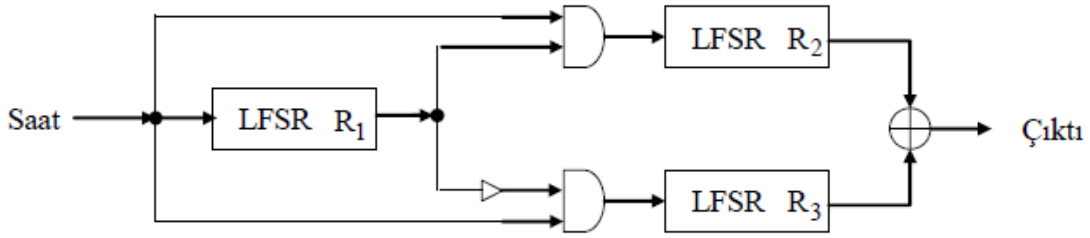
5.1.4. Saat Kontrollü Üreteçler

Doğrusal olmayan bileşim üreteçleri ve filtre üreteçlerinde kullanılan saat tetiklemesi, o anki durumun 1 adım ileri gitmesini ve anahtar dizisinin bitinin ya da bitlerinin her tetikleme de üretilmesini sağlamaktadır. Diğer bir deyişle saat düzenli bir şekilde tetiklenir. Saat kontrollü üreteçlerde ise ana fikir düzensiz sinyaller kullanarak LFSR'ların saat tetiklemelerinin sayısını kontrol etmektir. Düzensiz sinyaller, başka bir LFSR'ın çıkışı ya da şifrenin içsel bir değişkeni olabilir. Düzensiz olarak kontrol edilen LFSR'ın çıkışındaki doğrusallık yok edilerek düzenli olarak tetiklenen üreteçlere bu özelliğinden dolayı yapılan saldırıların önüne geçilebilir. Bu tür şifrelerdeki tasarım felsefesinde saat vuruşlarının sayısını düzensiz sinyaller kullanarak kontrol etme fikri vardır. Saat besleme sinyali bir doğrusal geri beslemeli öteleyici saklayıcı olabileceği gibi şifrenin diğer içsel bir yapısı da olabilir. Yani bu metotla doğrusal geri beslemeli saklayıcıların çıkışında ki doğrusallığın yok edilmesi amaç edinilir. 2 ayrı şekilde gerçekleştirilebilir. Bunlar;

- Alternatifli Adım Üreteci (The Alternating Step Generator)
- Büzülen Üreteç (The Shrinking Generator)

5.1.4.1. Alternatifli Adım Üreteci

Alternating Step Generator (ASG) Shrinking Generator'den daha komplike bir yapıya sahiptir. Bu yapı toplam 3 LFSR içermekte olup LFSR'lerin ilki diğer iki LFSR donanımını kontrol eder. LFSR R_1 kontrol eden LFSR R_2 ve R_3 kontrol edilen olarak adlandırılır. R_1 tarafından üretilen dizi $s = s_0, s_1, \ldots$ şeklinde, R_2 ve R_3 tarafından üretilen diziler ise sırasıyla $a = a_0, a_1, \ldots,\ b = b_0, b_1, \ldots$ve üretilen çıkış dizisi $z = z_0, z_1, \ldots$olsun. s_i çıkış dizisini elde etmek için R_1 tarafından üretilen s_i= 1 veya s_i= 0 durumları için sırasıyla R_2 ve R_3 nin aktif edilmesi gereklidir. O zaman z_i , i. adımda LFSR R_2 ve LFSR R_3 tarafından üretilen bitlerin modulo 2' sine eşit olacaktır. Bir başka ifadeyle LFSR R_2 ve LFSR R_3' nin XOR işlemine tabi tutulmasıyla elde edilen değer olacaktır. Şekil.5.12 alternatifli adım üretecinin yapısını göstermektedir.



Şekil.5.12. Alternetifli Adım Üreteci

Alternatifli adım üretecinin çalışma prensibi aşağıdaki gibi açıklanabilir:

- R_1 saklayıcısı sistem saati tarafından tetiklenir.
- R_1saklayıcısının çıkışı 1 ise, R_2 tetiklenir ve R_3 tetiklenmeden bir önceki bit değeri tekrar edilir. (İlk çevrimde (cycle) R_3 saklayıcısının ilk çıkış biti 0 alınır.)
- R_1 saklayıcısının çıkışı 0 ise, R_3 tetiklenir ve R_2 tetiklenmeden bir önceki bit değeri tekrar edilir. (İlk çevrimde (cycle) R_2 saklayıcısının ilk çıkış biti 0 alınır.)
- R_2 ve R_3 saklayıcılarının toplamı anahtar dizisi çıkışı olarak elde edilir.

R_1 saklayıcısının 2^{L_1} periyotlu bir Bruijn serisi ürettiğini düşünürsek ve R_2 ile R_3 saklayıcıları maksimum uzunluklu LFSR olmak üzere, OBEB(L_2 , L_3) =1, aralarında asal ise

- serinin periyodu 2^{L_1} $(2^{L_2} - 1)(2^{L_3} - 1)$ ' dir.
- s serisinin doğrusal karmaşıklığı λ(s) aşağıdaki ifadeyi sağlar.

$$(L_2 + L_3)\, 2^{L_1 - 1} < \lambda(s) < (L_2 + L_3)\, 2^{L_1}$$

- serisinde örüntülerin (patern) dağılımı hemen hemen düzgündür.

Alternatifli adım üretecinde kullanılan LFSR'lar R_1, R_2 , R_3 maksimum uzunluklu LFSR'lar olmalı ve LFSR'ların uzunlukları L_1, L_2 , L_3 birbirlerine göre asal olmalıdır. (OBEB (L_1, L_2)=1, OBEB(L_2 , L_3) =1, OBEB(L_1, L_3) =1) . Bunun ötesinde uzunlukları birbirlerine yakın olmalıdır. Yani, eğer L_1 = l ise $L_2 \approx$ l ve $L_3 \approx$ l olmalıdır. Alternatifli adım üretecine en iyi saldırı böl ve fethet (divide and conquer) saldırısıdır. Bu saldırı R_1 saklayıcısına 2l adım ile mümkündür. Eğer l = 128 ise üreteç bütün bilinen saldırılara karşı güvenlidir.

5.1.4.2. Büzülen Üreteç

Büzülen Üreteç (Shrinking generator(SG)) 1993 yılında Coppersmith ve arkadaşları tarafından geliştirilen iyi bilinen Sözde Rasgele Sayı Üreteçlerin(Psuedo Random Number Generators (PRNG))dendir. Akış şifreleri gibi uygulamalarda kullanılmak üzere tasarlanmış Büzülen üreteç, kavramsal yapısı yüzünden büyük ilgi görmüştür. SG Şekil.5.13'de gösterildiği gibi iki LFSR ve basit bir karşılaştırma devresinden oluşmaktadır. Bu yapısından dolayı LFSR'den daha iyi kriptografik özelliklere sahiptir. LFSR A a = a_0, a_1, … dizisini, LFSR B b = b_o, b_1, … dizisini üretsin. SG'nin çıkış dizisi ise z = z_0, z_1, … aşağıdaki denklemle elde edilir**.**

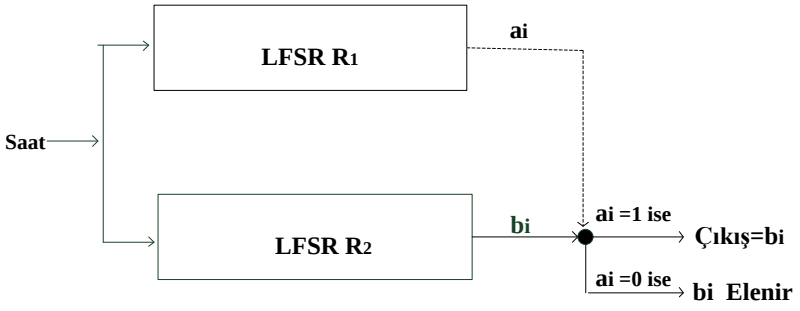
$$z(t) = \begin{cases} a(i_t) & if\ b_t = 1 \\ a(i_t - 1) & if\ b_t = 0 \end{cases} \tag{5.10}$$

Örneğin LFSR A ve B tarafından üretilen dizi aşağıdaki gibi olsun.

a=<u>1</u>, 0, 0, 1, 0, <u>1</u>, 1, 1, <u>0</u>, 0, <u>1</u>, <u>0</u>, 1, 1

b= 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0

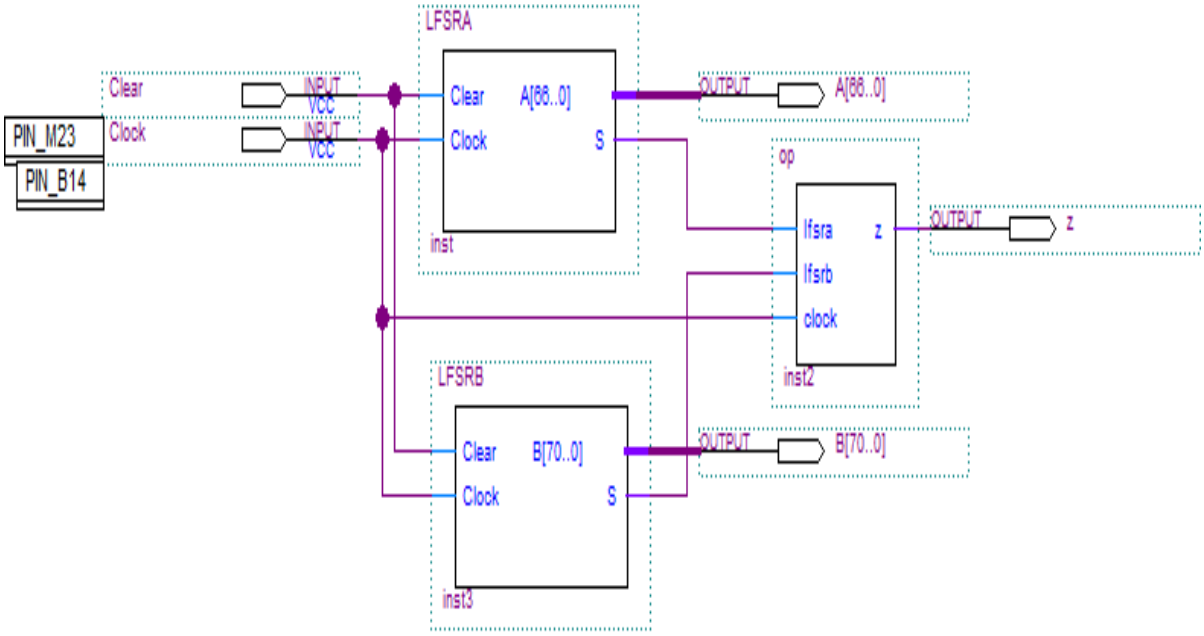
Buna göre SG tarafından üretilen dizi z= 1, 1, 0, 1, 0…. şeklinde olacaktır. Eğer LFSR-A ve LFSR-B L_1 ve L_2 bit ise,SG'nin periyodu $(2^{L2} - 1) \cdot 2^{L1-1}$ olacaktır. [12]'ye göre, bütün uzunluklar en az 64 bit olmak zorundadır. Bundan dolayı gcd(L_1, L_2) 1' e eşit olmak zorundadır.



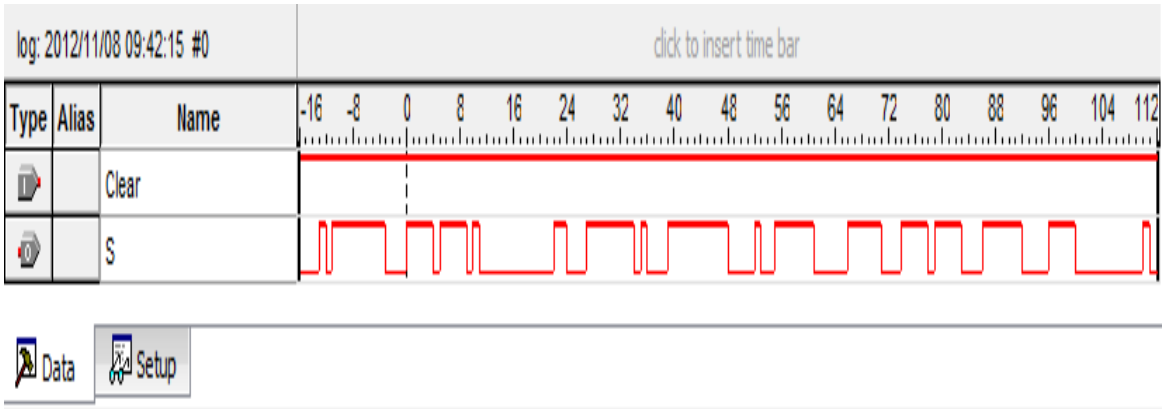
Şekil.5.13. Büzülen Üreteç

SG ve ASG PRNG'ler VHDL dili ile davranışsal olarak modellenmiş ve Cyclone IV FPGA board'ında gerçekleştirilmiştir. Gerçekleştirilen sistemlerde 67, 71, 131, 137 ve 141 bitlik LFSR yapıları kullanılmıştır.

Şekil.5.14'de SG'nin şematik gerçekleştirimini Şekil.5.15'de ise gerçek zamanda elde edilen rasgele bitlerin değişimini göstermektedir.



Şekil.5.14. Büzülen Üreteçin FPGA ortamında Şematik Olarak Gerçekleştirilmesi



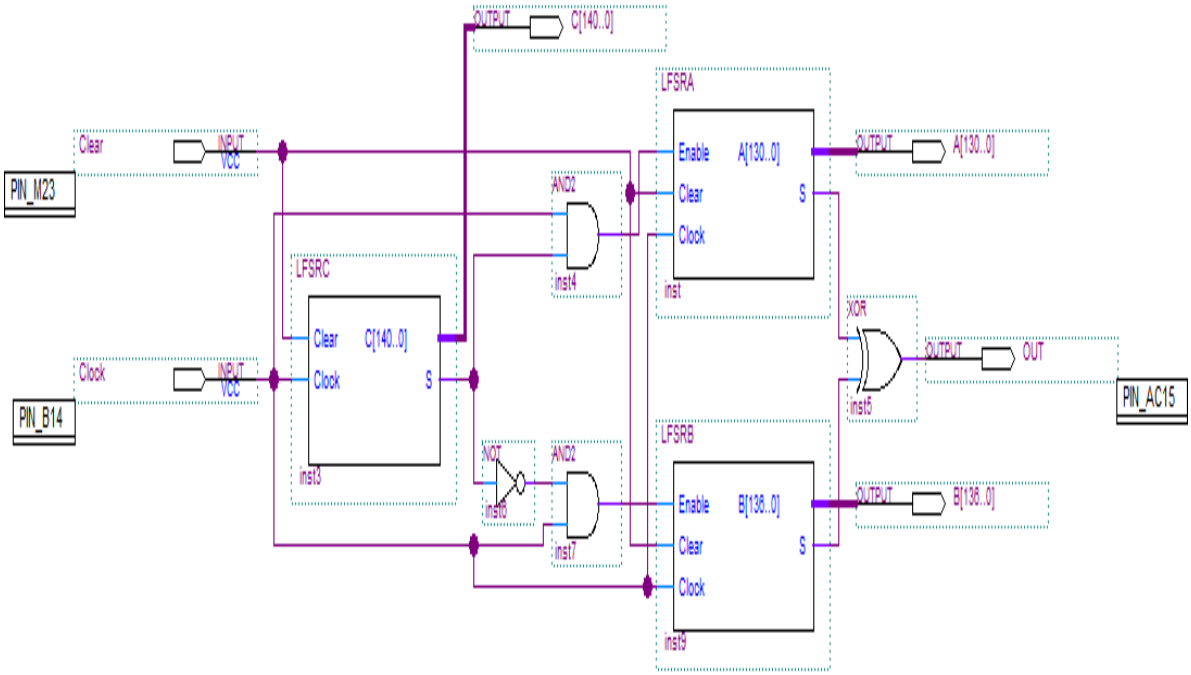
Şekil.5.15. Büzülen Üreteçten Elde Edilen Rasgele Bitlerin Değişimi

Saat kontrollü bir üreteç olan büzülen üreteç algoritması tarafından üretilen sayıların rasgeleliğini kontrol etmek için üretilen sayılar NIST test süitine tabi tutulmuştur. Tablo.5.3 NIST test süitine göre elde edilen elde edilen *P*-Değer sonuç değerlerini göstermektedir.

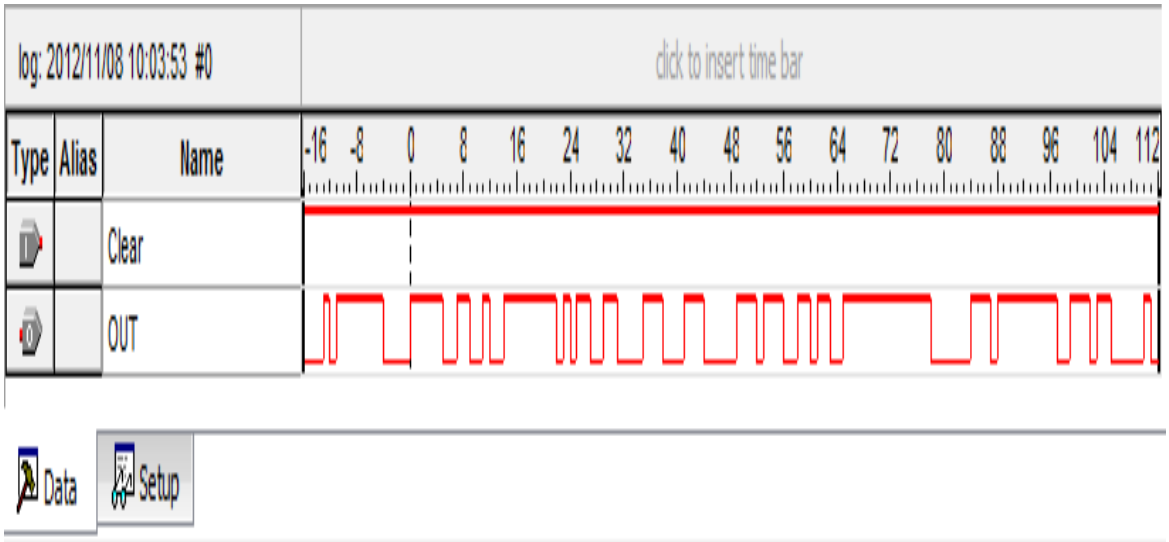
Tablo.5.3 Büzülen Üreteç İçin Test Sonuçları

Testler	Büzülen Üreteç İçin *P*-Değer
Frekans Testi	0.555
Blok Frekans Testi	0.247
Akış Testi	0.216
Bloktaki En Uzun Birler Testi	0.716
İkili Matris Rankı Testi	0.179
Ayrık Fourier Dönüşüm (Spectral)	0.886
Örtüşmeyen Şablon Eşleştirme Testi	0.292
Örtüşen Şablon Eşleştirme Testi Test	0.494
Maurer's "Universal Statistical" Testi	0.112
Doğrusal Karmaşıklık Testi	0.342
Seri Testi	0.389
Yaklaşık Entropi Testi	0.317
Kümülatif Toplamlar (Cusums) Testi	0.668
Rasgele Gezinim Testi	-
Rasgele Gezinim Değişken Testi	-

Şekil.5.16'de Alternatif Adımlı Üreteç(Alternating Step Generators)'in şematik gerçekleştirimini Şekil. 5.17'de ise gerçek zamanda elde edilen rasgele bitlerin değişimini göstermektedir.



Şekil.5.16. Alternatif Adımlı Üreteçin FPGA Ortamında Gerçekleştirilmesi



Şekil.5.17. Alternatifli Adım Üreteçten Elde Edilen Rasgele Bitlerin Değişimi

Saat kontrollü bir diğer üreteç olan alternatifli adım üreteç algoritması tarafından üretilen sayıların rasgeleliğini kontrol etmek için üretilen sayılar NIST test süitine tabi tutulmuştur. Tablo.5.4 NIST test süitine göre elde edilen P-Değer sonuçları gösterilmektedir.

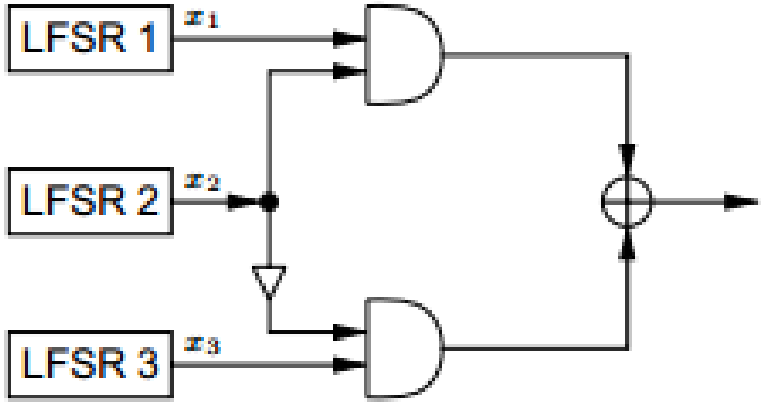
Tablo.5.4 Alternatifli Adım Üreteç İçin Test Sonuçları

Testler	Alternatifli Adım Üreteci *P*-Değer Sonuçları
Frekans Testi	0.21
Blok Frekans Testi	0.396
Akış Testi	0.558
Bloktaki En Uzun Birler Testi	0.844
İkili Matris Rankı Testi	0.641
Ayrık Fourier Dönüşüm (Spectral)	0.713
Örtüşmeyen Şablon Eşleştirme Testi	0.547
Örtüşen Şablon Eşleştirme Testi Test	0.481
Maurer's "Universal Statistical" Testi	0.645
Doğrusal Karmaşıklık Testi	0.353
Seri Testi	0.645
Yaklaşık Entropi Testi	0.446
Kümülatif Toplamlar (Cusums) Testi	0.619
Rasgele Gezinim Testi	-
Rasgele Gezinim Değişken Testi	-

5.1.5. Geffe Üreteci

Geffe generator, Şekil.5.18' te gösterildiği gibi L_1, L_2 ve L_3 uzunluğuna sahip 3 LFSR yapısından oluşmaktadır. Sistemin doğrusal olmayan davranış sergileyebilmesi için AND ve NOT kapılarının kullanılması ile sağlanmaktadır. Geffe generatörün doğrusal olmayan çıkış fonksiyonu denklem 5.11'de gösterildiği gibi XOR kullanılması ile sağlanmıştır.

$$f(x_1, x_2, x_3) = x_1x_2 \oplus (1 + x_2)x_3 = x_1x_2 \oplus x_2x_3 \oplus x_3 \quad (5.11)$$

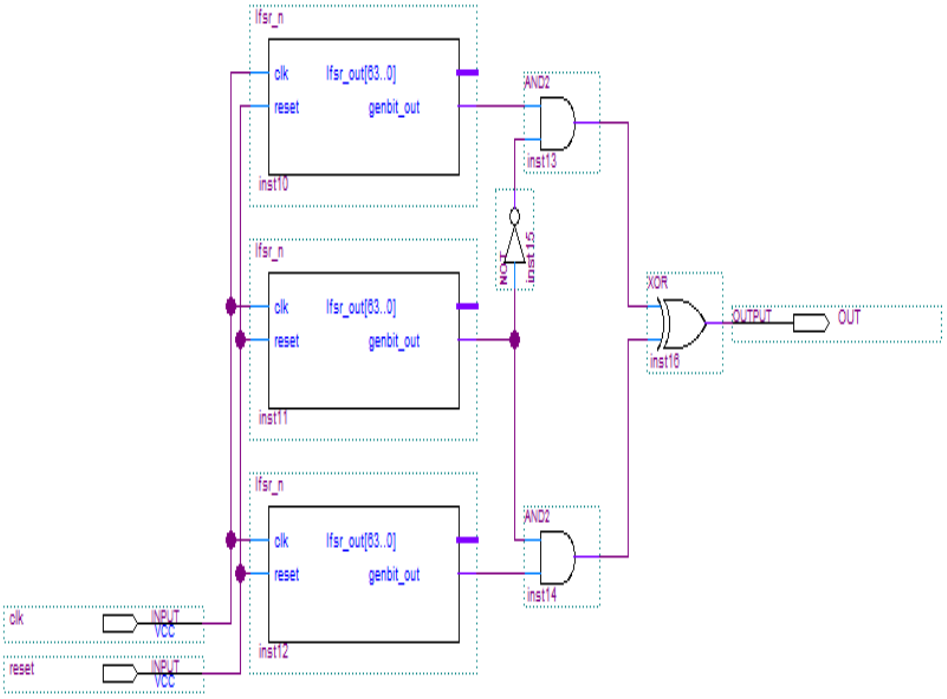


Şekil.5.18. Geffe Generatör

f fonkiyonu ile üretilen sayıların periyodu $(2^{L1}-1)(2^{L2}-1)(2^{L3}-1)$ ve lineer karmaşıklık $L=L_1L_2+L_2L_3+L_3$ ile belirlenir.

5.1.5.1. Geffe Üretecinin FPGA Ortamında Gerçekleştirilmesi

Geffe Üretecinin VHDL dili ile davranışsal olarak modellenmiş ve Cyclone IV FPGA board'ında gerçekleştirilmiştir. Gerçekleştirilen sistemlerde 64 bitlik LFSR yapıları kullanılmıştır. Şekil.5.19'de doğrusal olmayan bir bileşim üretecinin şematik gerçekleştirimini gerçek zamanda elde edilerek gösterilrmektedir.



Şekil.5.19. Geffe Üretecinin FPGA Ortamında Gerçekleştirilmesi

Geffe üreteç algoritması tarafından üretilen sayıların rasgeleliğini kontrol etmek için uygulanan NIST test sonuçları Tablo.5.5'te gösterilmektedir.

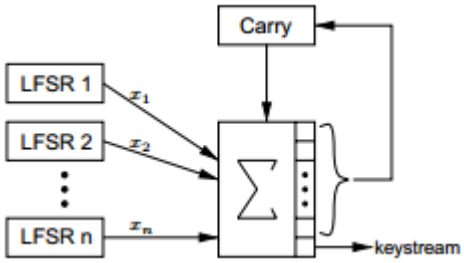
Tablo.5.5. Geffe Üreteci İçin Test Sonuçları

Testler	Geffe Üreteci *P*-Değer Sonuçları
Frekans Testi	0.640
Blok Frekans Testi	0.376
Akış Testi	0.484
Bloktaki En Uzun Birler Testi	0.617
İkili Matris Rankı Testi	0.219
Ayrık Fourier Dönüşüm (Spectral)	0.592
Örtüşmeyen Şablon Eşleştirme Testi	0.993
Örtüşen Şablon Eşleştirme Testi Test	0.321
Maurer's "Universal Statistical" Testi	0.307
Doğrusal Karmaşıklık Testi	0.502
Seri Testi	0.121
Yaklaşık Entropi Testi	0.619
Kümülatif Toplamlar (Cusums) Testi	0.691
Rasgele Gezinim Testi	--
Rasgele Gezinim Değişken Testi	--

5.1.6. Toplam Üreteç

Şekil.5.20'de gösterildiği gibi n tane farklı LFSR'nin üreteceği sayıların toplam fonksiyonu ile elde edilen doğrusal olmayan kombinasyonlu üreteçtir. Sistemin doğrusal davranış göstermemesi için toplam fonksiyonundan elde edilen taşma bitinin tekrar toplam fonksiyonuna verilmesi ile sağlanmaktadır. 2 adet LFSR'nin kullanılması durumunda elde edilen z çıkış değeri ve elde biti c, aşağıda verilen denklem.5.12' ye göre elde edilecektir.

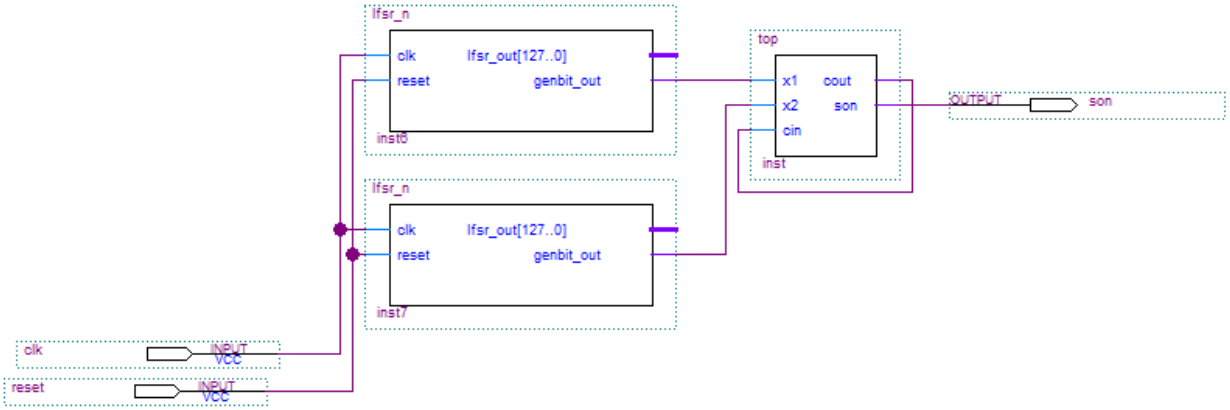
$$z_j=f_1(a_j,b_j,c_{j-1})=a_j \oplus b_j \oplus c_{j-1} \qquad 0 \leq j \leq m,$$
$$c_j=f_2(a_j,b_j,c_{j-1})=a_j b_j \oplus (a_j \oplus b_j)c_{j-1} \qquad 0 \leq j \leq m-1 \qquad (5.12)$$



Şekil.5.20. Toplam Generatör

5.1.6.1. Toplam Üreteçin FPGA Ortamında Gerçekleştirilmesi

Toplam üretecin VHDL dili ile davranışsal olarak modellenmiş ve Cyclone IV FPGA board'ında gerçekleştirilmiştir. Gerçekleştirilen sistemlerde 127 bitlik LFSR yapıları kullanılmıştır. Şekil.5.20'de doğrusal olmayan bir bileşim üretecinin şematik gerçekleştirimi gerçek zamanda elde edilerek gösterilrmektedir.



Şekil.5.21. Toplam Üretecin FPGA Ortamında Gerçekleştirilmesi

6. SONUÇLAR

Kriptografi gerçek rasgele sayılar üretme üzerinde kurulu olmasına karşılık, bilgisayar algoritmaları deterministik bir şekilde çalışır. Bu tez çalışmasında sözde rasgele sayı dizileri oluşturabilmek için literatürde bilinen akış şifreleme algoritmaları incelenmiştir. Akış şifreleme algoritmalarından LFSR tabanlı olan 5 ayrı algoritma donanımsal olarak tasarlanarak rasgele sayı üreteci şeklinde kullanılmıştır. Bu algoritmalar, Dorusal olmayan bileşim üreteci, doğrusal olmayan filtre üreteci, alternatif adımlı üreteç, büzülen üreteç ve Geffe üreteci donanımsal olarak FPGA ortamında gerçekleştirilmiştir. Bu tasarımlar rasgele sayı üretecin kriptolojide uygulanabilirliğini kontrol edilmiştir. Ayrıca bu kontrol edilebilirliği göstermek için elde edilen rasgele sayılar NIST testlerine tabii tutulmuştur.

Rasgele sayılar ister yazılım ister donanımsal olarak üretilsin bu sayıların rasgele olduklarını göstermek için literatürde bilinen testlerden geçme zorunluluğu vardır. Özellikle rasgele sayıların kriptografik uygulamalarda kullanılması için bu testlerin uygulanması gereklidir. Akış şifreleme algoritmalarının kullanılarak rasgele sayıların üretilmesine ilaveten bu çalışmada ayrıca NIST Test Suiti de incelenmiş ve test sonuçları ayrı ayrı değerlendirilmiştir.

KAYNAKLAR

[1] http://cacr.uwaterloo.ca/hac/about/chap6.pdf , 2014

[2] Lincoln D. S., "Web Security: A Step-by-Step Reference Guide", Addison Wesley Professional , Boston,32-48, 60-82 (1997).

[3] Dworkin M. "Computer Security: Recommendation for Block Cipher Modes of Operation, Methods and Techniques" NIST Special Publication, Gaithersburg, 800-838 (2001).

[4] Juan C. Cerda, Chris D. Martinez, Jonathan M. Comer, and David H. K. Hoe, "An Efficient FPGA Random Number Generator using LFSRs and Cellular Automata", IEEE 55th International Midwest Symposium on Circuits and Systems (MWSCAS) p:912-915,Tyler USA, 2012

[5] Jonathan M. Comer, Juan C. Cerda, Chris D. Martinez, and David H. K. Hoe, "Random Number Generators using Cellular Automata Implemented on FPGAs", 44th IEEE Southeastern Symposium on System Theory, p:67-72, Jacksonville,USA,2012.

[6] Büyüksaraçoğlu F., Buluş E., Sözde Rastsal Sayı Üretiminin Kriptografik Açıdan İncelenmesi, İletişim Teknolojileri Ulusal Sempozyumu (İTUSEM), Türkiye, 2009.

[7] M. S. Azzaz, C. Tanougast, S. Sadoudi, R. Fellah, A. Dandache, "A new auto-switched chaotic system and its FPGA implementation", Commun Nonlinear Sci Numer Simulat 18, 1792–1804,2013.

[8] . L. Dejuna, P. Zhena, "Research of True Random Number Generator Based on PLL at FPGA", Procedia Engineering 29, 2432 – 2437, 2012.

[9] J. R. Doyle, C. H. Chen, "Patterns in stock market movements tested as random number generators", European Journal of Operational Research 227 p:122–132, 2013.

[10] Q. Zhou, X. Liao, K. Wong, Y. Hu, D. Xiao, " True random number generator based on mouse movement and chaotic hash function", Information Sciences p:3442–3450, 2009.

[11] Sewak K, Rajput P, Panda Amit K, "FPGA Implementation of 16 bit BBS and LFSR PN Sequence Generator: A Comparative Study", In Proce. of the IEEE Student Conference on Electrical, Electronics and Computer Sciences 2012, 1-2 Mar 2012, NIT Bhopal, India.

[12] Goresky, M. and Klapper, A.M. Fibonacci and Galois representations of feedback-with-carry shift registers, IEEE Transactions on Information Theory, Nov 2002, Volume: 48, On page(s): 2826 – 2836.

[13] Panda Amit K, Rajput P, Shukla B, "Design of Multi Bit LFSR PNRG and Performance comparison on FPGA using VHDL", International Journal of Advances in Engineering & Technology (IJAET), Mar 2012, Vol. 3, Issue 1, pp. 566-571

[14] http://en.wikipedia.org/wiki/Stream_cipher, 2015.

[15] Kriptolojiye Giriş Ders Notları, Uygulamalı Matematik Enstitüsü, Kriptografi Bölümü ODTÜ, Türkiye Şubat 2004.

[16] Büyüksaraçoğlu F., Akış Şifrelerin Tasarım Teknikleri ve Güç Analizi Doktora Semineri, Trakya Üniversitesi Fen Bilimleri Enstitüsü, 2011.

[17] A. Menezes, P.v. Oorschot and S.Vanstone, Handbook of Applied Crytography, CRC Press 1997.

[18] Anne Canteaut, Stream Ciphers, Encyclopedia of Crytography and Security, 2005, available at:http://www-rocq.inria.fr/codes/Anne.Canteaaut/encyclopedia.pdf

[19]C.G. Günther, Alternating Step Generators Controlled by the Brujin Sequences, EUROCRYPT'87, Lecturer Notes in Computer Science, pp.91-103,1988.

[20] C.E. Shannon, Communication Theory of Secrecy Systems, Bell System Technical Journal, No. 30, pp. 50-64, 1949.

[21] L. Keliher, Linear Cryptanalysis of Substitution-Permutation Networks, PHd Thesis, 2003.

[22] D. R. Stinson, Cryptography: Theory and Practice, Second Edition, CRC Press, 2002.

[23] J. Lano, Cryptanalysis and Design of Synchronous Stream Ciphers, PHd Thesis, 2006.

[24] P. Ekdahl, On LFSR Based Stream Ciphers, PHd Thesis, November 2003.

[25] Anne Canteaut, Stream Ciphers, Encyclopedia of Cryptography and Security, 2005

[26] A. Menezes, P. v. Oorschot, and S. Vanstone, Handbook of Applied Cryptography, CRC Press, 1997.

[27] Revised NIST Special Publication 800-22, "A Statistical Test Suite for the Validation of Random Number Generators and Pseudo Random Number Generators for Cryptographic Applications", 2001.

[28] P. Ekdahl, "On LFSR Based Stream Ciphers", PHd Thesis, November 2003.

[29] NESSIE, New European Schemes for Signatures, Integrity, and Encryption, 2000- 2003. http://www.cryptonessie.org.

[30] ECRYPT. eSTREAM, the ECRYPT Stream Project., 2005-2008. http://www.ecrypt.eu.org/stream.

[31] Anne Canteaut, Stream Ciphers, Encyclopedia of Cryptography and Security, 2005, available at: http://www-rocq.inria.fr/codes/Anne.Canteaut/encyclopedia.pdf.

[32] Demirkol A. Ş., Kaotik Osilatör Girişli Adc Tabanlı Rastgele Sayı Üreteci, Yüksek Lisans Tezi, İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü, Türkiye, Haziran 2007.

[33] Canteaut A., Stream Ciphers, Encyclopedia of Cryptography and Security, 2005, available at: http://www-rocq.inria.fr/codes/Anne.Canteaut/encyclopedia.pdf.

[34] Sakallı M. T., Modern Şifreleme Yöntemlerinin Gücünün İncelenmesi, Doktora Tezi, Trakya Üniversitesi, Türkiye, 2006.

[35] US National Institute of Standards and Technology, Data Encryption Standard, Federal Information Processing Standards Publications, No. 46-3, 1999.

[36] Esra ERKEK and Taner TUNCER, " The Implementation of ASG and SG Random Number Generators", IEEE International Conference on System Science and Engineering, Óbuda University, Bécsi út 96/B, H-1034 Budapest, Hungary.

ÖZGEÇMİŞ

Esra ERKEK 1989 yılında Elazığ' da doğdu. İlk, orta ve lise öğrenimini Elazığ'da tamamladı. 2007 yılında Balakgazi Lisesini bitirdikten sonra 2012 yılında Fırat Üniversitesi, Mühendislik Fakültesi, Bilgisayar Mühendisliği Bölümünden mezun oldu. Aynı yıl içerisinde Fırat Üniversitesi, Fen Bilimleri Enstitüsü, Bilgisayar Mühendisliği Ana Bilim dalında yüksek lisans yapma hakkı kazandı.