KOCAELİ ÜNİVERSİTESİ * FEN BİLİMLERİ ENSTİTÜSÜ

KABLOSUZ ALGILAYICI AĞLAR İÇİN SKIPJACK ŞİFRELEME ALGORİTMASININ FPGA İLE GERÇEKLENMESİ

YÜKSEK LİSANS TEZİ

Ergin ERYILMAZ

Anabilim Dalı: Elektronik ve Bilgisayar Eğitimi

Danışman: Doç. Dr. İsmail ERTÜRK

KOCAELİ, 2009

KOCAELİ ÜNİVERSİTESİ * FEN BİLİMLERİ ENSTİTÜSÜ

KABLOSUZ ALGILAYICI AĞLAR İÇİN SKIPJACK ŞİFRELEME ALGORİTMASININ FPGA İLE GERÇEKLENMESİ

YÜKSEK LİSANS TEZİ

Ergin ERYILMAZ

Tezin Enstitüye Verildiği Tarih: 22 Mayıs 2009
Tezin Savunulduğu Tarih: 26 Haziran 2009

Tez Danışmanı	Üye	Üye
Doç.Dr. İsmail ERTÜRK	Doç. Dr. Celal ÇEKEN	Yrd. Doç. Dr. A. Turan ÖZCERİT

KOCAELİ, 2009

ÖNSÖZ ve TEŞEKKÜR

Kablosuz Algılayıcı Ağ (KAA) uygulamalarının her geçen gün daha yaygın hale gelmesi ve farklı uygulama alanlarında kullanılabilir olması, KAA üzerinde yapılan çalışmaların daha da yoğunlaşmasına yol açmıştır. Bu çalışmalar, ağırlıklı olarak KAA temel işlevlerini daha üst düzeye getirmeyi hedefleyen hem yazılımsal hem de donanımsal teknolojiler üzerine kurulmaktadır. İlerleyen teknolojinin sağladığı donanımsal ilerlemeler sonucunda, KAA'lar üzerindeki yenilikler ve gelişmeler yazılım ortamlarıyla sentezlenerek daha güvenli KAA'ların, uygulama alanlarında yer bulmalarını sağlanmıştır. Bütün bu bilgiler ışığında ileriye dönük örnek teşkil edebilecek bu tez çalışması, hem uygulama hem de yazılım olarak gerçekleştirilmektedir.

Yüksek lisans eğitimim ve tez çalışmam sürecinde, tez çalışmasının başlangıcından sonuna kadar yardımlarını hiçbir zaman esirgemeyen, bilgi ve deneyimleri ile beni her zaman destekleyen, değerli vakitlerini bana ayıran tez danışmanım Doç. Dr. İsmail ERTÜRK'e; maddi, manevi desteklerini esirgemeyen değerli hocam Dr. Sedat ATMACA'ya, arkadaşlarım Alper KARAHAN ve Süleyman ÇAKICI'ya; bilgi ve becerilerini benimle paylaşan iş arkadaşlarım Muhammet GÜZEL ve Metin ÇIKRIKÇIOĞLU'na teşekkürlerimi sunarım.

Ayrıca şu an sahip olduğum pozisyona gelmemde büyük emekleri geçen ve beni daima destekleyen annem Emine ERYILMAZ ve babam Başaran ERYILMAZ'a teşekkürlerimi sunarım.

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

TABLOLAR DİZİNİ

SEMBOLLER

bayt	: 8 bit'lik değer
G	: G-permütasyonu
kelime	: 16 bit'lik değer
MHz	: Megahertz
w	: 16 bit'lik veri değişkeni
W	: Watt
V^n	: Tüm n-bit değerlerden oluşan küme
$\oplus$	: Özel veya işlemi
||	: Veya işlemi

Alt İndisler

i	: *w* sembolü indisi [1, 4]
k	: Adım sayısı [0, 31]

Kısaltmalar

AES	: Advanced Encryption Standard
AND	: Ve işlemi
CCM	: Counter with Chiper Block Chaining
CFB	: Chipher Feedback
CMOS	: Complementary metal–oxide–semiconductor
CPLD	: Complex Programmeble Logic Device
CTR	: Counter Encrytion Mode
DES	: Data Encryption Standard
DESX	: Data Encryption Standard X
DSA	: Digital Signature Algorithm
ECC	: Elliptic Curve Cryptography
ECDH	: Elliptic Curve Diffie-Hellman
ECDSA	: The Elliptic Curve Digital Signature Algorithm
EEPROM	: Electrically Erasable Programmable Read-Only Memory
EPROM	: Erasable Programmable Read Only Memory
FIPS	: Federal Information Processing Standards
FPGA	: Field Programmable Gate Array
IDEA	: International Data Encryption Algorithm
I/O	: Input/Output
KAA	: Kablosuz Algılayıcı Ağ
LUT	: Lookup Table
OFB	: Output Feedback
PKC	: Public Key Encryption
PLD	: Programmable logic device

RIPEMD	: RACE Integrity Primitives Evaluation Message Digest
RSA	: Ron Rivest, Adi Shamir ve Len Adleman
RS-232	: Recommended Standard 232
SHA	: Secure Hash Algorithm
SKE	: Symmetric Key Encryption
SRAM	: Static Random Access Memory
UART	: Universal asynchronous receiver/transmitter
VHDL	: Very high speed integrated circuit Hardware Description Language
WSN	: Wireless Sensor Network
XOR	: Exlusive-OR işlemi

KABLOSUZ ALGILAYICI AĞLAR İÇİN SKIPJACK ŞİFRELEME ALGORİTMASININ FPGA İLE GERÇEKLENMESİ

Ergin ERYILMAZ

Anahtar Kelimeler: KAA, Ağ Güvenliği, Şifreleme, Skipjack, FPGA

Özet: Haberleşme güvenliği, son yıllarda yazılım ve donanım teknolojilerinde olduğu gibi, gelişen KAA uygulamalarında da gün geçtikçe önemini artıran vazgeçilmez bir unsur olmuştur.

Gelişen teknolojinin getirdiği yenilikler, günümüzde çok geniş bir kullanım alanı bulan KAA'lara önemli katkılar sağlamıştır. Bununla birlikte aynı teknolojiler, KAA uygulamalarında güvenlik önlemlerinin yok edilmesine yönelik girişimlerin de oluşmasına yol açmıştır. Bu durum, KAA güvenliğini ve bu güvenliğin bel kemiğini oluşturan şifreleme tekniklerini tüm KAA uygulamalarında vazgeçilmez hale getirmiştir.

Bu tez çalışmasının temel hedefi; KAA güvenliğinde kullanılan şifreleme algoritmalarının FPGA donanım ortamında gerçeklenmesidir. Bu amaç doğrultusunda KAA uygulamalarında sıklıkla tercih edilen Skipjack algoritması, VHDL dili kullanılarak, FPGA ile gerçeklenmektedir.

Bu tez çalışmasında donanım aracı olarak üzerinde Xilinx FPGA olan geliştirme kartı, yazılım aracı olarak; CodeGear C++ Builder 2009, Xilinx ISE Foundation ve Visual Studio 2008 kullanılmaktadır.

IMPLEMENTATION OF SKIPJACK CRYPTOLOGY ALGORITHM FOR WIRELESS SENSOR NETWORKS USING FPGA

Ergin ERYILMAZ

Keywords: WSN, Network Security, Cryptography, Skipjack, FPGA

Abstract: Networking security has become a vital fact day by day with an ever-increasing significance.

Improvements with the developing technologies have provided great contributions to WSNs' widespread deployment. In addition, the same technologies give rise to attempts which intent to break security precautions used in WSN applications. This situation has forced WSN security and cryptography methods create the backbone of this security in all WSN applications.

The main aim of the project presented in this thesis is to implement cryptography algorithms used in wireless network security as FPGA hardware platform. Therefore, Skipjack algorithm used frequently in WSN applications has been implemented using VHDL language.

In this thesis work, a Digilent starter board with Xilinx FPGA has been used as hardware platform, and CodeGear C++ Builder 2009, Xilinx ISE Foundation and Visual Studio 2008 have been used as software platform.

1. GİRİŞ

Bu tez çalışmasında, Kablosuz Algılayıcı Ağ'lar (KAA'lar) için önerilen bir şifreleme algoritması FPGA kullanılarak gerçekleştirilmiştir.

Kablosuz iletişim, elektronik ve elektro-mekanik sistemlerdeki son gelişmeler, düşük güçlü, küçük boyutlu, çok fonksiyonlu ve düşük maliyetli algılayıcı düğümlerinin kısa mesafede kesintisiz iletişimli kullanımına olanak sağlamaktadır. Bu küçük algılayıcı düğümler; algılama, veri işleme ve diğer bileşenlerle iletişim fonksiyonlarıyla kablosuz ağların temel düşüncesini meydana getirmektedir [1].

Bir kablosuz algılayıcı ağ, bir coğrafi alanda birbirine çok yakın dağılmış algılayıcı düğümlerinin birleşiminden oluşmaktadır. Algılayıcı düğümlerin pozisyonlarının önceden hesaplanma ya da saptanma zorunluluğu yoktur. Bu sayede, erişilemeyen bölgelerde rastgele algılayıcı düğüm dağıtılmasına ya da algılayıcıların felaket bölgelerinde kullanımına olanak sağlanabilir. Bu, aynı zamanda, algılayıcı düğüm protokollerinin ve algoritmalarının kendini denetleyebilir yeteneklerine sahip olması anlamına gelmektedir. KAA'ların ayırt edici diğer bir özelliği de ağı oluşturan düğümlerin doğrudan ya da dolaylı bir şekilde birbirlerine yardımcı olacak şekilde çalışmalarıdır. Algılayıcı düğümler, kendi üzerinde barındırdıkları işlemciyi kullanarak her iletim için aynı veri paketini göndermek yerine iletilmesi gereken gerekli veri paketini hesaplayacak kabiliyete de sahiptirler [1].

Yukarıda bahsedilen özellikler KAA'ların geniş bir yelpazede kullanımına olanak sağlamaktadır. Askeri alanlar, sağlık ve ev kullanımı bu alanlardan sadece bazılarıdır. Hızlı bir dağıtım, kendini organize edebilir yapı ve hata kontrol özelliklerine sahip olmasıyla, KAA'lar askeri kullanımlar için ideal bir ağ yapısıdır. Sağlık alanında algılayıcı düğümler, hastaların durumlarını görüntülemek ve yardımcı olmak amaçlı da kullanılabilmektedirler.

KAA'lardaki önemli bir hususta ağ güvenliğidir. Günümüzdeki karmaşık güvenlik protokolleri ve algoritmaları geleneksel kablolu ağ kullanımına uygundur. Bu tip ağlarda kullanılan güvenlik protokolleri ve algoritmaları kablosuz algılayıcı ağlar için tam uygulanabilir değildir. KAA'larda bu tip protokoller ve algoritmalar, etkin kaynak kullanımını sağlayacak şekilde optimize edilirler. KAA'larda güvenlik, geleneksel ağlardaki güvenliğe göre daha üst düzeyde olmasa da, her geçen gün daha geliştirilmektedir. Bu konuda, güvenliği artıran buna karşılık güç tüketimini ve kaynak kullanımını ise azaltan donanımsal ve yazılımsal çalışmalar yapılmaktadır.

FPGA'ler, KAA'larda kullanımı giderek artan, programlanabilir sayısal mantık tümleşik devreleridir. FPGA'ler ile hemen hemen tüm mantık işlemlerini gerçekleştirmek mümkündür.

Bu tezde sıklıkla kullanılan düşük hızlı mikrodenetleyicilerden herhangi birinin yerine Xilinx XC3S500E FPGA kullanılmaktadır. Mikrodenetleyici ile elde edilebilecek performans verileri, daha iyi sonuç alabilmek için FPGA ile denenmektedir. Seçilen FPGA düşük hızda çalışmaktadır ve Skipjack algoritmasının kaynak ihtiyaçlarını karşılayabilmektedir. Yapılan iş KAA'lara yönelik bir uygulama olduğundan çok yüksek kaynak ihtiyacı gerektiren bir FPGA kullanımına gerek kalmamaktadır. Gömülü yazılımın kod boyutunun düşük olması çok fazla kapıya sahip bir FPGA seçimine gerek duyulmamasına da olanak sağlamaktadır.

1.1. Literatürde Sunulan KAA Şifreleme (Kriptografi) Algoritması Uygulamaları

KAA'lardaki en büyük zorluklardan biri de protokol ve hizmetler için gereken kaynak rezervini sağlayabilmektir. Bundan dolayı şifreleme işlemlerinin, Simetrik Anahtar Şifreleme (Symmetric Key Encryption), Hash fonksiyonları ve Açık Anahtar Şifreleme (Public Key Cryptography) gibi ilkeler üzerine temellendirilmesi gereklidir. Bu ilkeler olmadan iletişim kanalının güvenliği, veri iletişimi gerçekleştiren ağ düğümlerinin doğruluğu ve veri paketlerinin bütünlüğü gibi KAA'larda esas olan güvenlik hizmetlerinin sağlanması olanaksız olacaktır.

Şifreleme ilkeleri, veri hesaplanabilirliği ve hafıza kullanımı açısından karmaşıktırlar. KAA'larda kullanılmak üzere seçilen donanım kaynakları, bu ilkeleri ve güvenlik hizmetlerini en kısa sürede işleyebilir yeterlilikte olmalıdırlar [5].

KAA'larda güvenliği sağlamada kullanılan yöntemlerden en önemlileri SKE (Symmetric Key Encryption) ve PKC (Public Key Cryptography)'dir. SKE, verinin bütünlüğünü ve güvenliğini garanti ederek bilginin temel güvenlik seviyesini sağlamaktadır. Eğer SKE, PKC ile karşılaştırılacak olursa; SKE, PKC'nin aksine karşılıklı veri iletişimine dahil olan düğümlerin doğrulama fonksiyonlarını gerçekleştirememektedir. Bununla birlikte SKE, güvenli anahtar altyapısı oluşturmak için Anahtar Yönetim Sistemine (Key Management System) ihtiyaç duyar. Diğer taraftan, PKC'nin işlem gereksinimlerinin de KAA'lar için yorucu ve maliyetli olduğu da bilinmektedir. PKC kullanımı KAA'lar için hemen hemen imkansız gözükse de Elliptic Curve Cryprography kullanımıyla gerçekleştirilebileceği düşünülmektedir [6].

Yukarıda bahsedildiği üzere KAA'larda şifreleme üç temel yöntem ile gerçekleştirilmekte ve bu yöntemlere bağlı farklı algoritmalar kullanılmaktadır. KAA'larda ister SKE, ister PKC veya Hash yöntemi kullanılsın, seçilen algoritma ile hedef; KAA güvenliğini en uygun kaynak kullanımı ile sağlayabilmektir.

SKE'nin iki temel tipi vardır. Bunlar; Blok Şifreleyiciler (Block Chiphers) ve Akış Şifreleyiciler (Stream Chiphers)'dir. Blok şifreleyiciler, sabit uzunluklu veri gruplarını değişmeyen dönüşümle işlerler. Tüm blok şifreleyiciler, blok boyutunu geçen verileri şifrelemek için sayıcı (CTR), şifre blok zincirleme (Chipher Block Chaining), MAC'li sayıcı (CCM) gibi gereksinimlere ihtiyaç duyar. Bunun yanında akış şifreleyiciler ise şifreleme sırasında değişen dönüşüme ve bağımsız haneleri işleyen yapıya sahiptir.

Basit ve en hızlı blok şifreleyicilerden biri de Skipjack [7] algoritmasıdır. Skipjack 80 bit uzunluklu anahtara sahip 64 bit'lik bir blok şifreleyici algoritmasıdır. 64 bit'lik veri bloğunu 16 bit'lik veri bloklarına bölerek, Kural A ve Kural B diye adlandırılan kuralları işleterek veriyi şifreler.

Diğer basit blok şifreleyici de blok boyutu, anahtar boyutu ve çevrim sayısı değişken parametreli RC5 [8] algoritmasıdır. Şifreleme, 3 ilke üzerinden gerçekleşir. Bunlar tamsayı ekleme, bit işlem düzeyinde XOR işlemi ve değişken rotasyonudur. Bununla birlikte RC5 algoritmasının anahtar düzenlemesi biraz karışıktır. RC5 eksikliklerini ortadan kaldırmak için RC6 [9] algoritması geliştirilmiştir.

En yaygın kullanılan fakat karmaşık olan blok şifreleyici de AES [10]'tir. AES anahtar boyutu 128, 192 veya 256 bit olabilen 128 bit'lik sabit blok boyutu kullanan, verileri 4x4 bayt dizi boyutunda işleyen bir algoritmadır.

Akış şifreleyici olarak da XOR tabanlı RC4 [10] algoritması, oldukça basit bir yapıya sahiptir. RC4, şifreleme için bitlerin rastgele akışlarını üretir. Bloklarının bit işlem düzeyinde AND, toplama ve karşılıklı değiştirme olması sebebiyle basit bir algoritmadır. RC4, 8 bit blok boyutu kullandığından çok fazla hafıza kullanımına ihtiyaç duymaz.

Yukarıda bahsedilen özelliklerinin yanında SKE'ler için anahtar dağıtımı büyük bir problemdir. Çünkü ağı oluşturan düğümlerin nerede bulunacakları ve bu düğümlerin anahtar dağıtımı için hangi diğer düğümlere komşu olacağı kestirilemez. Bu sorun, Anahtar Yönetim Sistemi'nin öneminin sürekli artmasında en büyük etkendir [11].

PKC yöntemini kullanan en ünlü algoritma RSA [12]'dır. RSA hem şifreleme hem de imzalama yapabilmektedir. Ancak işlem gereksinimleri, kaynak kısıtlığı olan algılayıcı düğümler için yüksek maliyetlidir. Bu yüzden yeni algoritmaların ECC (Elliptic Curve Cryprography) gibi daha uygun yaklaşımlar üzerine temellendirilmesi gerekmektedir [6]. ECC birçok yeni tipte ilke içermektedir. Bunlardan bazıları imzalar için ECDSA, ev anahtar kabulü (Key Agreement) için ECDH'tır.

Algılayıcı düğümler için faydalı olabilecek diğer yaklaşımlar NtruEncrypt [13], Rabin's Scheme [14] ve MQ-Scheme [15]'dır. KAA'larda yeni bir PKE yöntemi çalışmaları da önemli bir araştırma sahasıdır.

Bahsi geçen son yöntem olan Hash fonksiyonlarından ikisi SHA-1 [16] ve RIPEMD [17]'tir. Bu algoritmalar 32 bit'lik kelime boyutlarını kullanırlar. Fakat algılayıcı düğümle genellikle 8 bit ya da 16 bit işlemcileri barındırırlar. Bu uyumsuzluk nedeni ile KAA'lar için önemli sorun oluştururlar ve tercih edilmezler.

1.2. Tez Çalışmasının Amacı

KAA'larda veri güvenliği her geçen gün önemini artıran bir unsur olduğundan, güvenlik amacıyla kullanılan şifreleme yöntemleri ve algoritmaları bu konu üzerinde çalışan araştırmacılar tarafından artan bir ivme ile geliştirilmektedir. Bu yöntemler, donanımsal olarak küçülen ve kaynak tüketimi azaltılan algılayıcı düğümler için gün geçtikçe optimize edilebilir hale getirilmektedir. Kullanılan yöntem ve yazılımların iyileştirilmesi, donanımsal güncelleştirmeler ile paralel bir biçimde gerçekleştirilmektedir. Bu sayede daha az sistem kaynağı harcayan ve daha az güç tüketimine sahip donanımların kullanımını olanaklı kılınmaktadır. Önemi artan konu ise, yazılımsal iyileştirilmelerin uygulanabilirliğini sağlayan donanımsal iyileştirilmelerdir.

KAA'larda güvenlik, genellikle algılama işlerinin sonuçlarını da işleyen düşük hızlı ve düşük güç tüketimli bir mikrodenetleyici tarafından sağlanmaktadır. Bu mikrodenetleyiciler, geleneksel kablolu ağlarda kullanılan şifreleme algoritmalarının büyük bölümünü çeşitli iyileştirmelerle çalıştırabilir özelliktedir. Yalnız, seçilen

şifreleme yönteminin farklılığına göre, aynı mikrodenetleyici bazı durumlarda yetersiz kalmakta ve KAA algılayıcı düğümleri için kullanılabilir olamamaktadır.

Yukarıda bahsedilen hususların ışığında bu tez çalışmasında, KAA algılayıcı düğümlerde kullanılan şifreleme ve şifre çözme işlemini gerçekleştiren mikrodenetleyici yerine FPGA kullanarak, tezin ileriye dönük bir örnek çalışma olması amaçlanmıştır.

1.3. Tez Organizasyonu

Yapılan çalışmaların sunulduğu bu tez altı ana bölümden oluşmaktadır:

İlk bölümde tez çalışmasına konu olan problemin tanımı, çalışmanın amacı, literatürdeki benzer örnekleri ve tez çalışmasının amacı ve başlatılma sebebi hakkında bilgi sunulmaktadır.

İkinci bölümde kablosuz ağ güvenliği, kablosuz ağ güvenliğinde karşılaşılan güvenlik sorunları, bu sorunlara karşı çözüm önerileri, güvenliği sağlamada kullanılan yöntemler ve bu yöntemlere dayanan simetrik ve asimetrik şifreleme algoritmaları sunulmaktadır.

Üçüncü bölümde tez konusuna yönelik kullanılan FPGA'lerin mimarisi, üretim teknolojileri ve tez uygulamasında seçilen FPGA hakkında temel bilgiler sunulmaktadır.

Dördüncü bölümde FPGA donanımına yönelik VHDL dili ile yazılan gömülü yazılımın akış diyagramı ve yazılımın çalışması hakkında bilgiler verilmektedir.

Beşinci bölümde tezde kullanılan, FPGA gömülü yazılım geliştirme ortamı, kullanıcı arayüzü geliştirme ortamı ve kullanımı, donanım arayüzleri, deneysel çalışmalar, benzer uygulama örneği ve karşılaştırılmalı başarım değerlendirmelerinden bahsedilmiştir.

Son bölümde geliştirilen yazılımların uygulama performansları ve karşılaşılan güçlükler vurgulanarak diğer uyguluma örnekleri ile karşılaştırma sonuçlarının başarım değerlendirmesi yapılmakta ve ileriye dönük çalışmalardan bahsedilmektedir.

2. KABLOSUZ ALGILAYICI AĞ GÜVENLİĞİNDE KULLANILAN ŞİFRELEME YÖNTEMLERİ

2.1. Giriş

Hızla gelişen teknoloji ile kablosuz algılayıcı ağ düğümleri, daha küçük boyutlara inmekte, daha az güç harcamakta ve yeni fonksiyonlar kazanmaktadırlar. Bu yüzden KAA'lardaki güvenli iletişim teknikleri önemini gün geçtikçe artırmaktadır. Bu durumda öne çıkan önemli husus, kullanılan şifreleme yönteminin daha az sistem kaynağı harcayacak yani daha az güç tüketecek bir özelliğe sahip olmasıdır.

2.2. Kablosuz Algılayıcı Ağlarda Klasik Güvenlik Yöntemlerinin Uygulanabilirliliği

Güvenlik; kimlik doğrulama, bütünlük, gizlilik, inkar edilemezlik, tekrar edilemezlik karakterlerini içeren yaygın bir biçimde kullanılan bir terimdir [18]. KAA'larda kullanılan veriye bağlılık ve ihtiyaç duyma seviyesi arttıkça, ağ üzerinde aktarılan verinin güvenli iletimi daha da ön plana çıkmaktadır [19]. Güvenliği sağlamada kullanılan klasik yöntemlerden ikisi; Simetrik ve Asimetrik şifrelemedir. Simetrik ve Asimetrik şifreleme yöntemlerinin birbirlerine göre avantaj ve dezavantajları bulunmaktadır. Bununla birlikte her iki yöntemin de avantajlarını birleştiren daha etkin melez yöntemler de son zamanlarda önemini artırmaktadır. Bu iki yöntemin melezlenmesi sonucu oluşan dezavantajların ve eksikliklerin giderilmesi son zamanlarda KAA'larda önemini üst seviyeye çıkaran çalışmalardır.

2.3. KAA'larda Güvenlik Sorunları ve Çözüm Önerileri

Kablosuz algılayıcı ağlarda ön plana çıkan beş güvenlik açığı bulunmaktadır. Kimlik doğrulama, bütünlük, gizlilik, inkar edilemezlik ve tekrar edilemezlik bu önemli beş karakterdir.

Kimlik doğrulama (Authentication), bilgiyi gönderen tarafın kimliğinden emin olmaktır. Herhangi bir ağa, kimliği ağ tarafından bilinmeyen bir üye katılamaz. Bütünlük (Integrity), iletişimde kullanılan bilginin iletim sırasında bozulmamasıdır. Diğer bir ifade ile gönderilen ve alınan verinin birebir örtüşmesidir. Veri bütünlüğünü bozmaya yönelik ataklar KAA'larda jamming yöntemleri kullanılarak yapılmaktadır. Gizlilik (Privacy), iletişimde kullanılan verinin gizliliğinin sağlanmasıdır. İnkar edilemezlik (Non-repudiation), bilgiyi gönderen tarafın ve bilgiyi alan tarafın bilgiyi işlediğini inkar edememe durumudur. İletişim sırasında kontrollü veri işleme inkar edilemezlik prensibi üzerine kurulmuştur. Tekrar edilemezlik (Anti-playback), iletim sırasında gönderilen bilginin taklit edilip tekrar gönderilememesidir. Ağ yapısı, iletişimde gönderilen verilerin kopyalanıp tekrar gönderilmesini engelleyebilecek bir özelliğe sahip olmalıdır. Ağa, sahte (fake) erişimler bu şekilde engellenebilmektedir.

Yukarıda ifade edilen beş karakteristik özellik bir KAA'nın güvenlik omurgasını oluşturmaktadır. Bu özelliklerden herhangi birisinin eksikliği ya da yetersizliği tüm kablosuz sistem için tehdit oluşturabilecek bir unsur olmaktadır. Bu nedenle bu beş karakteristiği sağlamada kullanılan, belirgin iki yöntem bulunmaktadır. Bu yöntemler Şifreleme (Kriptografi) ve Steganografi'dir. Şifreleme verinin gizliliğini amaçlamakta, Steganografi ise verinin varlığının gizlenmesini amaçlamaktadır [20].

2.4. KAA'larda Kullanılan Şifreleme Yöntemleri ve Skipjack Şifreleme Algoritması

KAA'larda şifreleme yöntemi olarak Simetrik (Symmetric) ve Asimetrik (Public Key) şifreleme yöntemlerinden yararlanılmaktadır. Bunun yanında son zamanlarda bu iki yöntemin kısmi birleştirilmesi ile oluşturulan hibrit yöntemlerden de yararlanılmaktadır.

Simetrik şifreleme yönteminde ortak anahtar dağıtımı sorun teşkil edebileceğinden bu eksikliği gidermek için anahtar dağıtım şemalarından yararlanılmaktadır. Simetrik şifreleme yöntemindeki bu sorunu gidermek için seçilen diğer bir yöntem Asimetrik şifrelemedir. Bu yöntem KAA'lar için büyük önem taşıyan sistem kaynaklarının

daha fazla tüketilmesine neden olmakta ve ağır işlem gerektirmesi sebebiyle algılayıcı batarya ömrünün kısalmasına yol açmaktadır. Bununla birlikte, simetrik ve asimetrik yöntemi içeren hibrit yöntemler son zamanlarda önemini artırmaktadır.

KAA'ların simetrik şifreleme yöntemine daha uygun olması sebebiyle bu tez çalışmasında basit ama güçlü bir simetrik şifreleme algoritması olan Skipjack tercih edilmiştir. Skipjack basit ama güçlü algoritması ile ön plana çıkan ve KAA'lar için önemli unsur olan daha az sistem kaynağı tüketimine katkıda bulunan bir yapıya sahiptir.

2.4.1. Simetrik Şifreleme

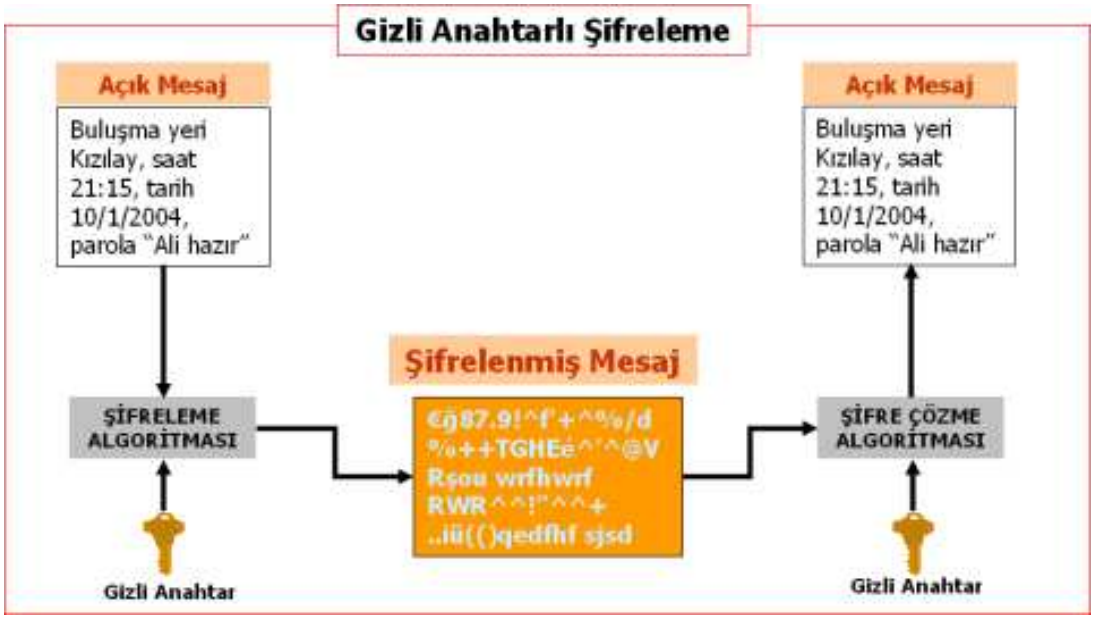
Pratikte simetrik şifrelemede, şifreleme ve şifre çözme işlemleri için aynı anahtar kullanılmaktadır. Simetrik şifreleme denmesinin nedeni, şifreleme ve şifre çözme işlemlerinde aynı anahtar ile birbirinin simetriği olan algoritmalar kullanılmasıdır. Şekil 2.1.'de bu durum görülebilmektedir.

Simetrik şifreleme yönteminin en büyük avantajı şifreleme hızının yüksek olmasıdır. Simetrik şifrelemeye yönelik tasarlanan donanımlar, saniyede yüzlerce Megabayt'lık veri işleyebilmektedir. Ayrıca simetrik şifreleme uzun geçmişe sahip geleneksel bir şifreleme tekniğidir.

Simetrik şifreleme tekniğinin en büyük dezavantajı, veriyi gönderen ve veriyi alan tarafların aynı şifreleme anahtarını biliyor olması zorunluluğudur. Bu yüzden gönderen ve alan taraf için anahtar paylaşımı güvenli yöntemlerle yapılmalıdır. Bu da anahtar dağıtım problemlerini beraberinde getirmektedir.

Simetrik şifreleme tekniğini kullanan algoritmalara örnek olarak XOR Şifreleme, DES [21], 3DES, IDEA, DESX, Skipjack, RC2, RC4, RC5 algoritmaları verilebilir. En bilinen simetrik şifreleme algoritmaları DES (Data Encryption Algorithm), 3DES, RC2 ve RC4'tür.

Bir simetrik şifrenin anahtar uzunluğu ne kadar uzun olursa şifreleme kuvveti de o derece yüksek olur. Günümüzde 40 bit anahtar uzunluğu yeterli görülmeyip zayıf kabul edilirken, 128 bit ve üzeri anahtar uzunluğuna sahip algoritmalar kuvvetli kabul edilmektedir [22, 23].



Şekil 2.1: Simetrik Şifreleme Yöntemi

2.4.1.1. Skipjack Algoritması ve Teknik Özellikleri

Skipjack, 80 bit kripto-değişkenli 64 bit'lik bir şifreleme algoritmasıdır. Uygulama modları; DES için yapılmış olan uygulama modları FIPS-81 tanımının bir alt kümesinden oluşmaktadır. Bunlar aşağıdaki modları içermektedir [7]. Bu tez çalışmasında tercih edilen Kodkitabı modudur.

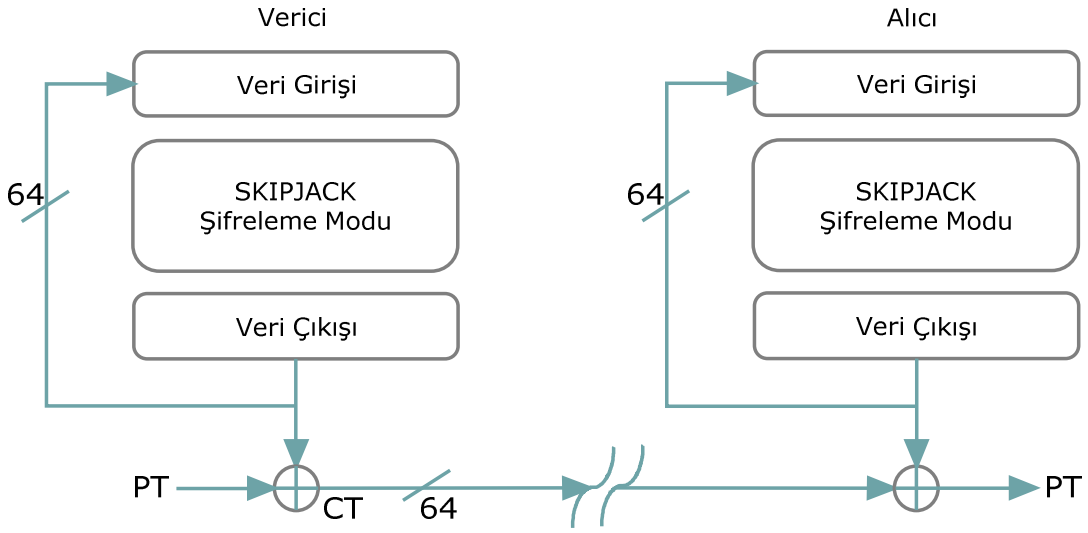
- Output Feed-Back (OFB) (Çıkış Geribeslemeli) Modlar — 64 bit
- Chipher Feed-Back (CFB) (Şifreleyici Geribeslemeli) Modlar — 64, 32, 16 veya 8 bit
- Codebook (Kodkitabı) — 64 bit
- Chipher-Block Chaining (Şifreleyici-Blok Zincirleme) — 64 bit

2.4.1.1.1. Skipjack uygulama modları

Skipjack dört uygulama modundan oluşmaktadır:

2.4.1.1.2. Çıkış Geribeslemeli (Output Feed-Back) mod

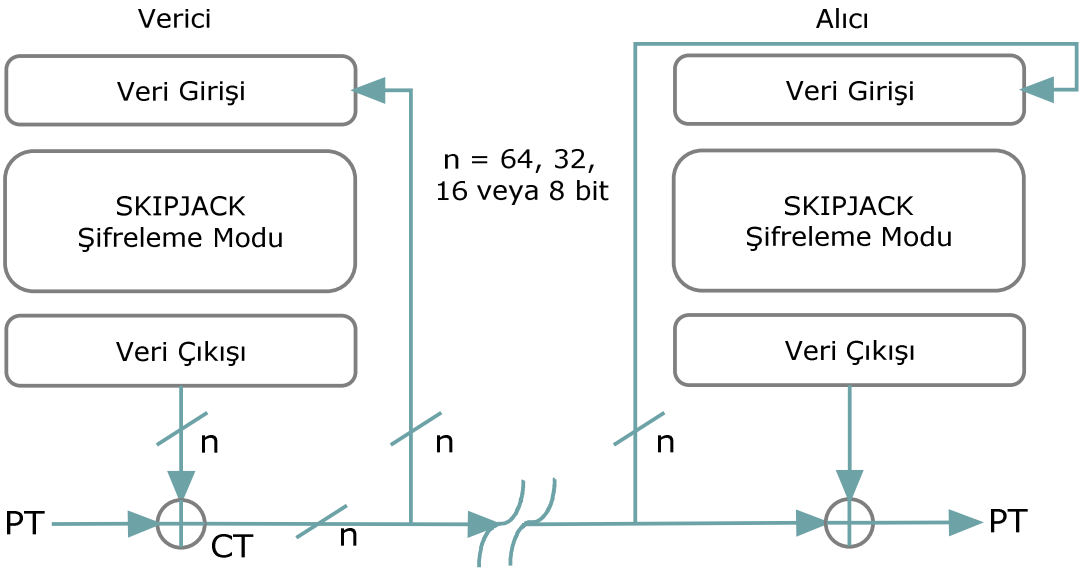
Şekil 2.2.'deki modda; girilen 64 bit'lik veri şifrelenir ve şifreleme sonucu oluşan 64 bit'lik şifreli veri tekrar bir sonraki çevrimde veri girişi olarak kullanılır. Tüm çevrim tamamlandıktan sonra oluşan son şifreli veri, şifrelenmek istenen asıl veri ile EX-OR işlemine tabi tutularak gönderilmeye hazır şifreli veri oluşturulur. Şifre çözme işlemi için de benzer yöntem ters yönde uygulanır. Son aşama olarak, şifre çözme işleminde elde edilen 64 bit'lik veri, verici tarafından gönderilen şifreli veri ile EX-OR'lanarak asıl veri elde edilir.



Şekil 2.2: Çıkış Geribeslemeli Mod

2.4.1.1.3. Şifreleyici Geribeslemeli (Chipher Feed-Back) mod

Şekil 2.3.'teki modda; çevrim sonucu oluşan şifreli veri asıl veri ile EX-OR'lanarak şifreli veri oluşturulur ve bir sonraki çevrimde tekrar kullanılır. Tüm çevrim sonucunda gönderilmeye hazır son şifreli veri oluşturulur. Şifre çözme işlemi de benzer şekilde şifre çözme çevrimleriyle ters yönde gerçekleştirilir.

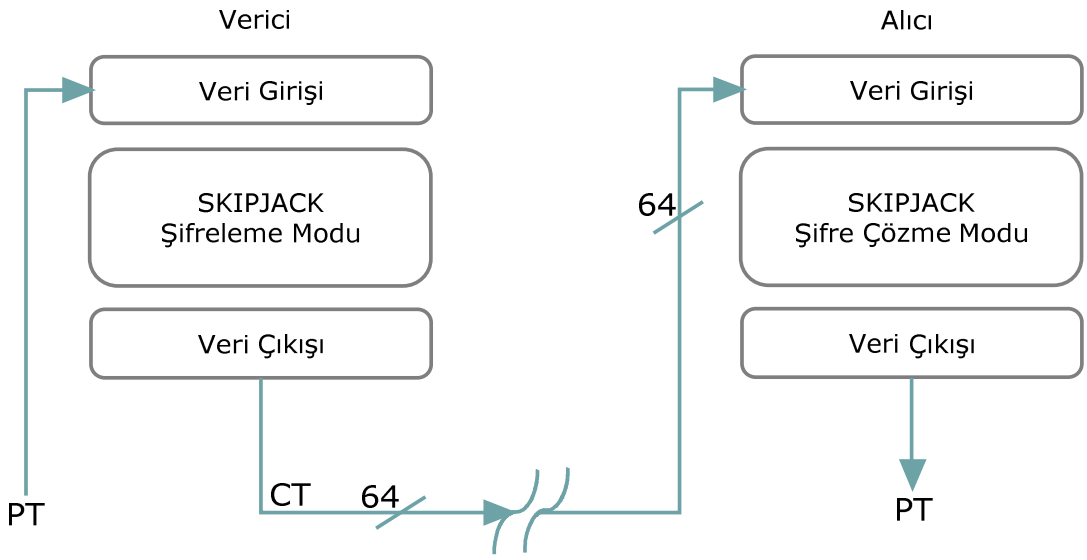


Şekil 2.3: Şifreleyici Geribeslemeli Mod

2.4.1.1.4. Kodkitabı (Codebook)

Şekil 2.4.'teki mod diğer modlara göre daha basit ve az işlem gücü gerektiren bir moddur. Şifrelenmesi istenen 64 bit'lik veri girilir ve 32 çevrim sonucunda şifreli veri oluşturulur. Diğer yöntemlerin aksine bu yöntemde asıl veri 1 kez veri girişinde kullanılır. Şifre çözme işlemi de benzer şekilde şifre çözme çevrimleriyle ters yönde 32 adımda gerçekleştirilir.

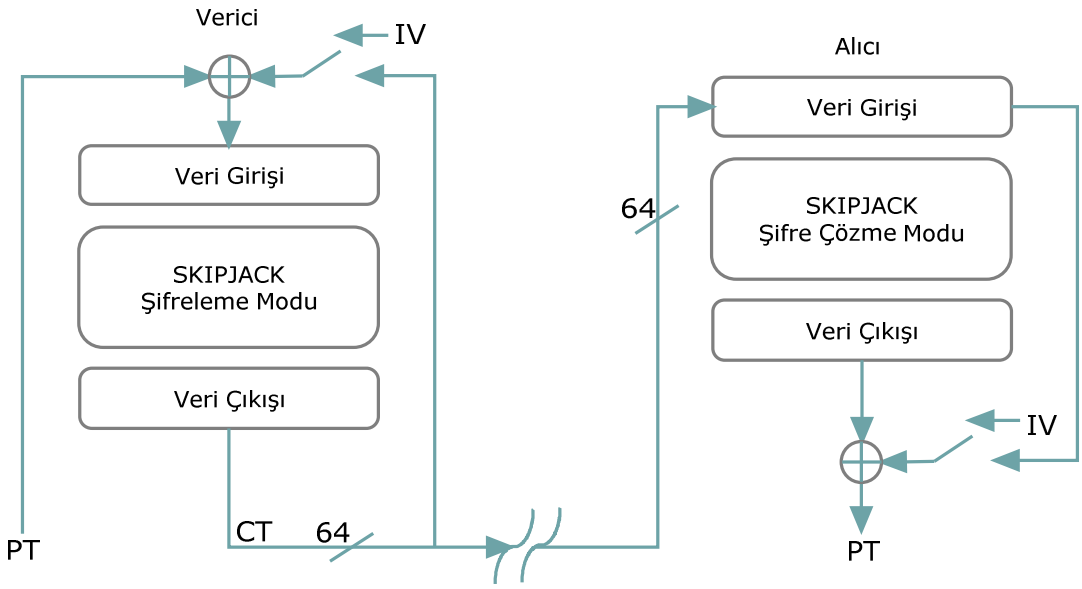
Daha az işlem gücü gerektirmesiyle bu tez çalışmasında bu mod uygulanmaktadır.



Şekil 2.4: Kodkitabı

2.4.1.1.5. Şifreleyici-Blok zincirleme

Şekil 2.5.'teki mod Şekil 2.3.'teki moda benzer şekilde kullanılır. Tek çevrim sonucu oluşan şifreli veri bir sonraki çevrimde asıl veri ile EX-OR'lanarak tekrar giriş verisi olarak kullanılır. Buradaki fark ise tek çevrimde oluşan şifreli verinin bir sonraki çevrimde kullanımının bir anahtara bağlanmasıdır. Şekilde görülen "IV", bu anahtarı temsil etmektedir. Benzer şekilde ters işlemler gerçekleştirilerek verici tarafından gönderilen şifreli veri asıl veriye dönüştürülür. Buradaki önemli nokta; şifrelemede ve şifre çözmede, anahtarların aynı çevrimde kapatılması ya da açılmasıdır. Aksi taktirde senkronizasyon sorunundan şifre çözme işlemi gerçekleştirilemez.



Şekil 2.5: Şifreleyici-Blok Zincirleme

2.4.1.2. Skipjack algoritmasının teknik özellikleri

2.4.1.2.1. Notasyon ve terminoloji

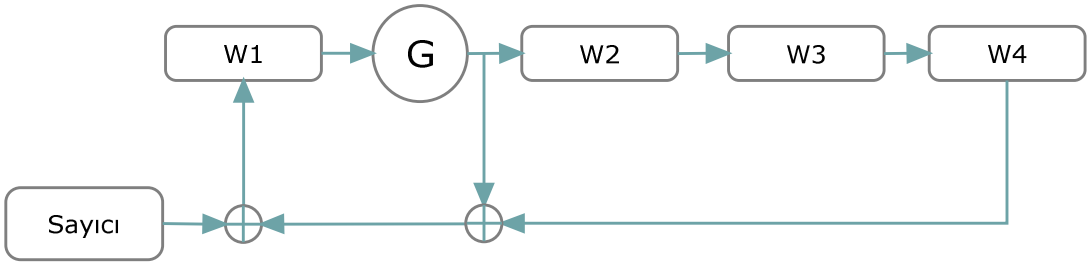
- V^n : Tüm n-bit değerlerden oluşan küme
- kelime : V^{16} öğesi, 16 bit'lik değer
- bayt : V^8 öğesi, 8 bit'lik değer
- V^n permütasyonu : V^n'den V^n'e dönüştürülebilir (bire-bir ve üzerine) bir fonksiyon. Bu yüzden değerlerin içerisindeki bitlerin değil de V^n içerisindeki değerlerin permütasyonu alınır.
- $X \oplus Y$: X ve Y'nin bit işlem düzeyinde özel-veya durumu
- $X \parallel Y$: X ve Y'nin bağlanmış hali. X ve Y'nin bayt olduklarını farz edelim. Bu durumda, $X \parallel Y = X.2^8 + Y$ yani kelime olacaktır. Diğer bir deyişle, X yüksek değerlikli bayt, Y düşük değerlikli bayt'tır.

2.4.1.2.2. Temel yapı

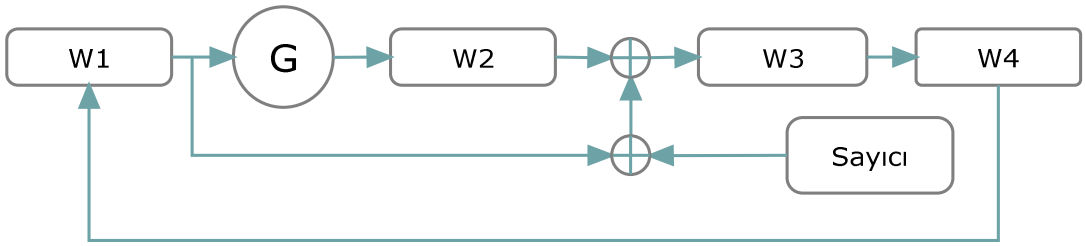
Skipjack 4 kelimelik (word) yani 8 bayt'lık veri bloklarını aşağıda gösterilen iki adım kuralını (A ve B) sırayla uygulayarak şifreler. Kural A Aşağıdaki işlemleri gerçekleştirir:

- G w_1'*i* üretir
- Yeni üretilen w_1; G çıkışı, sayıcı ve w_4'ün XOR'lu halidir.
- w_2 ve w_3 kelimeleri 1 kaydedici sağa kaydırılır; yani, sırasıyla w_3 ve w_4 olurlar.
- Yeni w_2 G çıkışıdır.
- Sayıcı 1 artırılır.

Kural B de benzer şekilde çalışmaktadır.



Şekil 2.6: Skipjack Adım Kuralı A



Şekil 2.7: Skipjack Adım Kuralı B

2.4.1.3. Kural A

$$w_1^{k+1} = G^k(w_1^k) \oplus w_4^k \oplus sayici^k$$

$$w_2^{k+1} = G^k(w_1^k)$$

$$w_3^{k+1} = w_2^k$$

$$w_4^{k+1} = w_3^k \qquad (2.1)$$

2.4.1.4. Kural B

$$w_1^{k+1} = w_4^k$$
$$w_2^{k+1} = G^k(w_1^k)$$
$$w_3^{k+1} = w_1^k \oplus w_2^k \oplus sayıcı^k$$
$$w_4^{k+1} = w_3^k \quad (2.2)$$

2.4.1.5. Ters Kural A

$$w_1^{k-1} = (G^{k-1})^{-1}(w_2^k)$$
$$w_2^{k-1} = w_3^k$$
$$w_3^{k-1} = w_4^k$$
$$w_4^{k-1} = w_1^k \oplus w_2^k \oplus sayıcı^{k-1} \quad (2.3)$$

2.4.1.6. Ters Kural B

$$w_1^{k-1} = (G^{k-1})^{-1}(w_2^k)$$
$$w_2^{k-1} = (G^{k-1})^{-1}(w_2^k) \oplus w_3^k \oplus sayıcı^{k-1}$$
$$w_3^{k-1} = w_4^k$$
$$w_4^{k-1} = w_1^k \quad (2.4)$$

2.4.1.6.1. Adım sırası

Algoritma toplam 32 adıma ihtiyaç duymaktadır.

- Şifrelemek İçin: Giriş w_i^0'dır ($1 \leq i \leq 4$ ve başlangıç adımı için $k = 0$'dır). Sayıcı 1'den başlatılır. Kural A 8 kez işletilir ve ardından Kural B de 8 kez işletilir. Bu 16 çevrimlik aşamadan sonra Kural A tekrar 8 kez işletilir ve ardından Kural B de tekrar 8 kez işletilir. Bu işlemler yapılırken her adımda sayıcı 1 artırılır.

Toplam 32 çevrimden sonra çıkış w_i^{32} $(1 \leq i \leq 4)$ olur.

- Şifre Çözmek İçin: Giriş w_i^{32}'dir ($1 \leq i \leq 4$ ve başlangıç adımı için $k = 32$'dir). Sayıcı 32'den başlatılır. Kural B^{-1} 8 kez işletilir ve ardından Kural A^{-1} de 8 kez işletilir. Bu 16 çevrimlik aşamadan sonra Kural B^{-1} tekrar 8 kez işletilir ve ardından Kural A^{-1} de tekrar 8 kez işletilir. Bu işlemler yapılırken her adımda sayıcı 1 azaltılır. Toplam 32 çevrimden sonra çıkış w_i^0 $(1 \leq i \leq 4)$ olur.

2.4.1.6.2. G-Permütasyonu

V^{16}'daki kriptodeğişken-bağımlı G permütasyonu dört çevrimlik Feistel yapıdır. Çevrim fonksiyonu F-tablosu olarak da adlandırılan bir sabit bayt'lardan oluşan tablodur (V^8'deki permütasyon). Her G çevrimi 1 bayt'lık kripto-değişkeni de birleştirmektedir.

Aşağıda fonksiyonun iki karakteristiği de verilmiştir:

- Rekürsif Olarak (Matematiksel Olarak):

  $g_i = F\ (g_{i-1} \oplus cv_{4k+i-3}) \oplus g_{i-2}$ (k adım sayısıdır ve başlangıç adım değeri 0'dır).

  F bayt kullanım tablosu ve cv_{4k+i-3} kripto-değişken listesinde $(4k + i - 3)$'üncü bayt olmak üzere;

  $Gk\ (w = g_1 \| g_2) = g_5 \| g_6$ eşitliği elde edilir. Bundan dolayı;

  $g_3 = F\ (g_2 \oplus cv_{4k}) \oplus g_1$
  $g_4 = F\ (g_3 \oplus cv_{4k+1}) \oplus g_2$
  $g_5 = F\ (g_4 \oplus cv_{4k+2}) \oplus g_3$
  $g_6 = F\ (g_5 \oplus cv_{4k+3}) \oplus g_4$

  olur.

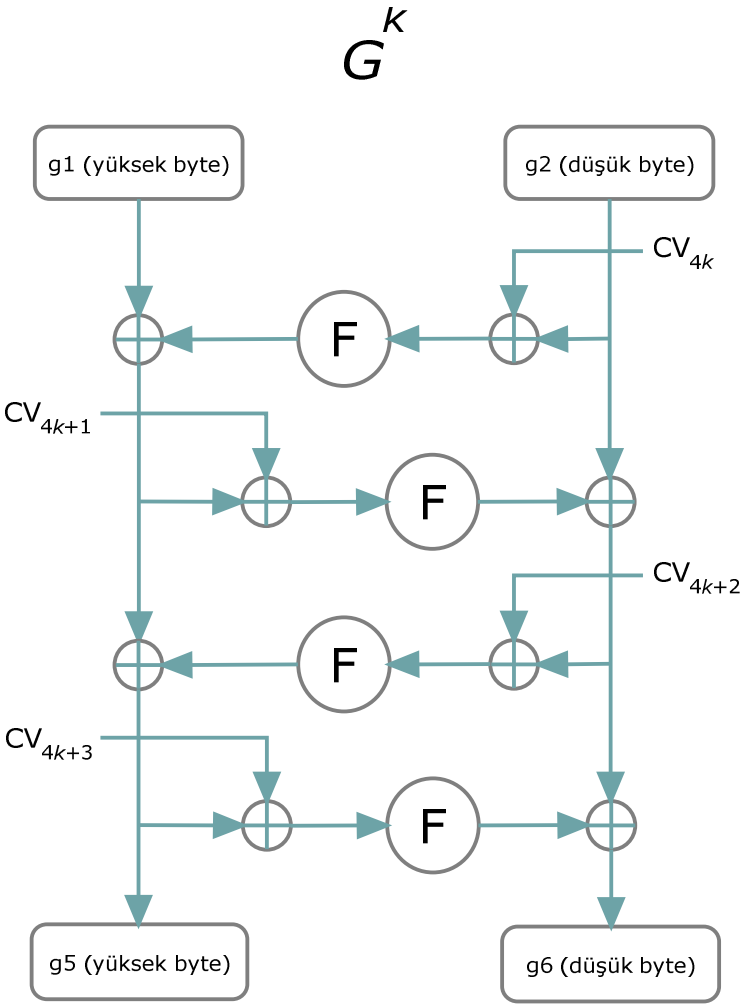
Benzer şekilde tersi;

$g_{i-2} = F(g_{i-1} \oplus cv_{4k+i-3}) \oplus g_i$ olmak üzere;

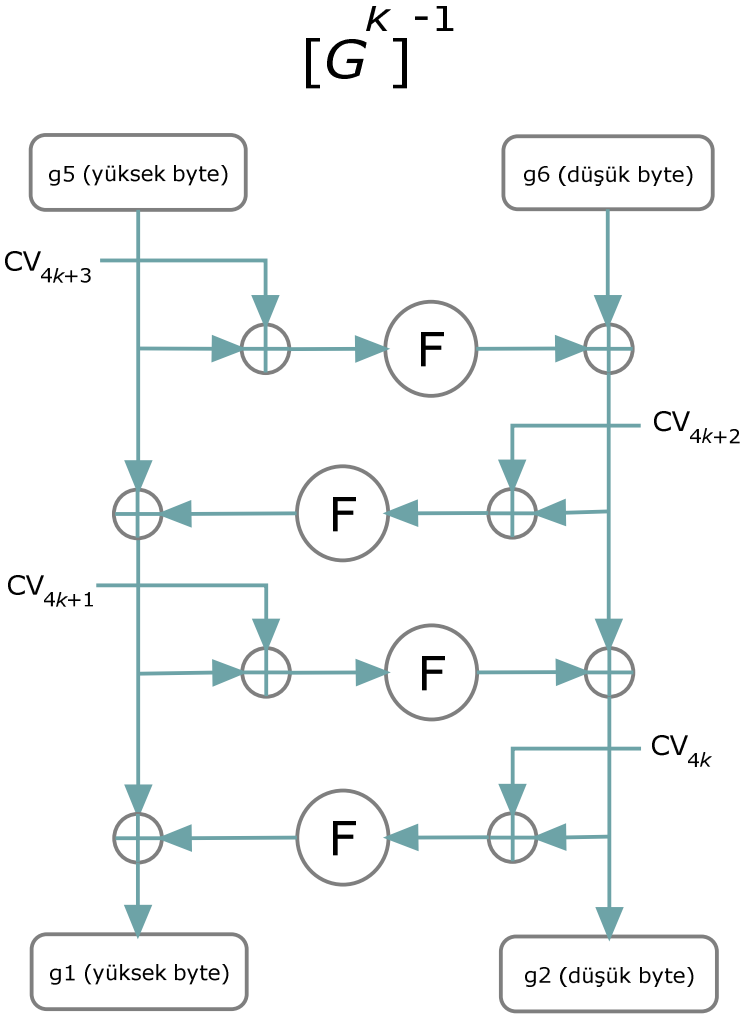
$[G^k]^{-1}(w = g_5 \| g_6) = g_1 \| g_2$

bulunur.

- Şematik Olarak:



Şekil 2.8: G Permütasyonu



Şekil 2.9: G Permütasyonu'nun Tersi

2.4.1.6.3. Kripto-değişken listesi

Kripto-değişken 0'dan 9'a kadara adlandırılan ve doğal sırasıyla kullanılan 10 bayt uzunluğunda bir değişkendir. G permütasyon tanımında verilen liste bayt'ları mod-10'a göre derlenir.

2.4.1.6.4. F-Tablosu

Skipjack F-Tablosu Tablo 2.1.'de onaltılık biçimde verilmiştir. Giriş indeksinin yüksek değerlikli 4 biti satır, düşük değerlikli 4 biti sütundur. Örneğin: F(7a) = d6.

Tablo 2.1: F-Tablosu

	x0	**x1**	**x2**	**x3**	**x4**	**x5**	**x6**	**x7**	**x8**	**x9**	**xA**	**xB**	**xC**	**xD**	**xE**	**xF**
x0	a3	d7	09	83	f8	48	f6	f4	b3	21	15	78	99	b1	af	f9
x1	e7	2d	4d	8a	ce	4c	ca	2e	52	95	d9	1e	4e	38	44	28
x2	0a	df	02	a0	17	f1	60	68	12	b7	7a	c3	e9	fa	3d	53
x3	96	84	6b	ba	f2	63	9a	19	7c	ae	e5	f5	f7	16	6a	a2
x4	39	b6	7b	0f	c1	93	81	1b	ee	b4	1a	ea	d0	91	2f	b8
x5	55	b9	da	85	3f	41	bf	e0	5a	58	80	5f	66	0b	d8	90
x6	35	d5	c0	a7	33	06	65	69	45	00	94	56	6d	98	9b	76
x7	97	fc	b2	c2	b0	fe	db	20	e1	eb	d6	e4	dd	47	4a	1d
x8	42	ed	9e	6e	49	3c	cd	43	27	d2	07	d4	de	c7	67	18
x9	89	cb	30	1f	8d	c6	8f	aa	c8	74	dc	c9	5d	5c	31	a4
xA	70	88	61	2c	9f	0d	2b	87	50	82	54	64	26	7d	03	40
xb	34	4b	1c	73	d1	c4	fd	3b	cc	fb	7f	ab	e6	3e	5b	a5
xC	ad	04	23	9c	14	51	22	f0	29	79	71	7e	ff	8c	0e	e2
xD	0c	ef	bc	72	75	6f	37	a1	ec	d3	8e	62	8b	86	10	e8
xE	08	77	11	be	92	4f	24	c5	32	36	9d	cf	f3	a6	bb	ac
xF	5e	6c	a9	13	57	25	b5	e3	bd	a8	3a	01	05	59	2a	46

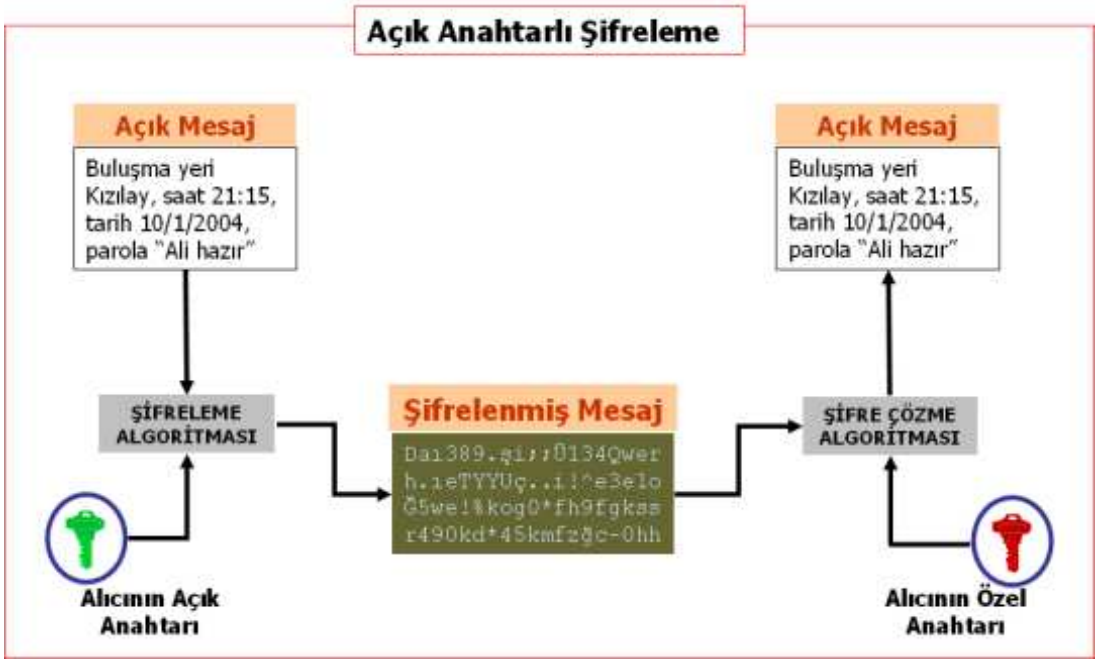
2.4.2. Asimetrik (Public Key) Şifreleme

Simetrik şifreleme tekniğindeki anahtar dağıtım problemlerini çözmek için Gizli anahtar (Private Key) ve Anonim anahtar (Public Key) çiftinden yararlanılan bir şifreleme yöntemidir. Anahtarlar karşılıklı değiştirilebilir niteliktedir. Karşı tarafa gönderilmek istenen veri gizli anahtar ile şifrelenir; karşı taraf da aldığı veriyi ancak gönderenin anonim anahtarı ile deşifre edebilir. Şekil 2.10.'da bu durum görülebilmektedir.

Asimetrik şifreleme veri bütünlüğünü garantiye alma amaçlı da kullanılmaktadır. Yani şifrelenmek istenen veri özel anahtarla şifrelenince, yalnızca şifreleyen tarafın özel anonim anahtarıyla şifre çözülebilir. Böylece, veriyi şifreleyenin anonim anahtarıyla şifreyi çözen taraf, gönderilen verinin, veriyi şifreleyen tarafından gönderildiğinden emin olur. Bu yöntem Sayısal İmza'nın da temelini oluşturur.

Asimetrik şifrelemenin simetrik şifrelemeden kayda değer farkı da anahtar yönetiminin basitliğidir. Asimetrik şifrelemenin zayıf yönü ise simetrik şifrelemeye göre yavaş kalmasıdır. Ancak kırılması daha zor bir yöntemdir. Buna ek olarak da anahtar uzunlukları daha fazladır.

En bilinen asimetrik şifreleme algoritmaları RSA, Diffie-Hellman, DSA ve Elliptic Curve Cryptography ‘dir. Bunlardan en meşhuru ve en çok kullanılanı RSA’dır. RSA’nın gücü, büyük sayıların asal çarpanlarına bölünebilmesindeki zorluğa temellendirilmiştir [24].



Şekil 2.10: Asimetrik Şifreleme Yöntemi

2.4.3. Şifreleme sistemlerin karşılaştırılması

Kimlik doğrulama, bütünlük, gizlilik, inkar edilemezlik, tekrar edilemezlik ve performans kriterleri göz önünde bulundurulursa Tablo 2.2.’deki veriler elde edilir.

Tablo 2.2: Kripto Sistemlerinin Karşılaştırılması

Konu	Simetrik Şifreleme	Asimetrik Şifreleme
Kimlik Doğrulama	-	Sağlar
Bütünlük	-	Sağlar
Gizlilik	Sağlar	Sağlar
İnkar Edilemezlik	-	Sağlar
Tekrar Edilemezlik	-	Sağlar
Performans	Hızlı	Yavaş

2.5. KAA ve Şifreleme

KAA güvenliği düşünüldüğünde, veri iletişiminde şifreleme yönteminin kullanılması kaçınılmazdır. Bu yüzden uygulamada şifreleme kullanılmayan bir KAA neredeyse yoktur. Çok geniş yelpazede kullanılan KAA'lar için askeri alanda kullanımda veri güvenliği en üst noktada olsa da günümüzde ev ortamında kullanılan KAA'larda da üst düzeyde veri güvenliği gerekebilmektedir.

KAA'larda güvenlik için klasik ağlarda kullanılan şifreleme yöntemleri seçilmektedir. Bu yöntemlerin kullanımı, şifreleme yöntemlerinin KAA'lara yönelik optimize edilmesiyle gerçekleştirilir. Aksi takdirde şifreleme algoritmalarını çalıştırabilmek olanaksız olacaktır.

2.6. Sonuç

Bu bölümde ifade edilen Simetrik ve Asimetrik şifrelemenin birbirlerine göre avantajları ve dezavantajları bulunmaktadır. KAA kullanımında hangi şifreleme yönteminin seçileceği; hafıza, güç tüketimi ve kaynak tüketimi gibi uygulamaya özel değişebilen değerlerin en uygun düzeyde karşılanabilmesine bağlıdır.

KAA'ların genel özellikleri göz önünde bulundurulduğunda ise güvenliği sağlama amaçlı kullanılabilecek en uygun şifreleme yönteminin Simetrik şifreleme olduğu görülmektedir. Simetrik şifreleme beraberinde getirdiği anahtar paylaşım problemlerine rağmen, sistem kaynaklarını en az tüketecek yapıya sahiptir.

Son zamanlarda Asimetrik şifreleme dezavantajları yazılımsal iyileştirmelerle azaltılmaktadır. KAA'larda yaygın olarak kullanılmasa da ileriye dönük çalışmalarla Asimetrik şifreleme yönteminin kullanım oranının artacağı tahmin edilmektedir.

Bir diğer hususta, üzerinde çalışılan yeni bir konu olan Simetrik ve Asimetrik şifreleme yöntemlerinin avantajlarını birleştiren melez şifreleme yöntemidir. Bu yöntem sayesinde Simetrik şifrelemenin hızı, kaynak ve hafıza kullanımı, güç tüketimi gibi öğeleri; Asimetrik şifrelemenin de güvenlik performansını artıran yönü ön plana çıkmaktadır. Eksikliklerinin olmasına rağmen yakın gelecek için KAA'larda sıklıkla kullanılabilecek bir yöntem olacağı düşünülmektedir.

3. SAHADA PROGRAMLANABİLİR KAPI DİZİLERİ (FPGA)

3.1. Giriş

Field-Programmable Gate Array (FPGA)'ler günümüzde sayısal devre tasarımı ortamının kilit mekanizması haline gelmişlerdir. Programlanabilir mantık fonksiyonelliğiyle ve programlanabilir ara bağlantı özelliğine sahip olan mimarileriyle sayısal donanım tasarımının güçlü parçasını oluşturmaktadırlar [25].

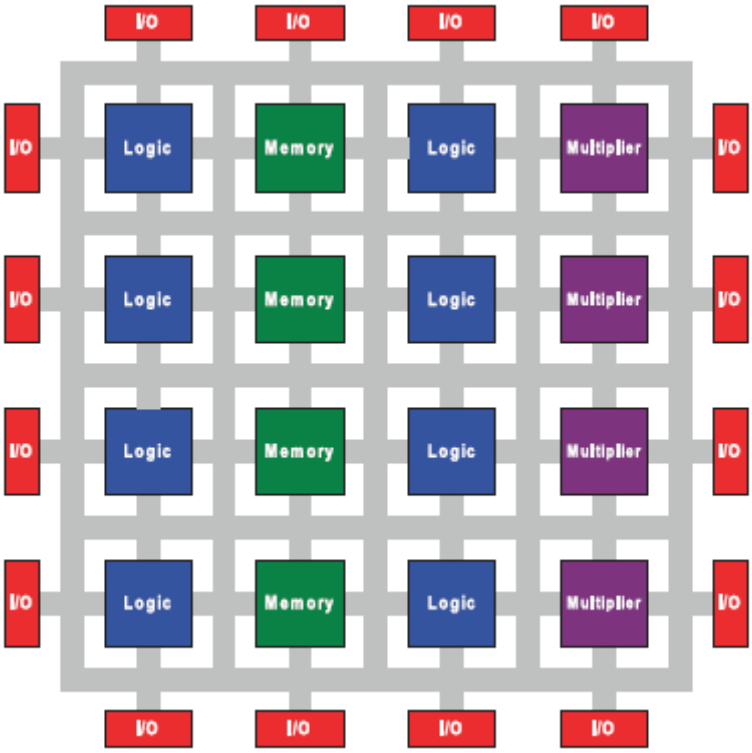
FPGA'ler, programlanabilir mantık blokları ve bu bloklar arasındaki ara bağlantılardan oluşan ve geniş uygulama alanlarına sahip olan sayısal tümleşik devrelerdir. Kullanıcı kaynaklı mantık devrelerini gerçekleştirme amacına yönelik üretilirler. Yani FPGA içerisindeki her mantık bloğunun nasıl davranacağı, mantık bloğunun nasıl bir fonksiyona sahip olacağı kullanıcı tarafından tasarlanabilmektedir. [26].

Tercih edilen şifreleme algoritmasının yazılımsal tasarımlarının hızını ve performansını donanımsal ortamda artırabilmek amacıyla, paralel veri işleme yeteneğine ve tekrar programlanabilir bir mimariye sahip olması sebebiyle FPGA tercih edilmiştir.

3.2. FPGA Mimarisi

FPGA'ler, Şekil 3.1.'de görüldüğü gibi, blokların kendi aralarında programlanabilmesini sağlayan programlanabilir yönlendirme yapısıyla çevrilen genel mantık, hafıza ve çoklayıcı (multiplexer) bloklarını içeren potansiyel olarak farklı programlanabilir mantık tiplerinden oluşur. Diziler, şekilde I/O ile gösterilen, programlanabilir giriş/çıkış bloklarıyla çevrilirler ve dış dünya bağlantısını sağlarlar.
FPGA için "Programlanabilir" terimi; fabrikasyon üretimi tamamlandıktan sonra, fonksiyonu tümleşik devreye programlayabilme yeteneğini ifade eder. Bu da sistem

tasarımcılarının fabrikasyon sürecinden sonra FPGA içerisindeki alanları tasarımlarına uygun kullanımına olanak sağlar. Tüm bunlar programlayabilme teknolojisinin sonuçlarıdır.



Şekil 3.1: Temel FPGA Yapısı

3.2.1. Programlama teknolojileri

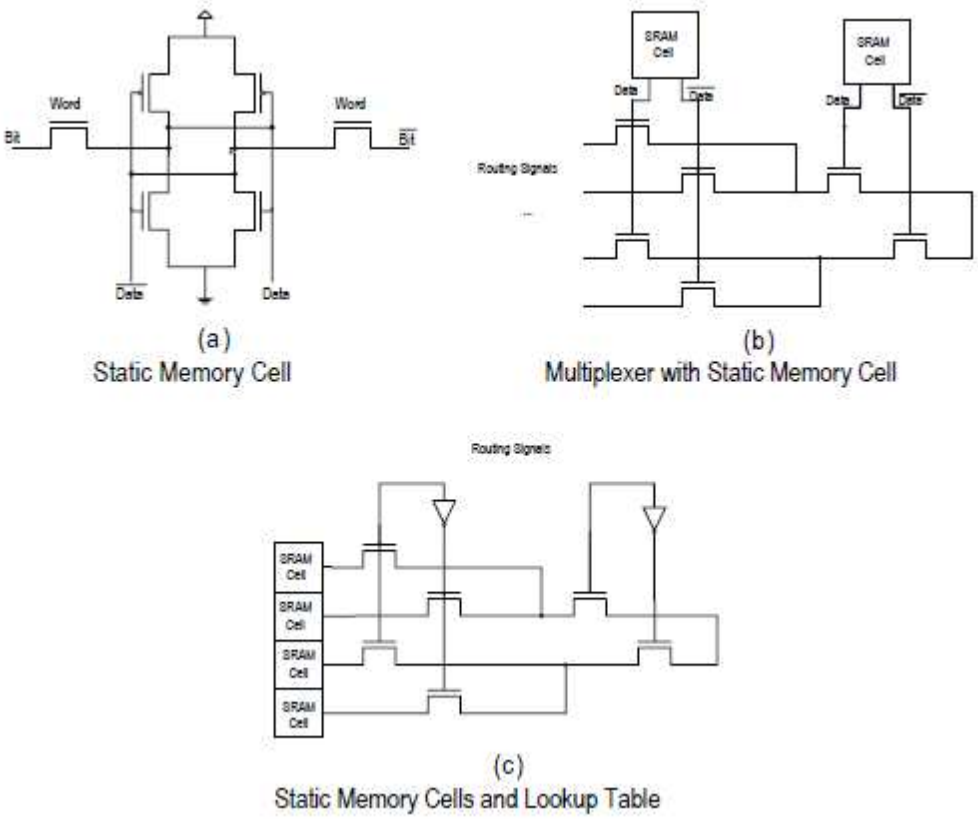
Her FPGA kendisine programlanabilme yeteneğini veren, programlanabilir anahtarları kontrol etmek için kullanılan belli başlı programlama teknolojisinden yararlanır. Programlanabilir mantık yapısına önemli derecede etki eden birkaç programlama teknolojisi vardır. EPROM [27], EEPROM [28], Flash [29], Static Memory (SRAM) [30] ve Anti-Fuse [31] bu yöntemlere örnek verilebilir.

Günümüzde sıklıkla kullanılan teknolojiler ise Flash, SRAM ve Karşıt-Sigorta'dır. Bu teknolojilerin ise birbirlerine göre avantaj ve dezavantajları bulunmaktadır.

3.2.1.1. SRAM programlama teknolojisi

Statik hafıza hücreleri, geniş yelpazede kullanılan ve Xilinx, Altere, Lattice gibi firmaların FPGA'lerinde bulunabilen, SRAM programlama teknolojisinin temelini oluşturan hücrelerdir.

Şekil 3.2.(a)'daki gibi statik hafıza hücreleri tüm FPGA'ye konfigüre edilebilirliği sağlamak amacıyla dağıtılırlar. SRAM'lerin iki önemli kullanımı bulunur. Çoğu, ara bağlantı sinyallerini süren çoklayıcı hatlarını seçme amaçlı kullanılırlar. Geriye kalan SRAM hücrelerinin çoğunluğu, tipik olarak SRAM tabanlı FPGA'lerde mantık fonksiyonlarını gerçekleştirmek için kullanılan LUT'larda veriyi depolamak için kullanılırlar. Şekil 3.2.(b) ve Şekil 3.2.(c) bu iki farklı yaklaşımı göstermektedir.



Şekil 3.2: Statik Hafıza Hücre Kullanımları

SRAM programlama teknolojisi iki öncelikli avantajından dolayı FPGA'ler için baskın bir yaklaşım olmuştur. Bunlar tekrar-programlanabilirlik ve standart CMOS işlem teknolojisinin kullanımıdır. Pratik açıdan bakılırsa, SRAM sonsuz kez

programlanabilir. Tasarlanan devre, sistemin beslemesi verildiğinde, tüm SRAM bitlerini başlangıç durumuna alır ve bitleri kullanıcı konfigürasyonuna göre konfigüre eder. Diğer programlama teknolojilerinin aksine, SRAM hücrelerinin kullanımı, standart CMOS'un işlem adımlarının ötesinde özel bir tümleşik devreye ihtiyaç duymaz. Sonuç olarak, SRAM tabanlı FPGA'ler en son CMOS teknolojisini kullanarak artırılan tümleşiklikten, yüksek hızlardan, daha az dinamik güç tüketiminden ve en az donanım boyutlarından faydalanabilirler. Fakat bunun yanında bu tip teknolojiye sahip FPGA'lerin bazı dezavantajları vardır:

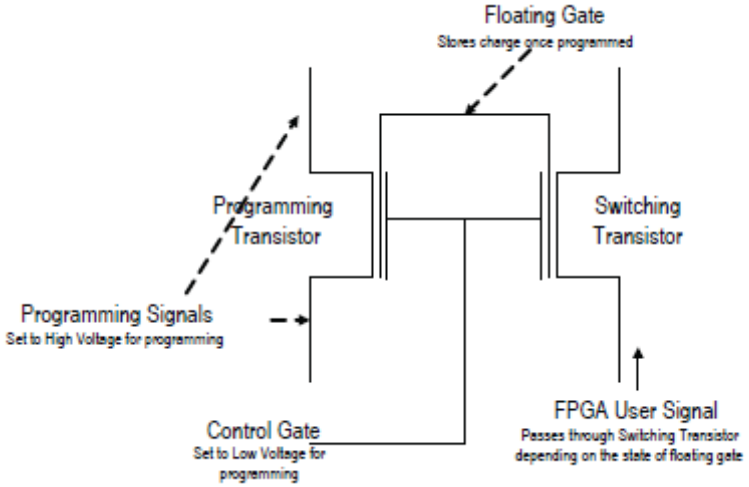
- Boyut: SRAM hücreleri 5 veya 6; sinyallerin birbirleriyle ara bağlantısını sağlayan programlanabilir öğeler de en azından 1 transistöre ihtiyaç duyar.
- Uçuculuk (Volatility): SRAM hücrelerinin kaybolabilirliği, besleme kesildiğinde mevcut konfigürasyonun korunabilirliğini sağlamak için harici bir depolama birimine gereksinim duyulmasına neden olur.
- Güvenlik: Konfigürasyon bilgisinin sistem açılışında yüklenmesi gerektiğinden, konfigürasyon bilgisinin engellenebilirliği yada çalınabilirliği söz konusu olabilir.
- Geçiş Transistörlerinin Elektriksel Sorunları: Geçiş transistörleri ideal anahtarlardan uzak, kayda değer iç dirence ve hissedilir kapasitif yüke sahiptirler.

3.2.1.2. FLASH/ E2PROM programlama teknolojisi

SRAM tabanlı mimarinin eksik kalan bazı adreslerine bir alternatif, yükü, transistörün üstünde yayılan bir kapı üzerine enjekte eden kayan kapı programlama teknolojisini kullanmaktır. Bu yaklaşım Flash veya EEPROM hafıza hücrelerinde kullanılır. Bu hücreler uçucu olmayan (non-volatile) özelliğe sahiptirler. Cihazın elektriğinin kesilmesi durumunda bilgiyi kaybetmezler.

Geçmişte, EEPROM hafıza hücreleri direk FPGA sinyallerini anahtarlamak için kullanılmamış, PLD tipli cihazlarda wired-AND fonksiyonlarını gerçekleştirmek için yaygın kullanılmışlardır [32]. Modern tümleşik devre fabrikasyon süreçleri ile hücreleri kayan kapı hücrelerini anahtar olarak kullanabilmek olası olmuştur. Flash hafıza hücreleri özellikle gelişmiş alan verimliliğinden dolayı kullanılırlar. Uçucu

olmayan hafıza tümleşik devreleri için Flash hafıza hücrelerinin geniş yelpazede kullanımı Flash üretim sürecinde geometrik olarak kalıcı küçülmelerden yararlanmayı sağlayacaktır. Şekil 3.3 Actel firmasının kullandığı Flash tabanlı yaklaşımı göstermektedir. Flash tabanlı mimari birçok büyük avantaj da sağlar. En



Şekil 3.3: Kayan Kapı (Floating Gate) Transistör

önemlisi uçucu olmayan hafıza yapısına sahip olmasıdır. Bu özellik SRAM tabanlı programlama teknolojisinde ihtiyaç duyulan harici kaynakları ortadan kaldırmaktadır. Bunun yanında Flash tabanlı programlama teknolojisi konfigürasyon verisinin yüklenmesini beklemeden elektrik verildiği anda fonksiyonunu gerçekleştirir.

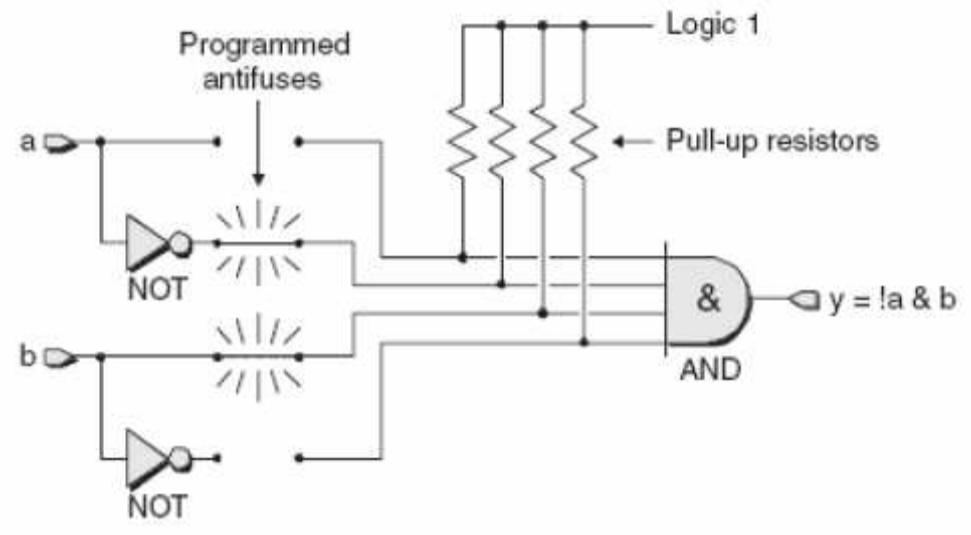
Flash tabanlı cihazların bir dezavantajı ise sınırlı sayıda programlanabilmeleridir. Diğer bir dezavantajları da standart olmayan CMOS işlemine ihtiyaç duymasıdır. SRAM tabanlı programlama teknolojisinde olduğu gibi transistör tabanlı anahtarların kullanımına bağlı olarak yüksek kapasite ve dirençten etkilenirler.

3.2.1.3. Karşıt-Sigorta programlama teknolojisi

SRAM tabanlı ve kayan kapı tabanlı programlama teknolojisine alternatif olarak geliştirilmiştir. Bu teknoloji, normal koşullarda çok yüksek direnç gösteren fakat programlanabilmesi için normalde bağlı fakat yakılınca düşük direnç gösteren yapılardan oluşmaktadır. Karşıt-Sigorta bağlantısı kalıcıdır.

Karşı-sigorta programlama teknolojisinin birincil avantajı az alan kullanımıdır. Diğer bir avantajı da diğer programlama teknolojilerine nazaran daha az direnç ve parazit kapasiteye sahip olmasıdır. Sigortaların az alan işgal edişi ve direnç, kapasite değerlerinin pratikte bulunan diğer teknolojilerdekinin aksine düşük oluşu cihaz başına daha fazla anahtar eklenmesini olanaklı kılar. Uçucu olmayan özelliği sayesinde cihaz programlandığında anında çalışabilmesine olanak sağlar. Sonuç olarak, programlama için FPGA'e bit akışının bir kez yapılmasından dolayı FPGA tasarımının güvenliğini artırır bir ortam oluşturur. Bu güvenliğin olanaklı kılınması ile cihaz programlandığında, programlama ara yüzünden erişimler de engellenmiş olunur [33].

En büyük eksikliği ise bir kez programlanabilir olmasıdır. Geliştirme ortamı oluşturmak için uygun bir seçim olmamaktadır. Şekil 3.4'te Karşı-Sigorta Programlama mimarisi görülmektedir.



Şekil 3.4: Karşıt-Sigorta Programlama Teknolojisi

3.2.1.4. Programlama teknolojilerinin karşılaştırılması

Yukarıda bahsi geçen programlama teknolojilerinin birbirlerine göre üstünlükleri ve eksikleri vardır. Tablo 3.1.'de bu teknolojilerin karşılaştırılması görülmektedir.

Tablo 3.1: Programlama Teknolojilerinin Karşılaştırılması

Özellik	SRAM	Flash	Karşıt-Sigorta
Uçuculuk	Evet	Hayır	Hayır
Tekrar Programlanabilme	Evet	Evet	Hayır
Alan	6 Transistör	1 Transistör	0 Transistör
Üretim İşlemi	Standart	Flash	Anti-Sigorta
Sistem İçi Programlanabilme	Evet	Evet	No
Anahtar Direnci	~500-1000 Ohm	~500-1000 Ohm	~20-100 Ohm
Anahtar Kapasitansı	~1-2 fF	~1-2 fF	<1-2 fF
Programlama Olanağı	%100	%100	>%90

3.3. Sonuç

Yukarıda bahsedilen üç programlama teknolojisi modern cihazlarda kullanılmakla birlikte SRAM tabanlı programlama teknolojisi en yaygın olarak kullanılır. İdeal bir programlama teknolojisi, standart CMOS işlemi kullanarak uçucu olmayan ve tekrar programlanabilen bir yapıya sahip olmalı ve düşük direnç ve düşük parazit kapasite sunmalıdır. Ne yazık ki günümüzde kullanılan bu teknolojilerden hiç biri tamamen tatmin edici değildir. SRAM tabanlı teknolojinin baskın olmasının birincil nedeni ise standart CMOS üretim işleminin bulunmasıdır. Bu sebeple de SRAM tabanlı teknolojinin, CMOS teknolojinin ileriye dönük geleceğinden dolayı sürdürülebilir olabileceği kabul edilebilir.

4. KABLOSUZ ALGILAYICI AĞLAR İÇİN SKIPJACK ŞİFRELME ALGORİTMASIN FPGA İLE GERÇEKLENMESİ

4.1. Giriş

Skipjack Algoritmasının FPGA ile gerçeklenmesinde Xilinx XC3S500E FPGA modeli ve VHDL dili kullanılmaktadır. Paralel (concurrent) veri işlemek için C yada C++ dili yerine daha alt seviye ve daha uygun dil olan VHDL tercih edilmektedir.

Algoritmanın çalışabilirliğini gösterebilmek amacıyla şifrelenecek ya da şifresi çözülecek verilerin, UART üzerinden PC Kullanıcı Arayüzü ile haberleşmesi sağlanmaktadır. Böylece kullanıcıya yönelik daha takip edilebilir/anlaşılır bir veri izleme ortamı oluşturulmaktadır.

VHDL gömülü yazılımı, şifreleme ve şifre çözme işlemlerinin yanında PC Kullanıcı Arayüzü ile iletişimi sağlayan UART iletişimi fonksiyonlarını da gerçekleştirmektedir. Bu iletişim fonksiyonlarını yerine getirmek, hiçbir şekilde şifreleme ve şifre çözme performansını etkilememektedir.

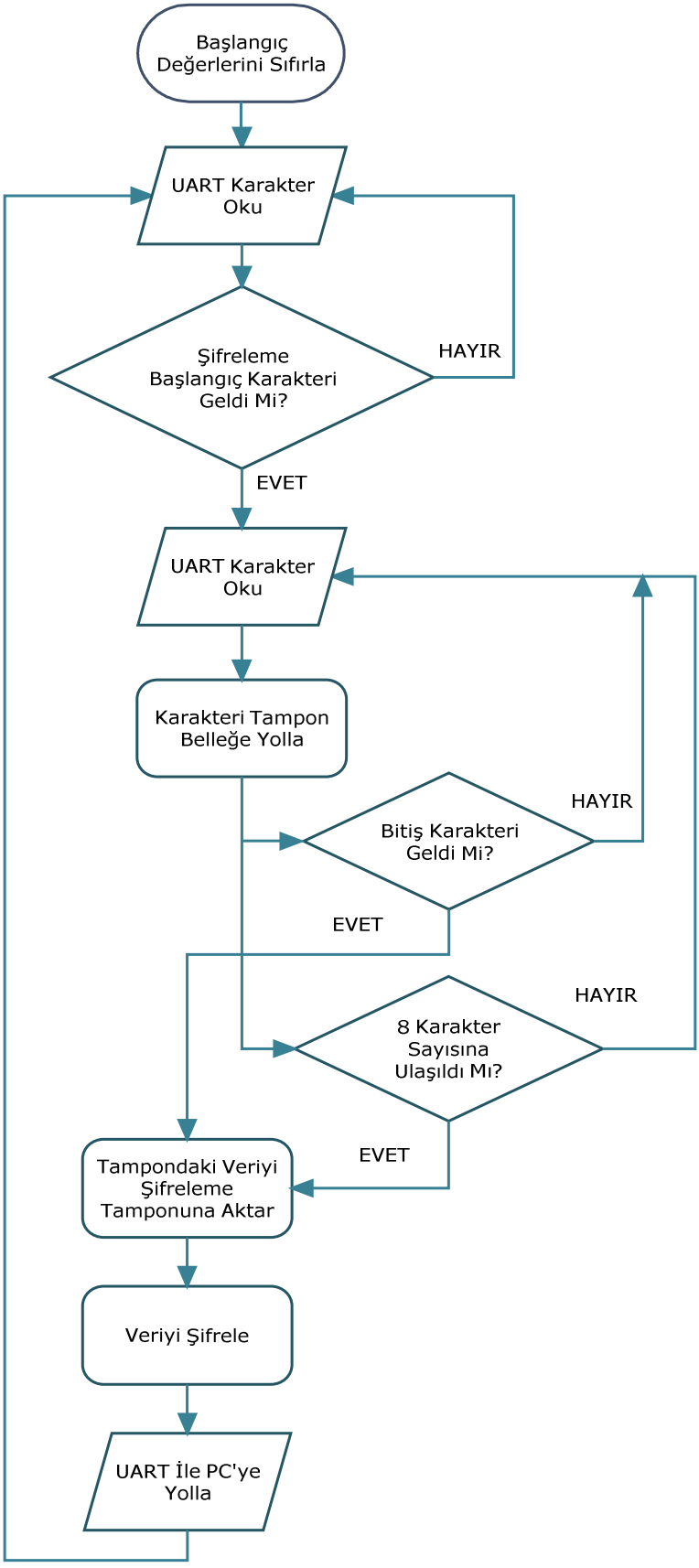
4.1.1. Skipjack VHDL yazılımının çalışma prensibi ve akış diyagramı

VHDL yazılımında Skipjack algoritmasını ilgilendiren kısımlar şifreleme ve şifre çözme olarak iki ana kısımdan oluşmaktadır. Şifreleme ve şifre çözme işlemlerinin her ikisi için de UART kullanılmakta ve PC Kullanıcı arayüzünden UART ile gelen veriler şifreleme ya da şifre çözme işlemine tabi tutulmaktadır. Bu işlemler bittikten sonra şifrelenmiş ya da şifresi çözülmüş veri yine UART kullanılarak PC Kullanıcı arayüzüne gönderilmektedir. Tek bir işlemde şifrelenecek ya da şifresi çözülecek veri uzunluğu 8 bayt yani 64 bittir. Şifreleme ve şifre çözme işlemi için ayrı bir başlangıç karakteri, işlemleri bitirmek için de her ikisi için ortak olan bir bitiş karakteri kullanılmaktadır. Donanım, şifrelenmiş ya da şifresi çözülmüş veri, PC kullanıcı

arayüzüne gönderildikten sonra tekrar yeni gelecek veriyi işleyebilecek durumda beklemektedir. Bu durumda şifrelenecek ya da şifresi çözülecek veri boyutunda bir sınır yoktur.

4.1.1.1. Skipjack VHDL şifreleme

Skipjack şifreleme işleminde tüm başlangıç değişkenleri sıfırlandıktan sonra UART'tan gelen veriler okunur. PC tarafından gönderilen karakter, şifreleme başlangıç karakteri ise bir sonraki adım olan şifrelenecek 8 karakteri okuma işlemine başlanır. Bu adımda UART'tan okunan veriler bir tampon belleğe alınır ve bitiş karakteri beklenir. Eğer herhangi bir sebeple bitiş karakteri gelmez ise, şifrelenebilecek en fazla karakter sayısı olan 8 değerine ulaşıldığında bu adım tamamlanır. Eğer bitiş karakteri gelir ise bu adım yine sonlandırılarak diğer adıma geçilir. Bu yeni adımda tampon bellekteki veri, şifreleme tampon belleğine aktarılır. Şifrelenecek karakter sayısı 8 değerinden küçük olsa bile şifreleme işlemi 8 bayt üzerinden yapılır. Çünkü Skipjack 64 bit yani 8 bayt'lık bir şifreleme algoritmasıdır. 8 karakterden küçük şifreleme işlemleri, şifrelenecek karakterlerin dışındaki tampon bellek alanı 0 kabul edilerek yapılır. Şifreleme işlemi tamamlandıktan sonra şifreli veri UART üzerinden PC kullanıcı arayüzüne gönderilir. Şekil 4.1.'de tüm bu adımların akış diyagramı gösterilmektedir.

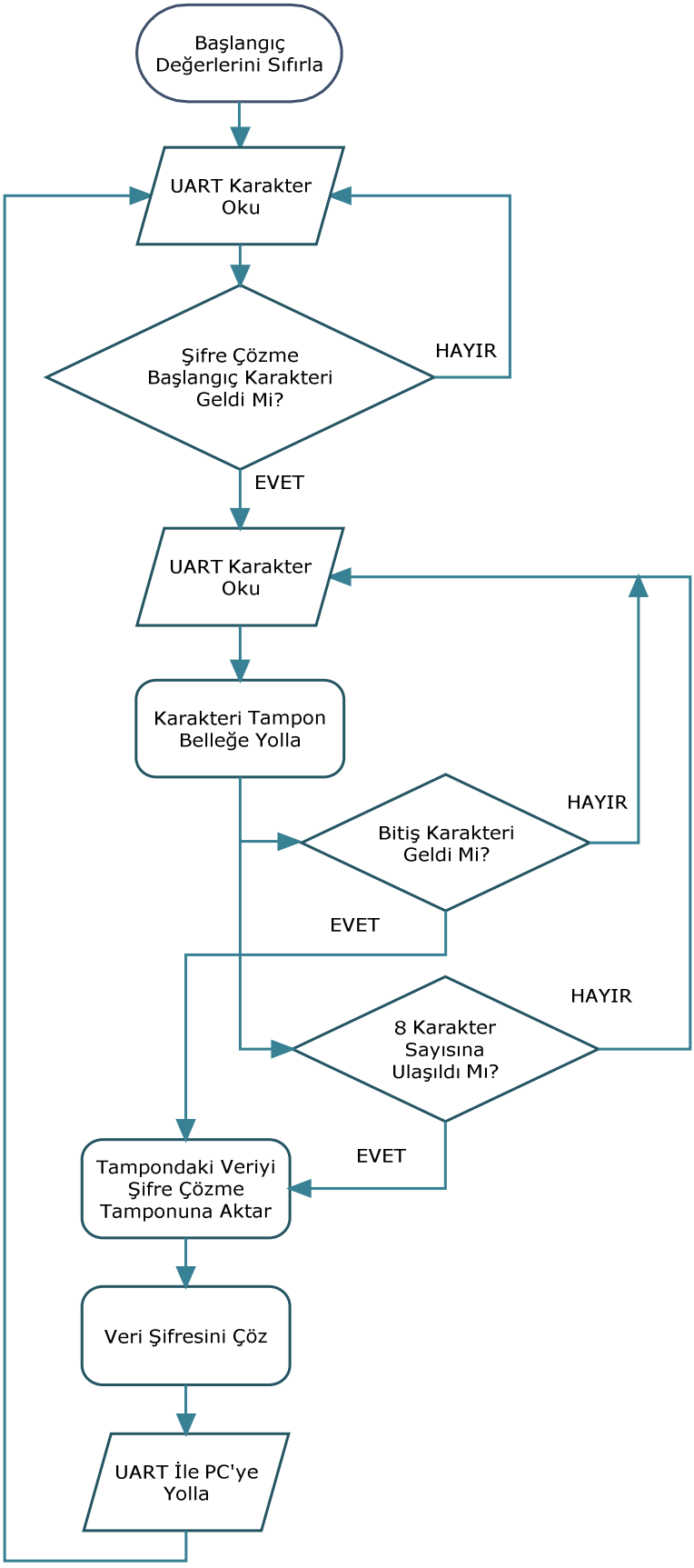


Şekil 4.1: Skipjack Şifreleme Akış Diyagramı

4.1.1.2. Skipjack VHDL şifre çözme

Skipjack şifre çözme işlemi, şifreleme işlemi ile aynı akış diyagramına sahiptir. Aradaki fark şifre çözme başlangıç karakterinin farklı olması ve şifreleme algoritması yerine şifre çözme algoritması kullanılmasıdır.

Skipjack şifre çözme işleminde tüm başlangıç değişkenleri sıfırlandıktan sonra UART'tan gelen veriler okunur. PC tarafından gönderilen karakter, şifre çözme başlangıç karakteri ise bir sonraki adım olan şifresi çözülecek 8 karakteri okuma işlemine başlanır. Bu adımda UART'tan okunan veriler bir tampon belleğe alınır ve bitiş karakteri beklenir. Eğer herhangi bir sebeple bitiş karakteri gelmez ise, şifresi çözülecek en fazla karakter sayısı olan 8 değerine ulaşıldığında bu adım tamamlanır. Eğer bitiş karakteri gelir ise bu adım yine sonlandırılarak diğer adıma geçilir. Bu yeni adımda tampon bellekteki veri, şifre çözme tampon belleğine aktarılır. Şifresi çözülecek karakter sayısı 8 değerinden küçük olsa bile şifre çözme işlemi 8 bayt üzerinden yapılır. Çünkü Skipjack 64 bit yani 8 bayt'lık bir şifre çözme algoritmasıdır. 8 karakterden küçük şifre çözme işlemleri, şifresi çözülecek karakterlerin dışındaki tampon bellek alanı 0 kabul edilerek yapılır. Şifre çözme işlemi tamamlandıktan sonra şifresi çözülen veri UART üzerinden PC kullanıcı arayüzüne gönderilir. Şekil 4.2.'de tüm bu adımların akış diyagramı gösterilmektedir.



Şekil 4.2: Skipjack Şifre Çözme Akış Diyagramı

4.2. Sonuç

Bu uygulamada veri akışını daha kolay izlenebilir/anlaşılır kılmak amacıyla UART kullanımı, PC ile haberleşmeyi sağlayarak kullanıcıya yönelik görsel bir ortamın alt yapısını sağlamaktadır. FPGA'nin port çıkışlarının kullanılması durumdaki şifrelenen ya da şifresi çözülen verinin izlenebilirliğinin zor olması olasılığı, bu uygulama ile ortadan kaldırılmaktadır.

KAA uygulamalarına yönelik durumlarda sadece FPGA port çıkışları ve girişleri kullanılacak ve bu durumda gömülü yazılımdan UART kısmı çıkarılacaktır. Bu durumu sağlamanın yazılımsal olarak hiçbir zorluğu yoktur.

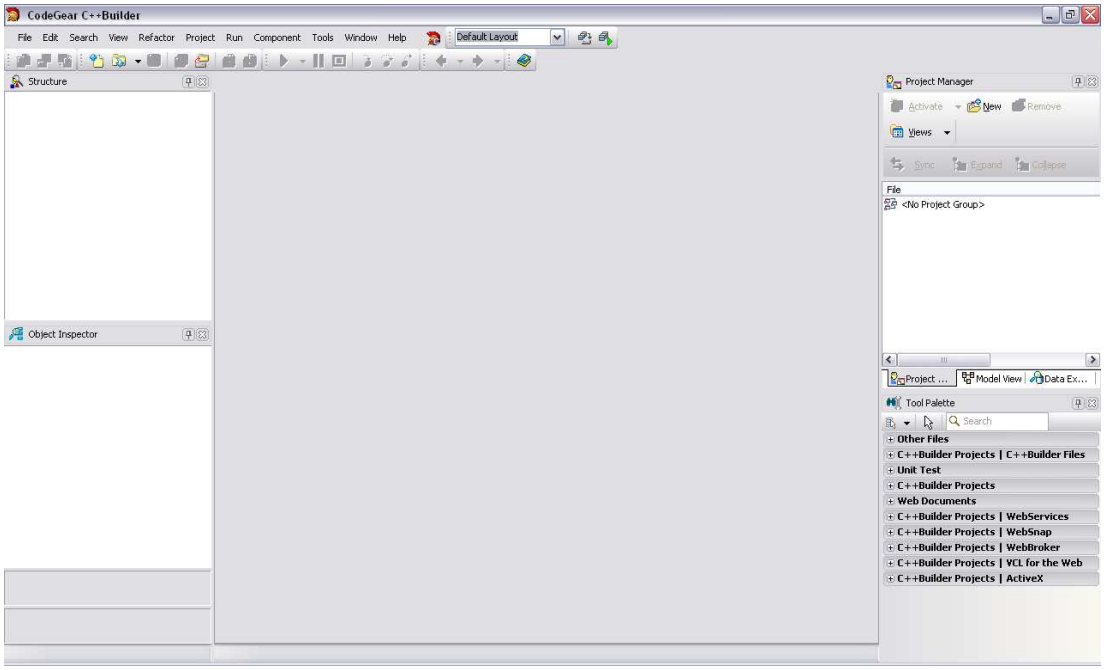
5. FPGA TABANLI SKIPJACK ALGORİTMASININ KAA UYGULAMA ÖRNEKLERİ

5.1. Giriş

Bu çalışmada, FPGA tabanlı Skipjack şifreleme algoritmasının KAA'larda kullanılmak üzere Xilinx FPGA bulunan bir geliştirme kartında Xilinx ISE arayüzü kullanılarak çalışması ve performansı değerlendirilmektedir. Elde edilen benzetim sonuçlarının donanımsal ortamda da elde edilmesi amaçlanmaktadır.

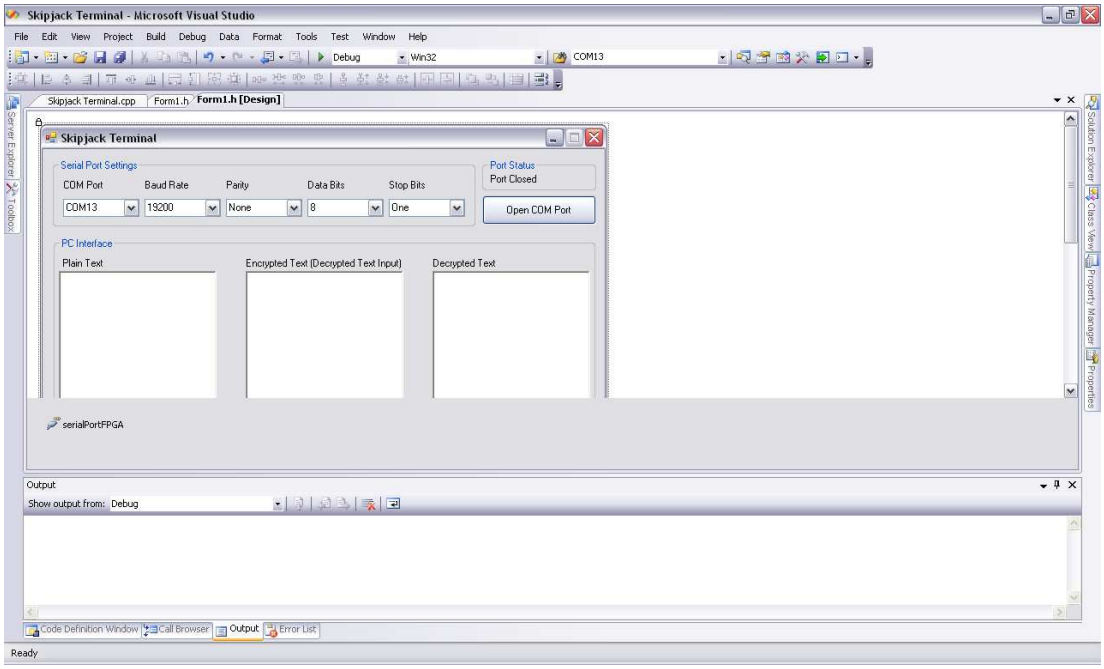
5.2. FPGA Tabanlı Skipjack Algoritmasının KAA Uygulama Örnekleri

Skipjack algoritmasının PC ortamında çalışabilirliğini gösterebilmek amacıyla Codegear C++ 2009 PC yazılımından yararlanılmaktadır. Skipjack Specification dokümanı doğrultusunda yazılan Skipjack algoritması yine aynı dokümanın içinde bulunan şifreleme ve şifre çözme örnek veri kümesi kullanarak bu yazılımda Console uygulaması olarak denenmekte ve sonuçlar aynı dokümanda bulunan veri kümesi ile karşılaştırılmaktadır. Skipjack uygulamasını VHDL dilinde yazabilmek için tercih edilen FPGA modelinin yazılım geliştirme ortamı olan Xilinx ISE Foundation kullanılmaktadır. Xilinx ISE Foundation programında, Codegear C++ 2009 ile yazılan C kodları VHDL diline uyarlanarak yazılmıştır. Şekil 5.1.'de görülen uygulama geliştirme ortamı ile Skipjack Console uygulaması gerçekleştirilmektedir.



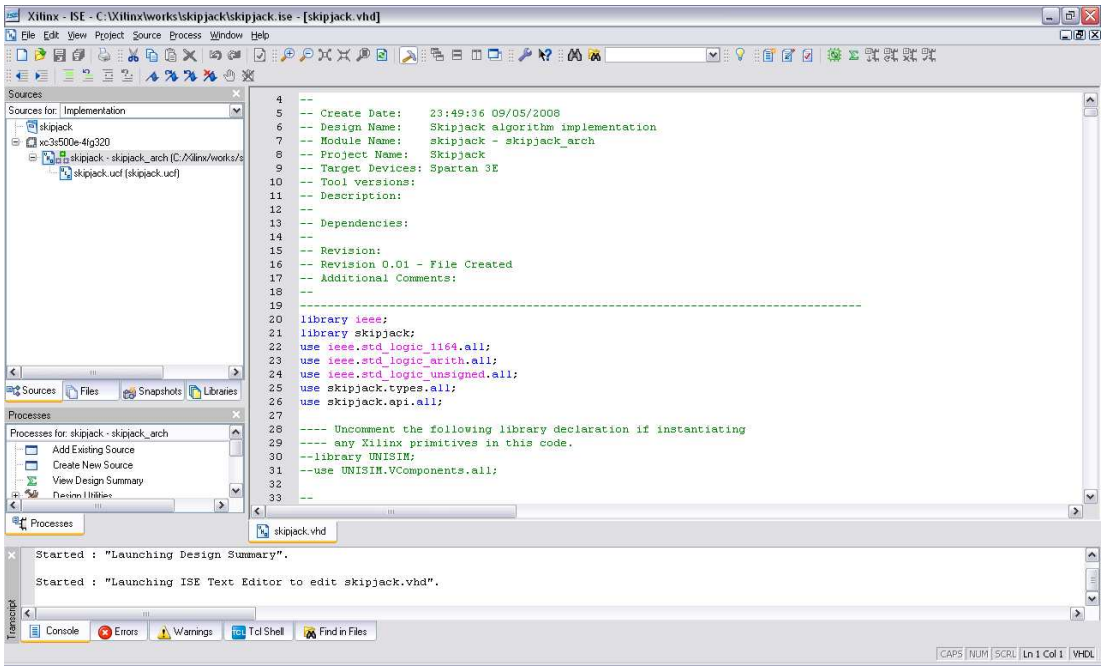
Şekil 5.1: CodeGear C++ 2009 Kullanıcı Arayüzü

Şekil 5.2.'de görülen uygulama geliştirme ortamı ile Skipjack Form uygulaması gerçekleştirilmektedir.



Şekil 5.2: Visual Studio C++ 2008 PC Arayüzü

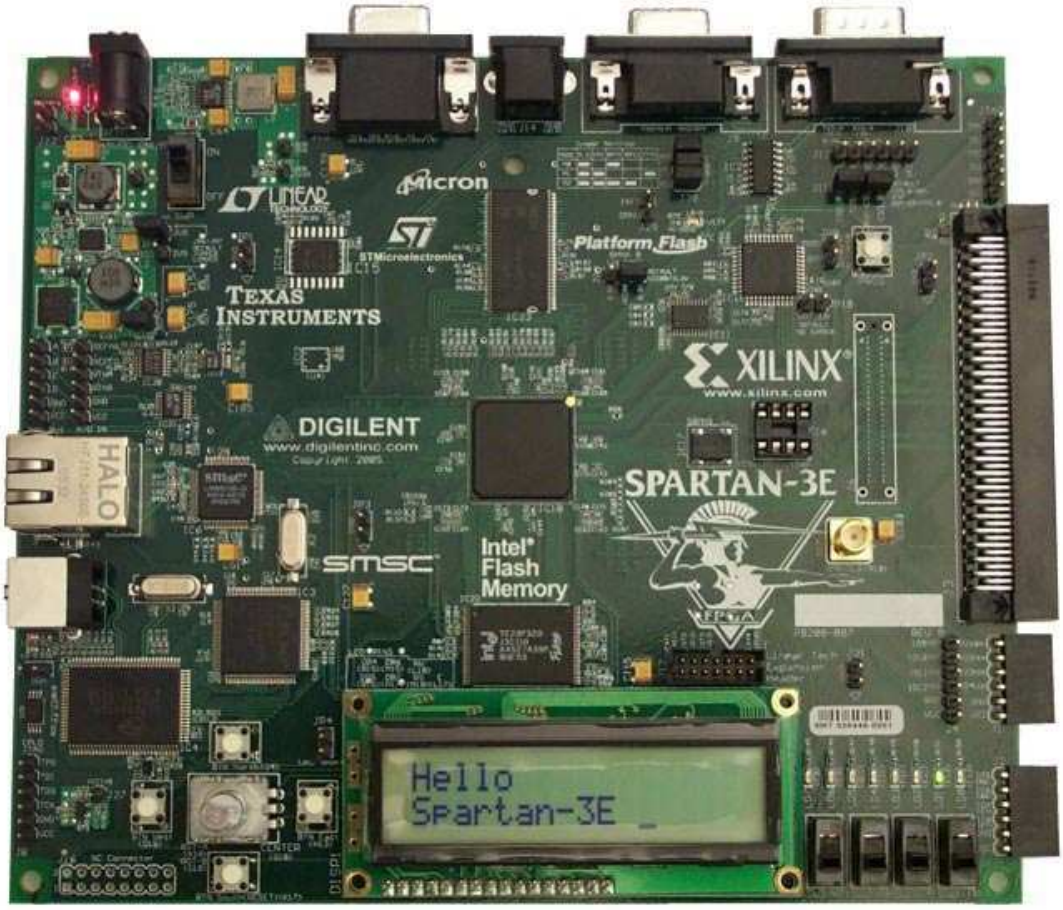
Şekil 5.3.'te görülen uygulama geliştirme ortamı ile FPGA gömülü yazılımı VHDL dili kullanarak yazılmıştır. Benzetim sonuçları yine aynı arayüz kullanarak elde edilmektedir.



Şekil 5.3: Xilix ISE Foundation

Skipjack algoritmasının gerçeklenmesini sağlamak amacıyla yazılacak VHDL ve C kodları Skipjack Specification göz önünde bulundurulacak şekilde yazılmıştır.

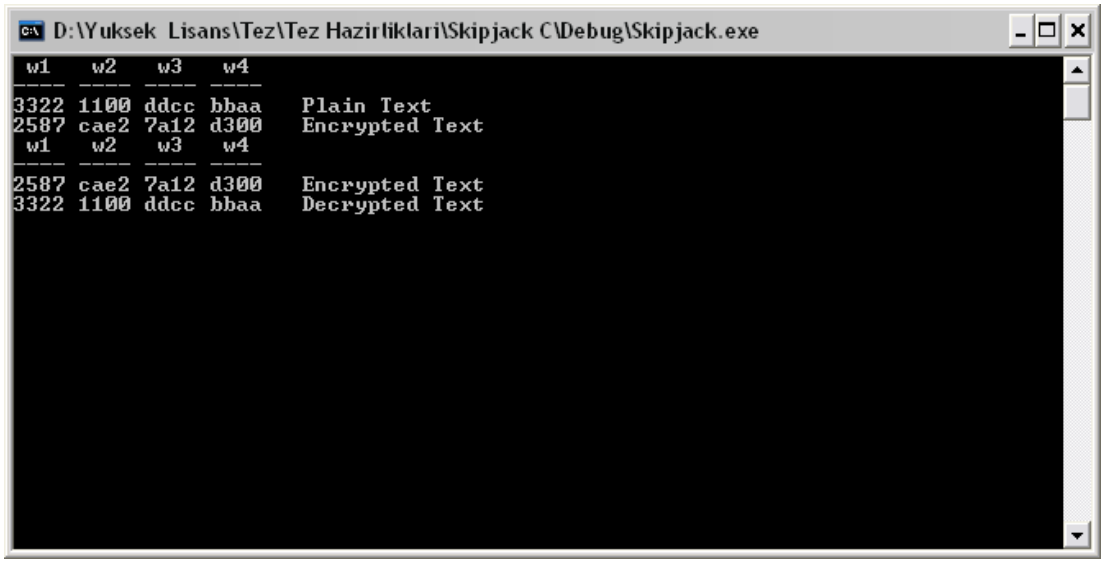
Donanım olarak, üzerinde Xilinx XC3S500E FPGA bulunan Şekil 5.4.'te görülen Digilent firmasının Spartan 3E Starter Board'u seçilmiştir. FPGA programlanması USB üzerinden gerçekleştirilmektedir. RS-232 seri port çıkışı, şifrelenmesi gereken veri bloğunu PC'den geliştirme kartına ve şifrelenmiş veri bloğunu da geliştirme kartından PC ortamına aktarmada kullanılmaktadır.



Şekil 5.4: Xilinx Spartan 3E Starter Board

5.3. Skipjack CodeGear C++ Console Uygulaması Arayüzü

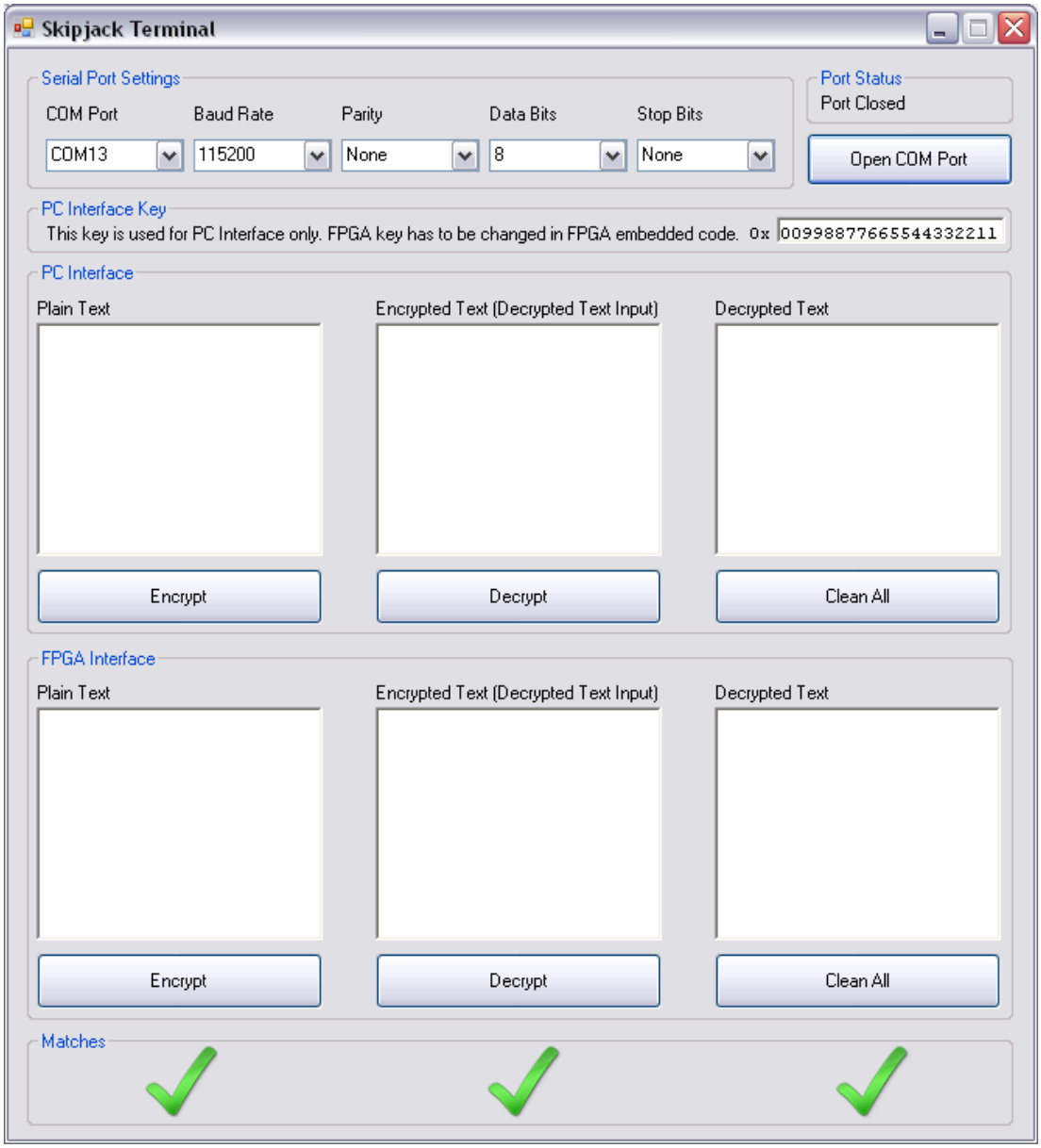
Şekil 5.5.'deki Console uygulamasında, Skipjack Specification dokümanında yer alan 64 bit'lik veri (Plain Text) girilerek şifreli veri (Encrypted Text) oluşturulmaktadır. Oluşan şifreli veri tekrar şifre çözme işlemine tabi tutulup başta girilen 64 bit'lik veri (Decrypted Text) birebir elde edilmektedir. Skipjack algoritmasının C kodlarının çalıştığının görülebilmesi için bu küçük Console uygulamasından yararlanılmaktadır.



Şekil 5.5: Skipjack Console Uygulaması

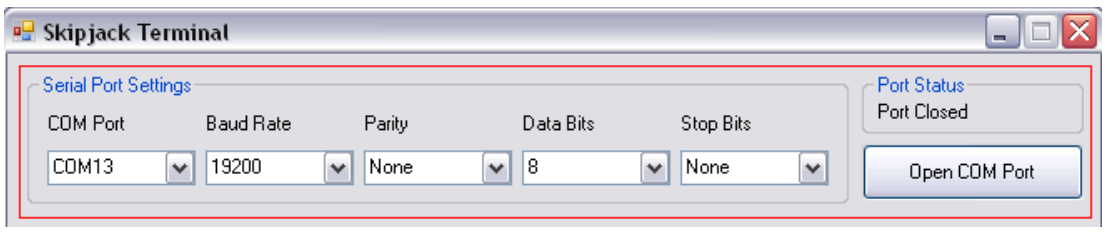
Şekil 5.6.'da görülen Skipjack Terminal adlı Kullanıcı arayüzü ile hem PC tarafında hem de PFGA tarafında şifreleme ya da şifre çözme işlemi yapılabilmektedir. PC ile FPGA board arasındaki veri iletişimi UART ile sağlanmaktadır. UART hızı Kullanıcı arayüzü ile değiştirilebilmektedir. Aynı zamanda PC ve FPGA board tarafından birbirinden bağımsız şifrelenen ya da şifresi çözülen veriler doğrulamayı sağlamak amacıyla birbirleriyle karşılaştırılabilmektedir.

Bu amaçla PC ve FPGA arayüzleri şekilde görüldüğü gibi birbirlerinden ayrılmışlardır. Terminal yazılımı dili ve arayüzde kullanılan tüm nesnelerin dili araştırmacılara hitap edebilecek bir durum oluşabileceği düşüncesiyle İngilizce yazılmıştır.



Şekil 5.6: Skipjack PC Kullanıcı Arayüzü

Şekil 5.7.'de PC Kullanıcı arayüzünün üst kısımda bulunan, PC ile FPGA board arasında veri iletişimini sağlayan UART'ın ayarlarını değiştirebilmeyi ve seri portun açılmasını veya kapanmasını sağlayan kısım görülmektedir.



Şekil 5.7: UART Port Ayarları ve Port Kontrol

COM Port etiketli seçim kutusunda, yazılımın çalıştırıldığı PC'de bulunan tüm portlar gözükmekte ve FPGA board ile iletişim kurulacak seri port seçilebilmektedir. Varsayılan port adı COM13'tür.

Baud Rate etiketli seçim kutusunda seri iletişim baud hızı seçilebilmektedir. Varsayılan baud hızı 19200'dür.

Parity etiketli seçim kutusuyla seri iletişim parity (eşlik) seçimi yapılabilmektedir. Varsayılan durum olarak parity kullanılmamaktadır.

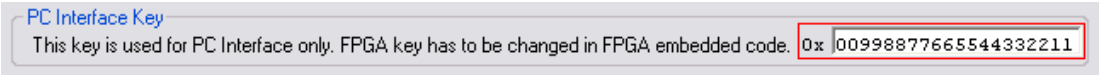
Data Bits etiketli seçim kutusuyla kaç bit'lik iletişim yapılacağı seçilebilmektedir. Varsayılan Data Bits değeri 8'dir.

Stop Bits etiketli seçim kutusuyla ne kadar stop biti kullanılacağı seçilmektedir. Varsayılan Stop Bits değeri 1'dir.

Port Status etiketli grup kutusunun içinde portun açıldığındaki ya da kapandığındaki durumları belirtilmektedir. Port durumu program açılış durumda daima kapalıdır.

Open COM Port etiketli buton ile port açılması ya da kapanması sağlanmaktadır. Port açıldığında buton etiketi "Close COM Port" olarak ve Port Status "Port Opened" olarak değişmektedir. Port kapatıldığında buton etiketi "Open COM Port" olarak ve Port Status "Port Closed" olarak değişmektedir.

Şekil 5.8.'de, PC Interface etiketli grup kutusu içinde gerçekleştirilecek şifreleme ve şifre çözme işlemleri için kullanılacak 20 karakter uzunluğundaki anahtar belirleme metin kutusu görülmektedir. Şifrelemede kullanılacak anahtar, ilgili haneler rakam veya A-F arası harf olmak şartıyla değiştirilerek belirlenmektedir. 20 karakter uzunluğu onaltılık sistem girişi içindir. Mevcut anahtar uzunluğu 10 bayt'a tekabül etmektedir.



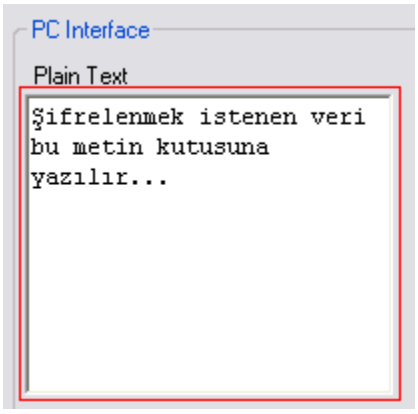
Şekil 5.8: PC Interface Key Belirleme Metin Kutusu

FPGA tarafından gerçekleştirilen işlemlerde kullanılan anahtar ise VHDL ile yazılmış kod üzerinde kalıcı (hard-coded) olarak değiştirilebilmektedir.

Şekil 5.9.'da Skipjack algoritması ile şifrelemede ya da şifre çözmede kullanılan metin kutuları görülmektedir. Şifreleme ve şifre çözme işlemleri PC kullanıcı arayüzü tarafından çözülerek ilgili metin kutularında gösterilmektedir. FPGA tarafının işlemlerini gerçekleştiren "FPGA Interface" etiketli grup kutusunda bulunan metin kutuları da aynı özelliklere sahiptir ve FPGA tarafından şifrelenen ya da şifresi çözülen verileri ilgili metin kutularında göstermektedirler. Bu metin kutuları, hem PC arayüzü için hem de FPGA arayüzü için 3'er tanedir.

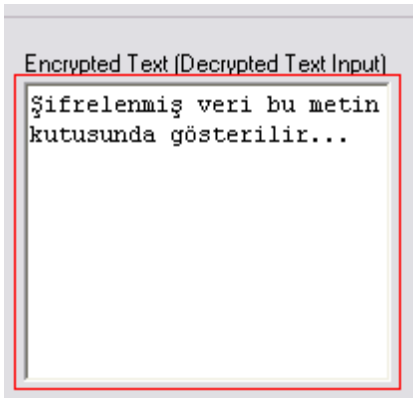
Şekil 5.9: Skipjack Şifreleme/Şifre Çözme PC Interface

Şekil 5.10.'da Şifrelenmek istenen verinin girileceği metin kutusu görülmektedir. Bu metin kutusuna istenilen karakter katarı girilebilir. Aynı durum "FPGA Interface" grup kutusu içerinde bulunan Plain Text etiketli metin kutusu için de geçerlidir.



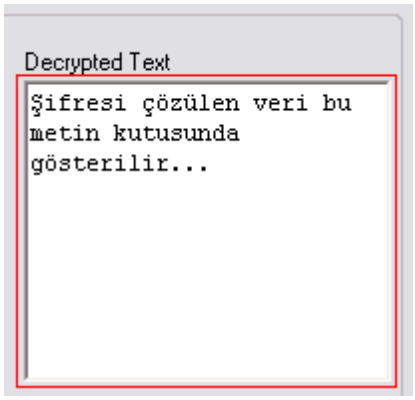
Şekil 5.10: Plain Text Metin Kutusu

Şekil 5.11.'de Şifrelenmiş verinin gösterileceği metin kutusu görülmektedir. Aynı zamanda bu metin kutusu, tez uygulamasına yönelik şifresi çözülmek istenen verinin metin kutusu olarak da kullanılmaktadır. Bu metin kutusu tüm ascii karakterleri basmayı desteklemediğinden, veriler onaltılık sistemde gösterilmektedir. Aynı durum "FPGA Interface" grup kutusu içerinde bulunan Encrypted Text etiketli metin kutusu için de geçerlidir.



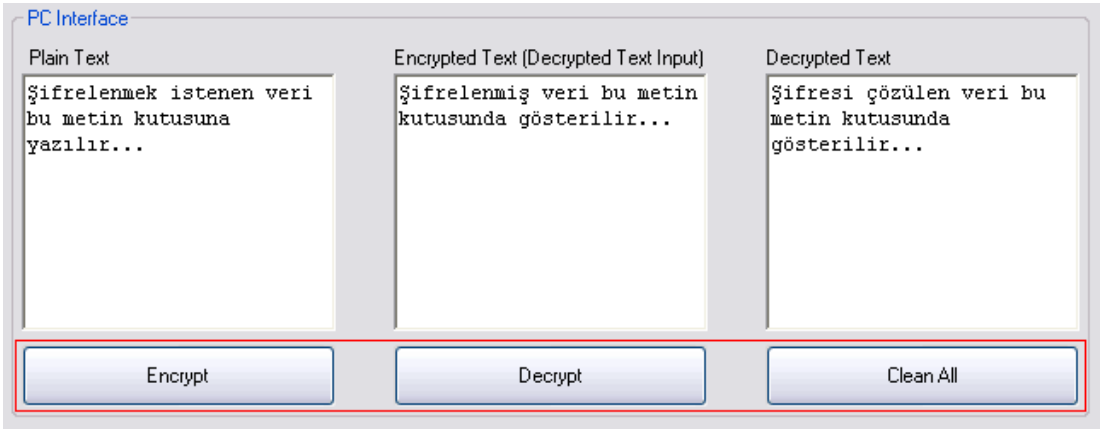
Şekil 5.11: Encrypted Text Metin Kutusu

Şekil 5.12.'de Şifresi çözülen verinin gösterileceği metin kutusu görülmektedir. Bu metin kutusunda Encrypted Text (Decrypted Text Input) etiketli metin kutusundaki verinin şifresi çözülmüş hali gösterilmektedir. Veri şifrelendikten sonra şifrelenmiş veri bozulmadığı sürece, şifrelenmesi istenen veri (Plain Text etiketli metin kutusuna girilen veri) aynen elde edilir. Aynı durum "FPGA Interface" grup kutusu içerinde bulunan Decrypted Text etiketli metin kutusu için de geçerlidir.



Şekil 5.12: Decrypted Text Metin Kutusu

Şekil 5.13.'te Hem PC arayüzünde hem de FPGA arayüzünde aynı fonksiyona sahip işlem butonları görülmektedir.



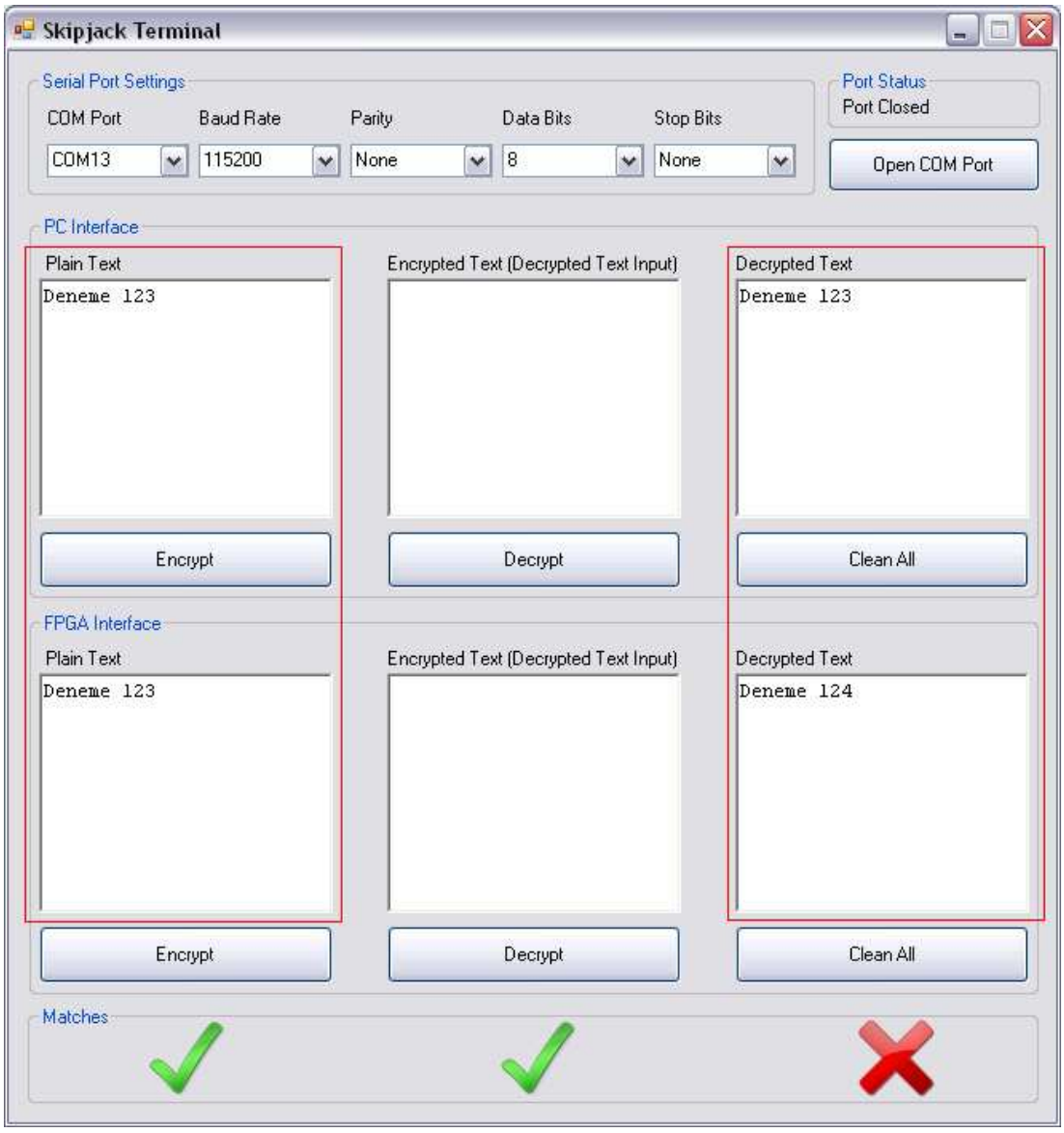
Şekil 5.13: İşlem Butonları

Encrypt etiketli buton ile "Plain Text" metin kutusuna girilen veri şifrelenmektedir. Şifreli veri "Encrypted Text (Decrypted Text Input)" metin kutusunda gösterilmektedir.

"Clean All" ile ilgili grup kutusu içindeki tüm metin kutuları içeriği silinmektedir.

Decrypt etiketli buton ile "Encrypted Text (Decrypted Text Input)" metin kutusuna girilen verinin ya da bu metin kutusunda şifreleme işlemi sonucunda oluşan verinin şifre çözme işlemi gerçekleştirilir. Şifresi çözülen veri "Dencrypted Text" metin kutusunda gösterilmektedir.

Şekil 5.14.'te PC arayüzü tarafından şifrelenen ya da şifresi çözülen verilerin FPGA arayüzü tarafından şifresi çözülen veriler ile birebir olup olmadığını belirtmek amacıyla kullanılan, Matches etiketli grup kutusu içinde bulunan resim formatlı işaretler görülmektedir. PC Interface metin kutuları ile FPGA interface metin kutuları şekilde görüldüğü gibi birebir karşılaştırılır. Eğer metin kutuları verileri aynı ise yeşil renkli tik resmi, değil ise kırmızı renkli X resmi gösterilir.



Şekil 5.14: Karşılaştırma Bildirimleri

5.4. Donanım Arayüzü

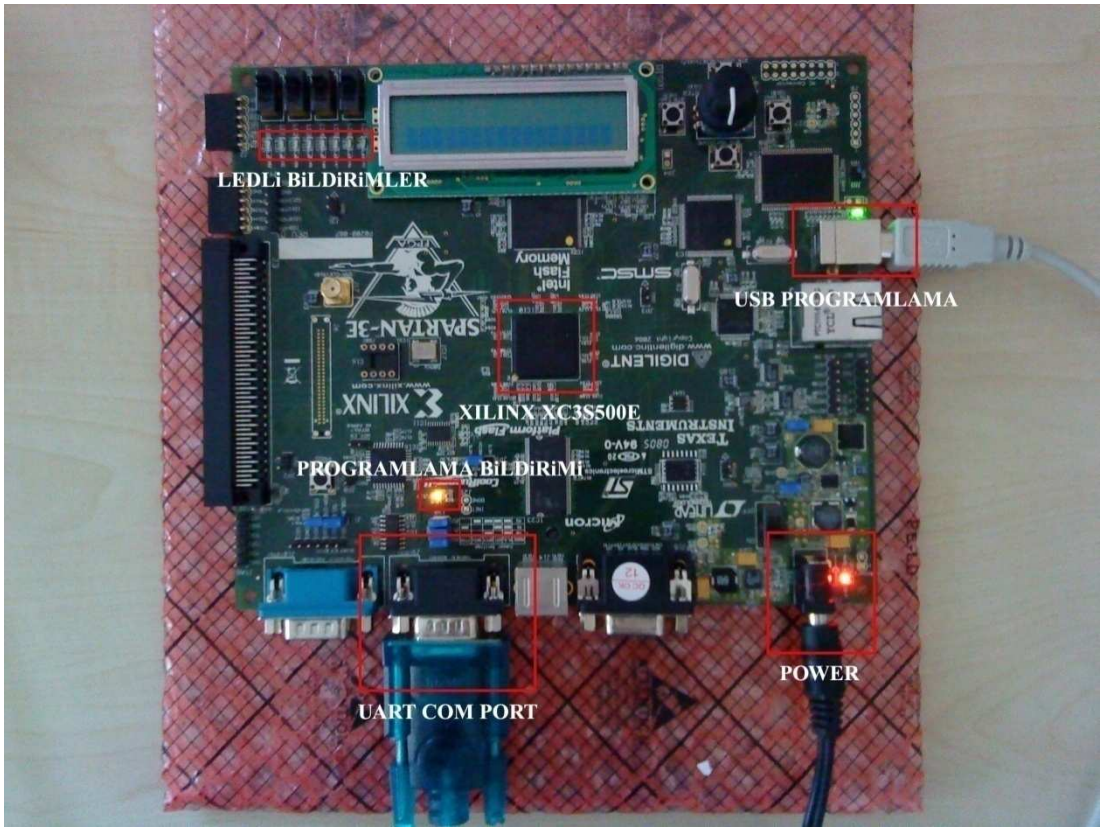
FPGA board ile PC Kullanıcı arayüzü arasındaki veri iletimini sağlamak amacıyla Şekil 5.15.'de görülen ve Xilinx Spartan 3E Starter Board üzerinde bulunan UART COM Port kullanılmaktadır. Bu port üzerinden, şifrelenmek istenen veri veya şifrelenmiş veri iletimi yapılmaktadır. Bu port'un veri iletim hızı hiçbir şekilde şifreleme ve şifre çözme işlemleri hızını etkilememektedir.

Şekil 5.15: UART COM Port

Şekil 5.16.'da şifreleme, şifre çözme, UART TX tampon dolu ve UART RX tampon dolu bildirimlerini göstermek amacıyla kullanılan ledler görülmektedir. Ledler, şekle göre soldan sağa doğru ilgili bildirim olduğunda sırasıyla yakılırlar.

Şekil 5.16: Ledli Bildirimler

Şekil 5.17.'de Digilent FPGA board ve fonksiyonel bileşenleri görülmektedir.



Şekil 5.17: Digilent FPGA Board ve Donanım Arayüzleri

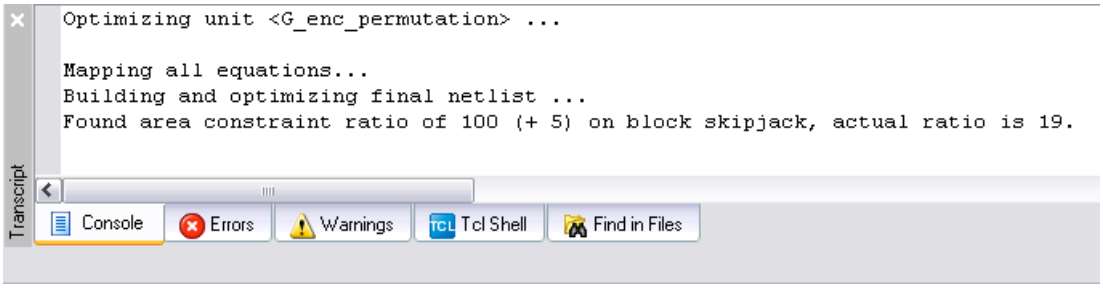
5.5. Deneysel Çalışmalar

Bu alt başlıkta tez çalışmasında kullanılan Skipjack algoritması ile internet ortamında örnek olarak bulunan bir başka Skipjack algoritmasının [34] karşılaştırmalı olarak benzetim performans analizleri sunulmaktadır. İnternet ortamında bulunan örnek Skipjack algoritması, VHDL kullanılarak yazılması sebebiyle birebir karşılaştırma için uygundur. Karşılaştırma yapılırken Xilinx ISE Foundation sonuçları kullanılmaktadır. Örnek algoritmanın şifre çözme algoritması bulunmadığından sadece şifreleme algoritmaları karşılaştırılmaktadır.

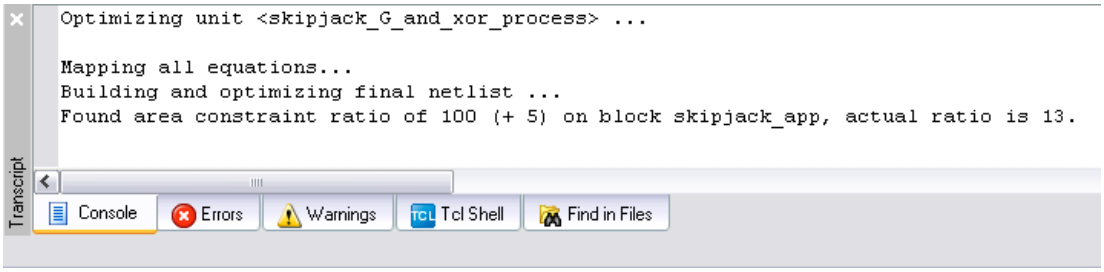
Aynı zamanda, tez çalışmasında kullanılan Kullanıcı arayüzü ve Donanım arayüzü tarafından şifrelenen ya da şifresi çözülen veriler için örnek uygulama sunulmaktadır. Görsel olarak sunulan örnek uygulamada, metin kutularına girilen karakterler Kullanıcı arayüzünde ve Donanım arayüzünde kullanılmak üzere rastgele seçilen karakterlerdir.

5.5.1. FPGA Kısıtlama Oranı (Constraint Ratio) karşılaştırması

Şekil 5.18. ve Şekil 5.19.'daki sonuçlar kıyaslanırsa, tez çalışmasının kısıtlama oranının örnek çalışmaya göre yaklaşık % 31,57 daha düşük olduğu görülecektir. FPGA'ler için kısıtlama oranının artması gömülü yazılım oranının artması anlamına gelmektedir. Bu durumda, tez çalışması diğer örnek çalışmaya göre FPGA'de yaklaşık % 6 daha az yer kaplamaktadır.



Şekil 5.18: Örnek Çalışma Kısıtlama Oranı



Şekil 5.19: Tez Çalışması Kısıtlama Oranı

5.5.2. FPGA Tasarım Özeti (Design Summary) karşılaştırması

Şekil 5.20. ve Şekil 5.21.'deki sonuçlar yüzdelik olarak kıyaslanırsa tez çalışması tasarım özeti sonuçlarının örnek çalışmaya göre daha başarılı olduğu görülmektedir. Bu sonuçlar sayesinde kısıtlama oranı da düşürülmüş olmaktadır.

Device Utilization Summary				[-]
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Flip Flops	161	9,312	1%	
Number of 4 input LUTs	1,514	9,312	16%	
Logic Distribution				
Number of occupied Slices	776	4,656	16%	
Number of Slices containing only related logic	776	776	100%	
Number of Slices containing unrelated logic	0	776	0%	
Total Number of 4 input LUTs	1,514	9,312	16%	
Number of bonded IOBs				
Number of bonded	212	232	91%	
Number of BUFGMUXs	1	24	4%	

Şekil 5.20: Örnek Çalışma Tasarım Özeti

Device Utilization Summary				[-]
Logic Utilization	**Used**	**Available**	**Utilization**	**Note(s)**
Number of Slice Flip Flops	271	9,312	2%	
Number of 4 input LUTs	1,399	9,312	15%	
Logic Distribution				
Number of occupied Slices	780	4,656	16%	
Number of Slices containing only related logic	780	780	100%	
Number of Slices containing unrelated logic	0	780	0%	
Total Number of 4 input LUTs	1,399	9,312	15%	
Number of bonded IOBs				
Number of bonded	132	232	56%	
Number of BUFGMUXs	1	24	4%	

Şekil 5.21: Tez Çalışması Tasarım Özeti

5.5.3. FPGA güç tüketimi karşılaştırılması

Şekil 5.22. ve Şekil 5.23.'deki tablo Xilinx ISE Foundation yazılımının sunduğu yardımcı bir program olan Xilinx XPower Analyzer ile oluşturulmaktadır. Bu yardımcı yazılım, FPGA üzerindeki gömülü yazılımın ne kadar güç tüketimi yapacağını hesaplamakta kullanılmaktadır. Elde edilen benzetim verileri gerçek değerlere çok yakındır.

Her iki çalışmanın 8 Mhz'deki sonuçları tablolardaki gibi olmak üzere, tez çalışması toplam güç tüketimi (Total Power) örnek çalışmaya göre yaklaşık % 3,53 daha azdır.

Name	Value	Used	Total Available	Utilization (%)
Clocks	0.00049 (W)	1	---	---
Logic	0.00123 (W)	1514	9312	16.3
Signals	0.00678 (W)	1544	---	---
IOs	0.00873 (W)	212	232	91.4
Total Quiescent Power	**0.08120 (W)**			
Total Dynamic Power	**0.01723 (W)**			
Total Power	**0.09842 (W)**			
Junction Temp	**27.6 (degrees C)**			

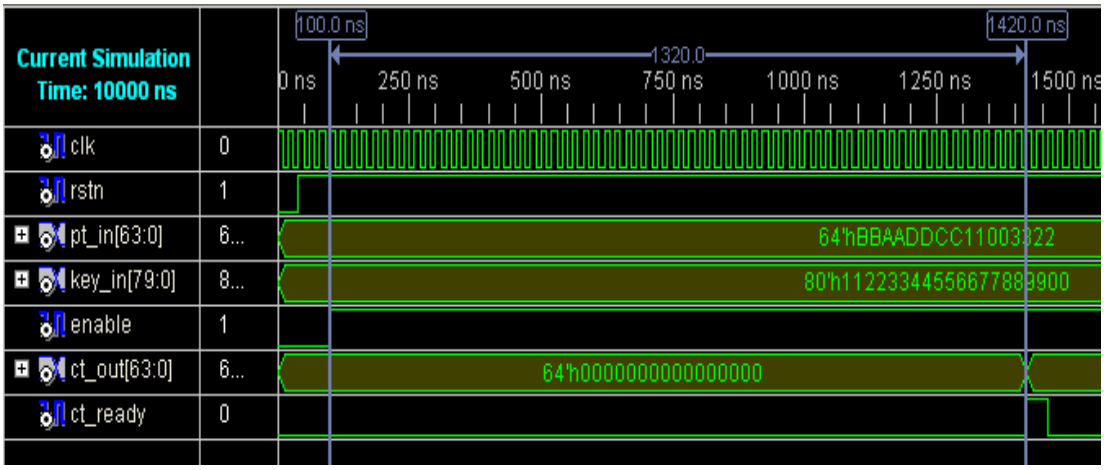
Şekil 5.22: Örnek Çalışma Güç Tüketimi (8 MHz'de)

Name	Value	Used	Total Available	Utilization (%)
Clocks	0.00051 (W)	1	---	---
Logic	0.00144 (W)	1399	9312	15.0
Signals	0.00504 (W)	1483	---	---
IOs	0.00680 (W)	132	232	56.9
Total Quiescent Power	**0.08116 (W)**			
Total Dynamic Power	**0.01379 (W)**			
Total Power	**0.09494 (W)**			
Junction Temp	**27.5 (degrees C)**			

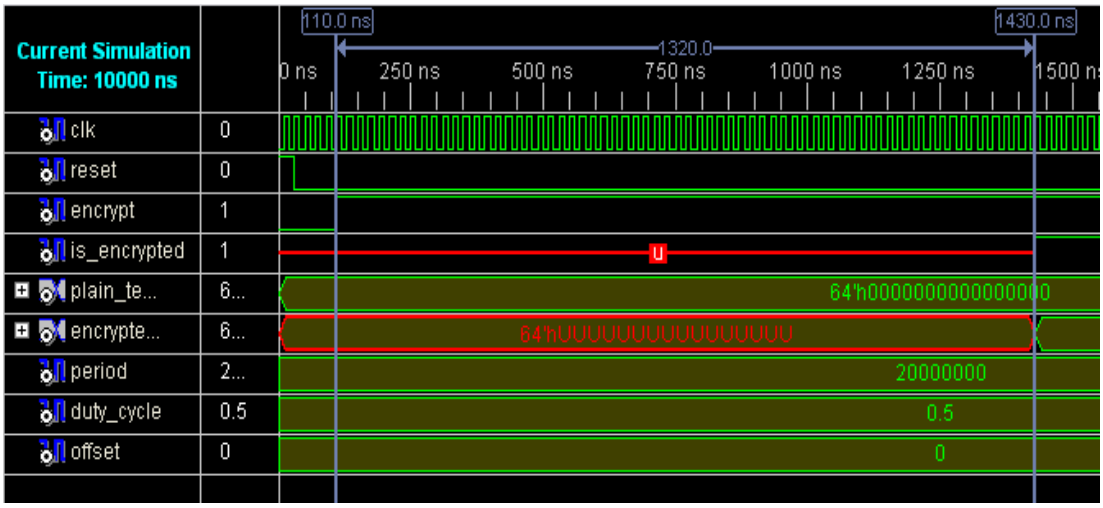
Şekil 5.23: Tez Çalışması Güç Tüketimi (8 MHz'de)

5.5.4. FPGA şifreleme süresi karşılaştırılması

Şekil 5.24. ve Şekil 5.25.'de iki işaretleyici çubuk arasındaki fark zaman görülmektedir. İlk çubuklar şifrelemeyi başlatan giriş sinyalin yükselen kenarına diğer çubuklar da şifreleme işleminin bittiğini belirten çıkış sinyalinin yükselen kenarına hedeflenmektedir. Bu iki çubuk arasındaki fark süre, şifreleme için geçen süreyi göstermektedir. Her iki çalışma için şifreleme süresi aynı olup şekillerde görüldüğü gibi 1320 ns'dir.



Şekil 5.24: Örnek Çalışma Şifreleme Süresi (50 MHz'de)



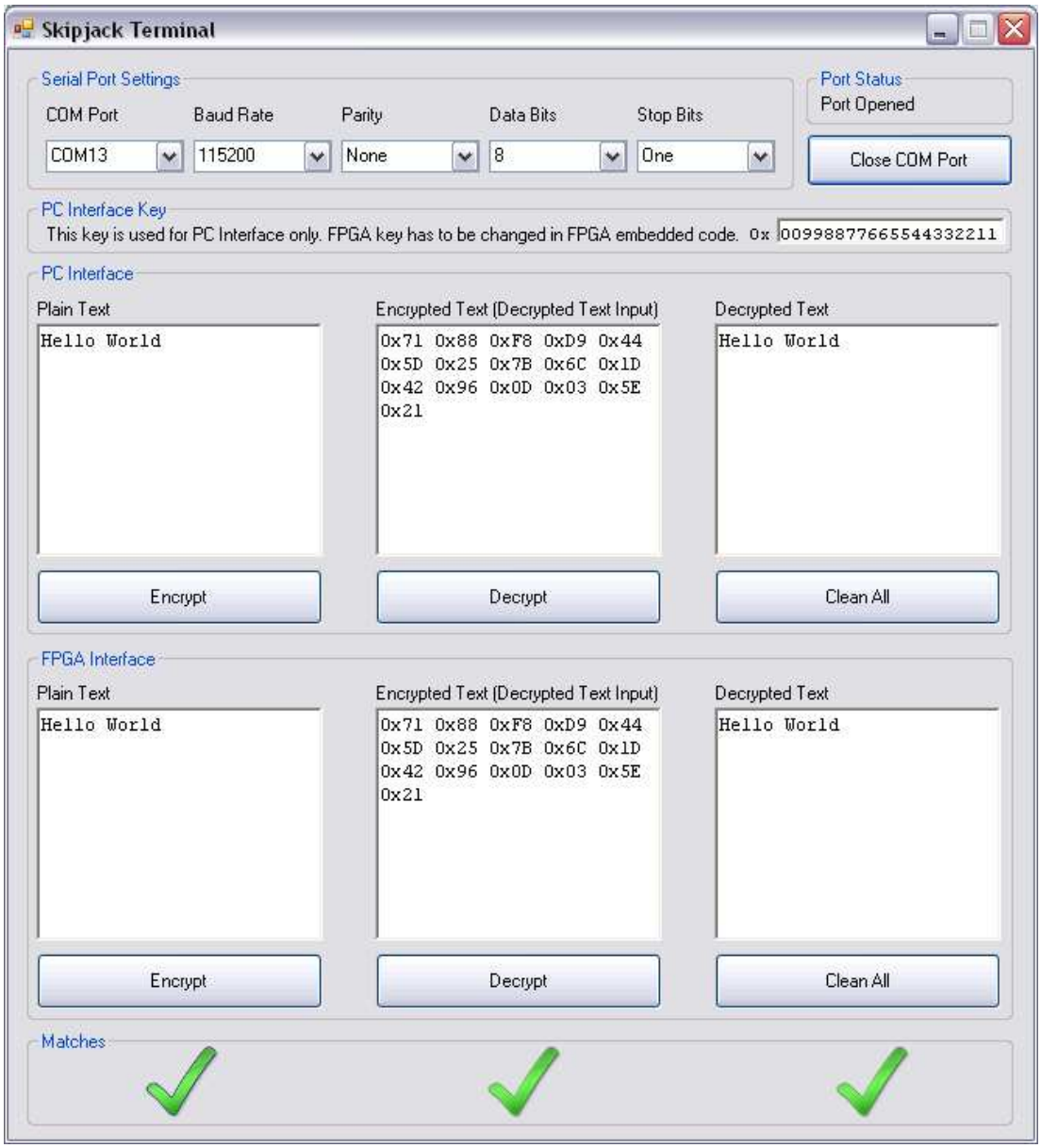
Şekil 5.25: Tez Çalışması Şifreleme Süresi (50 MHz'de)

5.6. Örnek Şifreleme ve Şifre Çözme Uygulaması

Şekil 5.26.'da şifreleme ve şifre çözme işlemelerinden sonra Kullanıcı arayüzü metin kutularının son durumları görülmektedir.

Örnekte, hem PC arayüzü hem de FPGA arayüzü için şifrelenmesi istenen veri "Hello World" karakter katarıdır. Her iki arayüz için girilen veri aynıdır ve karşılaştırma sonucunun doğru olduğunu soldaki yeşil tik işareti belirmektedir. Karşılaştırma sonuçları her iki arayüz için ortadaki metin kutularında gösterilmektedir. Her iki arayüz için sonuçlar birebir aynı olduğu için, karşılaştırma sonucunun doğru olduğunu ortadaki yeşil tik işareti belirmektedir. Şifre çöz butonuna (Decrypt etiketli) basıldığında ortadaki metin kutusundaki veriler şifre

çözme işlemi için girdi olarak kullanılmaktadır. Şifre çözme işlemi sonuçları sağdaki metin kutularında gösterilmektedir. Yine her iki arayüz için sonuçlar birebir aynı olduğundan, sağdaki yeşil renkli tik işareti ile sonuçların doğru olduğu belirtilmektedir. İlgili metin kutuları karşılaştırmalarında farklılık görülmediğinden veri tutarsızlığını belirtmek amacıyla kullanılan kırmızı renkli X işareti görülmemektedir.



Şekil 5.26: Örnek Skipjack Şifreleme / Şifre Çözme

Sonuç olarak; her iki arayüz tarafından Decrypted Text etiketli metin kutusunda gösterilen karakter katarı “Hello World”, giriş karakter katarı ile aynı olduğundan Skipjack algoritmasının hem PC arayüzü tarafında hem de FPGA araüyüzü tarafında doğru bir şekilde çalıştığı kanıtlanmaktadır.

5.7. Sonuç

Yukarıdaki veriler doğrultusunda tez çalışması ile örnek çalışma arasında bir genel performans karşılaştırması yapılırsa, tez çalışması performansının örnek çalışma performansına göre çok daha iyi olduğu görülmektedir. Çalışmalar, şifreleme için aynı süreyi harcasalar da bu tez çalışması, gömülü yazılım olarak % 31,57 gibi daha az kod boyutuna sahip olması sebebiyle örnek çalışmaya göre daha başarılıdır. Bunun yanı sıra, toplam güç tüketiminde örnek çalışmaya göre de % 3,53 daha az güç tüketimi başarısı sağlanmaktadır.

Sonuç olarak tez çalışması ile örnek çalışma en uygun performansları kıyaslandığında, % 31,57 gibi büyük oranda gömülü yazılımdaki düşüş nedeni ve % 3,53’lük daha az güç tüketimi ile tez çalışmasının daha başarılı olduğu görülmektedir.

6. SONUÇ VE DEĞERLENDİRMELER

Skipjack algoritması ve arayüzlerinin geliştirildiği bu tez çalışmalarında, algoritmanın FPGA ortamında gerçeklenmesi ile KAA uygulamalarında günümüzde çok yeri olmayan, ancak ilerleyen teknolojinin getirdiği yenilikler ile gittikçe önemli bir konuma sahip olmaya başlayan FPGA donanımlarının KAA uygulamalarında önemini artıracağı düşünülerek, tez çalışmasının araştırmacılar için ileriye dönük bir örnek olması hedeflenmektedir.

Uygulama sonucunda elde edilen veriler, FPGA kullanımının mikrodenetleyici kullanımına nazaran önemli hız artışı sağladığı; elde edilen bu hız artışının daha kısa sürede veri işleme yeteneği getirdiği bilinmektedir. Daha kısa sürede veri işleme kabiliyetinin de sistem kaynakları kullanımında ve güç tüketiminde azalmaya etkisi olacağı düşünülmektedir.

Bu tez çalışmasında elde edilen sonuçlar doğrultusunda, KAA uygulamalarında FPGA kullanımın, mikrodenetleyicilere nazaran işlevsel olarak başarımı oldukça iyileştirdiği ve buna bağlı olarak ağ güvenliğini artırdığı değerlendirilmektedir. Her ne kadar FPGA'ler KAA uygulamalarında kritik bir öneme sahip olan güç tüketimi konusunda istenilen verimi sağlayamasa da ilerleyen teknoloji ile çok kısa sürede mikrodenetleyici seviyesinde güç tüketimi olan FPGA'lerin üretilmesi KAA uygulamalarındaki donanım tercihlerini büyük oranda değiştirebilecektir. Bu durumda bahsi geçen güç tüketimi azalımı ile FPGA'lerin başarımlarının mikrodenetleyicilerin başarımlarından genel olarak daha iyi olacağı aşikardır.

Uygulamada FPGA kullanımının getirdiği bir dezavantaj bulunmaktadır. Bu dezavantaj şu an için üretilen FPGA'lerin güç tüketimin KAA uygulamalarında kabul edilebilecek seviyenin biraz üstünde olmasıdır. Bu, FPGA kullanımın getirdiği zorluk olarak düşünülmektedir. Yalnız ilerleyen birkaç yılda her donanımda olduğu gibi FPGA üretiminde de daha az güç tüketen FPGA'lerin olacağı kesindir. Bu

noktadan hareketle, yakın zamanda FPGA kullanımı olan daha yüksek başarımlı KAA uygulamalarının görülebileceği bilinmektedir.

Tüm bu veriler ışığında, önümüzdeki birkaç yıla yönelik ön bir çalışma niteliğindeki bu tez çalışmasının önemli bir uygulama örneği olduğu değerlendirilmektedir. Mevcut KAA uygulamalarında kullanılan şifreleme/şifre çözme işlemlerini, FPGA'lerin işlevsel olarak daha yüksek başarım ile gerçekleştirmesi, güç tüketimin günümüz donanımları seviyesini inmesiyle çok daha artacak, şifreleme/şifre çözme işlemleri dışında görev yapan FPGA'lerin kullanımının önünü açacaktır. Buna ek olarak, sunulan tez çalışması, FPGA'lerin getirdiği işlevsel hız artışını kanıtladığı gibi sadece şifreleme/şifre çözme gerçekleştiren çok daha güçlü donanımların da oluşmasına yol açabilecek bir niteliktedir.

KAYNAKLAR

[1] Akyıldız, I.F., Su, W., Sankasubramaniam, Y., Çayırcı, E., "Wireless Sensor Networks: A Survey", ***Computer Networks***, 393 – 422, (2002).

[2] Virk K., Madsen J., "Functional Testing of Wireless Sensor Node Designs", ***Innovations in Information Technology, IIT '07. 4th International Conference on,*** (2007).

[3] *What are FPGAs?* [online], http://www.fpga4fun.com/FPGAinfo1.html (**Ziyaret Tarihi: 29 Mart 2009**).

[4] Vieira, M.A.M., Coelho, C.N., Jr. da Silva, D.C., Jr. da Mata, J.M.., "Survey On Wireless Network Devices", ***Emerging Technologies and Factory Automation***, Proceedings. ETFA '03. IEEE Conference , vol.1, no., pp. 537-544 vol.1, 16-19 Sept. (2003).

[5] Roman R., Alcarez C., Lopez J., "A Survey of Cryptographic Primitives and Implementations for Hardware Constrained Sensor Network Nodes", ***Mobile Networks and Applications***, 12:231-244, August (2007).

[6] Blake I., Seroussi G., "Smart NP. Elliptic Curves In Cryptography.", ***Cambridge Uni. Press***, ISBN 0-521-65374-6, (2000).

[7] *Specification of Skipjack* [online], http://csrc.nist.gov/groups/ST/toolkit/ documents/skipjack/skipjack.pdf (**Ziyaret Tarihi: 29 Mart 2009**).

[8] Rivest Ronald L., *The RC5 encryption algorithm*. [online], http://people.csail.mit.edu/rivest/Rivest-rc5.pdf (**Ziyaret Tarihi: 22 Haziran 2009**).

[9] Rivest RL, Robshaw MJB, …, *The RC6 Block Cipher, V1.1, August 20, 1998* [online], ftp://ftp.rsasecurity.com/pub/rsalabs/rc6/rc6v11.pdf (**Ziyaret Tarihi: 29 Mart 2009**).

[10] FIPS 197, *Advanced Encryption Standard (AES)* [online], http://csrc.nist.gov/ publications/fips/fips197/fips-197.pdf (**Ziyaret Tarihi: 29 Mart 2009**).

[11] Roman R., Alcarez C., Lopez J., "Applying key infrastructures for sensor networks in CIP/CIIP scenarios.", ***In proceeding of the 1st international workshop on critical information infrastructures security, (CRITIS 2006).***, Aug. (2006).

[12] Rivest R., Shamire A., Adleman L., "A Method for Obtaining Digital Signature and Public-Key Cryptosystems.", **Commu. ACM**, 21(2):120-126 (1978).

[13] Hoffstein J., Pipher J., Silverman J. H., "NTRU: A Ring Based Public Key Cryptosystem." ***In proceedings of the 3rd algorithmic number theory symposium***, *(ANTS 1998).*, Jun. (1998).

[14] Rabin MO., "Digitalized Signatures and Public Key Functions As Intractable As Factorization.", ***Technical Report MIT/LCS/TR-212***, MIT (1979).

[15] Wolf C., Preneel B., "Taxonomy of Public Key Schemes Based On The Problem of Multivariate Quadratic Equations.", ***Cryptology ePrint Archive***, Report 2005/077 (2005).

[16] Eastlake D., Jones P., "US secure hash algorithm 1 (SHA1).", **Rfc 3174**.

[17] Dobbertin H., Bosselaers., "RIPEMD-160, A Strengthened Version of RIPEMD.", ***In proceedings of the 3rd international workshop on fast software encryption (FSE 1996).***, Feb. (1996).

[18] Undercoffer, J., Avancha, S., Joshi, A., and Pinkston, J., "Security for Sensor Networks", ***CADIP Research Symposium***, (2002).

[19] Hollar, S., "COTS Dust", Master's Thesis, Electrical Engineering and Computer Science Department, **UC Berkeley**, (2000).

[20] Karaboğa, D., Ökdem S., "Kablosuz Algılayıcı Ağlarında Güvenli İletişim Teknikleri", ***3rd Information Security & Cryptology Conference With International Participation***, December (2008).

[21] *Specification of Data Encryption Standard (DES)* [online], http://www.itl.nist.gov/fipspubs/fip46-2.htm (**Ziyaret Tarihi: 15 Nisan 2009**).

[22] *Symmetric Encrytion* [online], http://www.ficora.fi/en/index/palvelutaiheittain/tietoturva/salausmenetelmat/symmetrinensalaus.html (**Ziyaret Tarihi: 29 Mart 2009**).

[23] *Açık Anahtar Altyapısı Eğitim Kitabı* [online], http://www.kamusm.gov.tr/tr/Bilgideposu/Belgeler/teknik/aaa/index.html (**Ziyaret Tarihi: 29 Mart 2009**).

[24] *Asymmetric Encrytion* [online], http://www.ficora.fi/en/index/palvelut/palveluta iheittain/tietoturva/salausmenetelmat/epasymmetrinensalaus.html (**Ziyaret Tarihi: 29 Mart 2009**).

[25] Kuon I., Tessier R., Rose J., "FPGA Architecture: Survey and Challenges", ***Foundations and TrendsR_ in Electronic Design Automation Vol. 2, No. 2***, (2007).

[26] Aktaş M., *FPGA Mimarisi* [online], http://www3.itu.edu.tr/~orencik/BilgMimYenYakl2007/Mehmet_Aktas/FPGA_Mimarisi_Rapor.pdf (**Ziyaret Tarihi: 15 Nisan 2009**).

[27] D. Frohman-Dentchkowsky, “A Fully-Decoded 2048-bit Electrically Programmable MOS ROM”, ***IEEE International Solid State Circuits Conference Digest of Technical Papers***, pp. 80–81, February (1971).

[28] R. Cuppens, C. D. Hartgring, J. F. Verwey, H. L. Peek, F. A. H. Vollebraft, E. G. M. Devens, and I. A. Sens, “An EEPROM for Microprocessors and Custom Logic”, ***IEEE Journal of Solid-State Circuits***, vol. 20, no. 2, pp. 603–608, April (1985).

[29] D. C. Guterman, L. H. Rimawi, T.-L. Chiu, R. D. Halvorson, and D. J. McElroy, “An Electrically Alterable Nonvolatile Memory Cell Using a Floating-Gate Structure”, ***IEEE Transactions on Electron Devices***, vol. 26, no. 4, pp. 576–586, April (1979).

[30] W. Carter, K. Duong, R. H. Freeman, H. Hsieh, J. Y. Ja, J. E. Mahoney, L. T. Ngo, and S. L. Sze, “A User Programmable Reconfiguration Gate Array”, ***Proceedings of the IEEE Custom Integrated Circuits Conference***, pp. 233–235, May (1986).

[31] J. Birkner, A. Chan, H. T. Chua, A. Chao, K. Gordon, B. Kleinman, P. Kolze, and R. Wong, “A Very-High-Speed Field-Programmable Gate Array Using Metalto-metal Antifuse Programmable Elements”, ***Microelectronics Journal***, vol. 23, no. 7, pp. 561–568, November (1992).

[32] S. D. Brown, R. Francis, J. Rose, and Z. Vranesic, “Field-Programmable Gate Arrays”, ***Kluwer Academic Publishers***, (1992).

[33] Actel Corporation, *Axcelerator family FPGAs* [online], http://www.actel.com/ documents/AX DS.pdf, (**Ziyaret Tarihi: Mayıs 2005**).

[34] *Legacy Skipjack Algorithm* [online], http://www.isaakian.com/VHDL/SKIP JACK, (**Ziyaret Tarihi: 19 Mayıs 2009**).

ÖZGEÇMİŞ

1983 yılında Kocaeli ilinde doğdu. İlk ve orta öğrenimini aynı ilde tamamladı. Lise öğrenimini Kocaeli Sabancı Anadolu Teknik Lisesi'nde tamamladı. 2002 yılında Kocaeli Üniversitesi Teknik Eğitim Fakültesi Elektronik ve Bilgisayar Eğitimi Bölümü Elektronik Öğretmenliği programına girdi ve 2006 senesinde Elektronik Öğretmeni olarak mezun oldu. Aynı yıl Kocaeli Üniversitesi Fen Bilimleri Enstitüsü Elektronik ve Bilgisayar Eğitimi Anabilim Dalı'nda yüksek lisansa başladı. Kısa bir dönem kontrol sistemleri ve otomasyon üzerine bir firmada çalıştıktan sonra 2007 yılında Devnet Bilişim Teknolojileri San. ve Tic. Ltd. şirketinde işe başladı. Halen aynı şirkette AR-GE araştırmacı olarak görev yapmaktadır.