T.C.

TRAKYA ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

ŞİFRELEME ALGORİTMASININ FPGA'DA UYGULANMASI

Yasin AKMAN

YÜKSEK LİSANS TEZİ

BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

Tez Danışmanı: YRD. DOÇ. DR. TARIK YERLİKAYA

EDİRNE-2015

T.Ü. Fen Bilimleri Enstitüsü onayı

PROF. DR. MUSTAFA ÖZCAN

Fen Bilimleri Enstitüsü Müdürü

Bu tezin Yüksek Lisans tezi olarak gerekli şartları sağladığını onaylarım.

PROF. DR. YILMAZ KILIÇASLAN

AnabilimDalı Başkanı

Bu tez tarafımca (tarafımızca) okunmuş, kapsamı ve niteliği açısından bir Yüksek Lisanstezi olarak kabul edilmiştir.

YRD.DOÇ.DR. TARIK YERLİKAYA

Tez Danışmanı

Bu tez, tarafımızca okunmuş, kapsam ve niteliği açısından Bilgisayar MühendisliğiAnabilim Dalında bir Yüksek Lisans tezi olarak oy birliği ile kabul edilmiştir.

Jüri Üyeleri İmza

YRD. DOÇ. DR. TARIK YERLİKAYA

YRD. DOÇ. DR. ERCAN BULUŞ

YRD. DOÇ. DR. DENİZ TAŞKIN

Tarih: 05/01/2015

T.Ü. FEN BİLİMLERİ ENSTİTÜSÜ

BİLGİSAYAR MÜHENDİSLİĞİ YÜKSEK LİSANS PROGRAMI

DOĞRULUK BEYANI

İlgili tezin akademik ve etik kurallara uygun olarak yazıldığını ve kullanılan tüm literatür bilgilerinin kaynak gösterilerek ilgili tezde yer aldığını beyan ederim.

10/12/2014

YASİN AKMAN

Yüksek Lisans Tezi
Şifreleme Algoritmasının FPGA'da Uygulanması
T.Ü. Fen Bilimleri Enstitüsü
Bilgisayar Mühendisliği Anabilim Dalı

ÖZET

Gelişmiş Veri Şifreleme Standardı (AES), elektronik verilerin korunması amacıyla FIPS (Federal Information Processing Standart) onayı ile yayımlanmış bir kriptografik algoritmadır. AES algoritması yazılımsal ve donanımsal olarak programlanabilir. Bu tezde, AES algoritmasının FPGA ve bilgisayardaki şifreleme süresinin karşılaştırılması sunulmaktadır. FPGA'da, Verilog DTD (Donanım Tanımlama Dili) kullanılmıştır. 128 bit giriş metni (input) ve anahtarına uzunluğuna (key length) sahip AES (AES-128) algoritması Xilinx ISE Design Suite 13.3 programında sentezlenerek benzetimi oluşturulmuştur. Ek olarak, AES-128 algoritması bilgisayar üzerinde C programlama dili kullanılarak kodlanmıştır. FPGA ve bilgisayarın şifreleme zamanı ölçülmüştür. AES-128 şifreleme algoritmasını, FPGA'nın bir bilgisayardan daha hızlı çalıştırdığı gözlemlenmiştir. AES-128'in FPGA'daki şifreleme süresi 390ns ve bir bilgisayardaki şifreleme süresi 11µs'dir.

Yıl : 2015
Sayfa Sayısı : 62
Anahtar Kelimeler : FPGA, Şifreleme, AES, AES-128, Şifreleme Zamanı

Master's Thesis
Encryption Algorithm Implementation on FPGA
Trakya University Institute of Natural Sciences
Department Of Computer Engineering

ABSTRACT

Advanced Encryption Standard (AES), which is approved and published by Federal Information Processing Standard (FIPS), is a cryptographic algorithm that can be used to protect electronic data. The AES algorithm can be programmed in software or hardware. In this thesis, presents encryption time comparison of the AES algorithm on FPGA and computer. Verilog HDL (Hardware Description Language) used on FPGA. The AES algorithm with 128-bit input and key length (AES-128) was synthesized and simulated on Xilinx ISE Design Suite 13.3. In addition, AES-128 coded C programming language on computer. I measured the time of encryption on FPGA and computer. It was observed that, the AES encryption algorithm runs on the FPGA faster than on a computer. Encryption time is 390ns of AES-128 on FPGA and 11µs of AES-128 on a computer.

Year : 2015
Number of Pages : 62
Keywords : FPGA, Encryption, AES, AES-128, Encryption Time

TEŞEKKÜR

Öncelikle, beni her zaman destekleyen, Trakya Üniversitesi Bilgisayar Mühendisliği Öğretim Üyesi Değerli Hocam Yrd. Doç. Dr. Tarık YERLİKAYA'ya, FPGA alanındaki yapıcı eleştirileri ile beni destekleyen Trakya Üniversitesi Bilgisayar Mühendisliği Öğretim Üyesi Değerli Hocam Yrd. Doç. Dr. Deniz TAŞKIN'a, İngilizce çeviri alanında tüm bilgi birikimi ile bana destek olan KTO Karatay Üniversitesi Hazırlık Sınıfı Koordinatörlüğü Öğretim Elemanı Değerli Arkadaşım, Okt. Samet HASIRCIOĞLU'na, yayınımı özenle inceleyerek üzerinde çalışılması gereken kısımları tespit ederek, yapıcı eleştirilerde bulunan Kocaeli Üniversitesi Bilişim Sistemleri Mühendisliği Öğretim Üyesi Değerli Hocam Doç. Dr. Mehmet YILDIRIM'a, her konuda desteklerini yanımda hissettiğim Sakarya Üniversitesi Bilgisayar Mühendisliği Öğretim Üyelerinden Değerli Hocalarım Doç. Dr. Cüneyt BAYILMIŞ ve Doç. Dr. Celal ÇEKEN'e, Trakya Üniversitesi Bilgisayar Mühendisliği Bölümü hocalarıma, Kocaeli Üniversitesi Teknoloji Fakültesi Bilişim Sistemleri Mühendisliği Bölümü hocalarıma, Namık Kemal Üniversitesi Bilgisayar Mühendisliği Bölümü hocalarıma sonsuz teşekkürü bir borç bilirim.

İÇİNDEKİLER

ÖZET i
ABSTRACT ii
TEŞEKKÜR iii
İÇİNDEKİLER iv
SİMGELER VE KISALTMALAR vii
TABLOLAR DİZİNİ viii
ŞEKİLLER DİZİNİ ix
1. GİRİŞ 1
2. PROGRAMLANABİLİR MANTIKSAL AYGITLAR 2
2.1. PROM (Programmable Read Only Memory) 2
2.2. PLA (Programmable Logic Array) 3
2.3. PAL (Programmable Array Logic) 4
2.4. GAL (Generic Array Logic) 4
2.5. CPLD (Complex Programmable Logic Device) 5
2.6. ASIC (Application Specific Integrated Circuit) 6
2.6.1. Gate Array (Kapı Dizisi) 7
2.6.2. Standart Cell (Standart Hücre) 8
2.6.3. Structured ASIC (Yapısal ASIC) 9
2.6.4. Full Custom (Tam Özel) 10
2.7. FPGA (SAHADA PROGRAMLANABİLİR KAPI DİZİSİ) 11
2.7.1. FPGA Yapısı 12
2.7.1.1. Programlanabilir Mantıksal Bloklar (Configurable Logic Blocks-CLB) 13
2.7.1.2. Giriş/Çıkış Blokları (I/O Blocks) 14
2.7.1.3. Ara Bağlantı Kanalları (Interconnection Resources) 15
2.7.1.4. Blok RAM 16
2.7.2. FPGA Programlama Teknolojileri 17
2.7.2.1. Bir Kez Programlanabilir FPGA (OTP FPGA) 17
2.7.2.1.1. EPROM Teknolojisi 17
2.7.2.1.2. Sigorta (Fuse) Teknolojisi 19
2.7.2.1.3. Karşıt Sigorta Teknolojisi 20
2.7.2.2. Tekrar Programlanabilir FPGA (Re-Programmable FPGA) 22

2.7.2.2.1. SRAM Merkezli FPGA (SRAM-Based FPGA) 22
2.7.2.2.2. Flash Merkezli FPGA (Flash-Based FPGA) 23
2.7.2.3. FPGA Programlama Teknolojilerinin Karşılaştırılması 23
2.7.3. Güvenlik Açısından FPGA 24
2.7.3.1. Kopyalanmaya Karşı Koruma 25
2.7.3.2. Tersine Mühendislik İçin Karşı Koruma 26
2.7.3.3. Değiştirmeye Karşı Koruma 27
2.7.3.4. Sonuç 28
2.7.4. FPGA Tasarım Akışı 28
2.7.4.1. Tasarım Girişi (Design Entry) 29
2.7.4.2. Sentez (Synthesis) 30
2.7.4.3. Eşleştirme (Map) 30
2.7.4.4. Yerleşim ve Yol Belirleme (Place & Route) 30
2.7.4.5. Benzetim (Simulation) 31
2.7.4.6. Kaynak Kod Oluşturma (Bitstream Generation) 31
2.7.5. FPGA'nın Donanım Tanımlama Dili ile Programlanması 31
2.7.5.1. Verilog HDL 32
2.7.5.1.1. Yapısal Biçim (Structural Style) 33
2.7.5.1.2. Veri Akışı Biçimi (Dataflow Style) 34
2.7.5.1.3. Davranışsal Biçim (Behavioral Style) 34
3. GELİŞMİŞ ŞİFRELEME STANDARDI (AES) 36
3.1. AES'nin Genel Yapısı 38
3.1.1. Bayt'ları Yer Değiştirme 40
3.1.2. Satırları Öteleme 41
3.1.3. Sütunları Karıştırma 41
3.1.4. Tur Anahtarı Ekleme 42
3.1.4.1. Anahtar Genişletme 43
3.1.4.1.1. Kelime Öteleme (Rotword) 45
3.1.4.1.2. Kelime Yer Değiştirme (Subword) 45
3.1.4.1.3. Döngü Sabitleri (RCon - Round Constants) 45
4. AES'NİN FPGA'DA UYGULANMASI VE ŞİFRELEME SÜRESİ ÖLÇÜMÜ 48
5. AES'NİN BİLGİSAYAR ÜZERİNDE ŞİFRELEME SÜRESİ ÖLÇÜMÜ 53
6. SONUÇLAR VE TARTIŞMA 55
ÖNERİLER 57
KAYNAKLAR 58
ÖZ GEÇMİŞ 61

TEZ ÖĞRENCİSİNE AİT TEZ İLE İLGİLİ BİLİMSEL FAALİYETLER 62

SİMGELER VE KISALTMALAR

ABD	: Amerika Birleşik Devletleri
AES	: Advanced Encryption Standard
ASIC	: Application Specific Integrated Circuit
CLB	: Configurable Logic Blocks
CMOS	: Complementary Metal Oxide Semiconductor
CPLD	: Complex Programmable Logic Device
DES	: Data Encryption Standard
DTD	: Donanım Tanımlama Dili
EPROM	: Erasable Programmable Read Only Memory
FIPS	: Federal Information Processing Standard
FIPS-197	: Federal Information Processing Standards Publication 197
FPGA	: Field Programmable Gate Array
GAL	: Generic Array Logic
GCC	: GNU Compiler Collection
HDL	: Hardware Description Language
LUT	: Look Up Table
MOS	: Metal Oxide Semiconductor
NIST	: The National Institute of Standards and Technology
OTP	: One Time Programmable
PLA	: Programmable Logic Array
PLD	: Programmable Logic Device
PROM	: Programmable Read Only Memory
RAM	: Random Access Memory
RCon	: Round Constants
RTL	: Register Transfer Level
SRAM	: Static Random Access Memory
TDES	: Triple Data Encryption Standard
TI	: Texas Instruments
Verilog HDL	: Verilog Hardware Description Language
VHDL	: Very High Speed Integrated Circuit Hardware Description Language
XOR	: eXclusive OR
$\oplus$	: Özel Veya

TABLOLAR DİZİNİ

Tablo 2.1 FPGA Programlama Teknolojilerinin Karşılaştırılması[6] 24
Tablo 3.1 Son Elemeye Kalan Beş Algoritmanın Hangi Açıdan İncelendikleri[31] 37
Tablo 3.2 AES Anahtar Boyutuna Göre, Blok Boyutu, Anahtar Uzunluğu ve Tur Sayısının Karşılaştırılması..... 38
Tablo 3.3 S-Kutusu..... 40
Tablo 3.4 Özel Veya (XOR) İşlemi Doğruluk Tablosu..... 43
Tablo 3.5 Döngüler ve word'ler Arasındaki İlişki[34]..... 43
Tablo 3.6 AES-128 için Döngü Sabitleri..... 45
Tablo 3.7 AES-128 için Döngü Sabitlerinin Hesaplanması..... 46
Tablo 3.8 AES-128 için Anahtarların Elde Edilmesi 46
Tablo 4.1 Giriş Bloğu, Şifreleme Anahtarı ve Şifrelenmiş Metin Değerleri 49
Tablo 4.2 128 bitlik Gelişmiş Şifreleme Standardı (AES-128) İşlem Sonuçları[29] 50
Tablo 5.1 AES Şifreleme Süresinin Ölçüldüğü Bilgisayarın Özellikleri 53

ŞEKİLLER DİZİNİ

Şekil 2.1 Programlanabilir Mantıksal Aygıtların Sınıflandırması 2
Şekil 2.2 Prom Yapısı[1] 3
Şekil 2.3 PLA Yapısı[2] 3
Şekil 2.4 PAL Yapısı[3] 4
Şekil 2.5 GAL Yapısı[4] 5
Şekil 2.6 CPLD Yapısı 6
Şekil 2.7 ASIC Tasarım Yöntemleri 7
Şekil 2.8 Basit Kapı Dizisi Hücrelerinin Örneği 7
Şekil 2.9 Standart Hücre Yapısı 8
Şekil 2.10 Yapısal ASIC Örnekleri 9
Şekil 2.11 Genel Yapısal ASIC 10
Şekil 2.12 PLD ve ASIC Arasındaki Boşluk 11
Şekil 2.13 Xilinx XC2064-50 FPGA 12
Şekil 2.14 FPGA Yapısı[9] 13
Şekil 2.15 Programlanabilir Mantık Bloğunun Yapısı 13
Şekil 2.16 Programlanabilir Mantıksal Bloğun Yapısı[11] 14
Şekil 2.17 FPGA Programlanabilir G/Ç Bloğu[13] 15
Şekil 2.18 FPGA Programlanabilir Bağlantı 16
Şekil 2.19 Switch Matrix 16
Şekil 2.20 Xilinx 7 serisi Blok RAM yapısı 17
Şekil 2.21 Standart MOS Transistor ve EPROM Transistor 18
Şekil 2.22 EPROM Transistor Tabanlı Hafıza Hücresi 18
Şekil 2.23 Programlanmamış Sigorta Programlama Teknolojisi 19
Şekil 2.24 Programlanmış Sigorta Programlama Teknolojisi 20
Şekil 2.25 Programlanmamış Karşıt Sigorta Teknolojisi 21
Şekil 2.26 Programlanmış Karşıt Sigorta Teknolojisi 21
Şekil 2.27 Karşıt Sigorta Teknolojisinin Programlamadan Önceki ve Sonraki Hali 22
Şekil 2.28 SRAM Yapısı 23
Şekil 2.29 FPGA Tasarım Süreci[12] 29
Şekil 2.30 Basit Verilog HDL Kod Parçası 30
Şekil 2.31 Verilog HDL'in Yapısal Biçimi[25] 33
Şekil 2.32 Verilog HDL'in Veri Akış Biçimi[25] 34
Şekil 2.33 Verilog HDL'de Davranışsal Biçim[25] 35
Şekil 3.1 AES Durum Matrisi 38
Şekil 3.2 Tüm AES Algoritması[33] 39
Şekil 3.3 Baytları Yer Değiştirme İşlemi 41
Şekil 3.4 Satırları Öteleme İşlemi 41
Şekil 3.5 Sütunları Karıştırma İşlemi 42

Şekil 3.6 Tur Anahtarı Ekleme İşlemi 42
Şekil 3.7 AES-128 için 44 word'ün oluşturulması[34] 44
Şekil 3.8 Kelime Öteleme İşlemi (Rotword) 45
Şekil 4.1 Xilinx ISE Desing Suite Programının Ekran Görüntüsü 48
Şekil 4.2 AES-128 Şifreleme İşlemi Benzetim Sonucu 52
Şekil 5.1 AES-128'in Bilgisayar Üzerindeki Şifreleme Süresi 54

1. GİRİŞ

Teknolojinin hızla geliştiği bilgi çağı olarak adlandırılan günümüzde, bilgisayar, internet, akıllı telefonlar, gömülü sistemler ve saymakta zorlanabileceğimiz elektronik sistemler hayatımızın vazgeçilmez bir parçası olmuştur. Bu ortamda, verilerin korunması çok önemlidir. Verilerin bulundukları ortamlarda özellikle de bir yerden başka bir yere gönderilirken, aktarımın güvenli bir şekilde sağlanabilmesi için çeşitli şifreleme algoritmaları geliştirilmiştir. Geliştirilen şifreleme algoritmalarının, gelişen çağın ihtiyaçlarına yanıt verebilmeleri çok önemlidir.

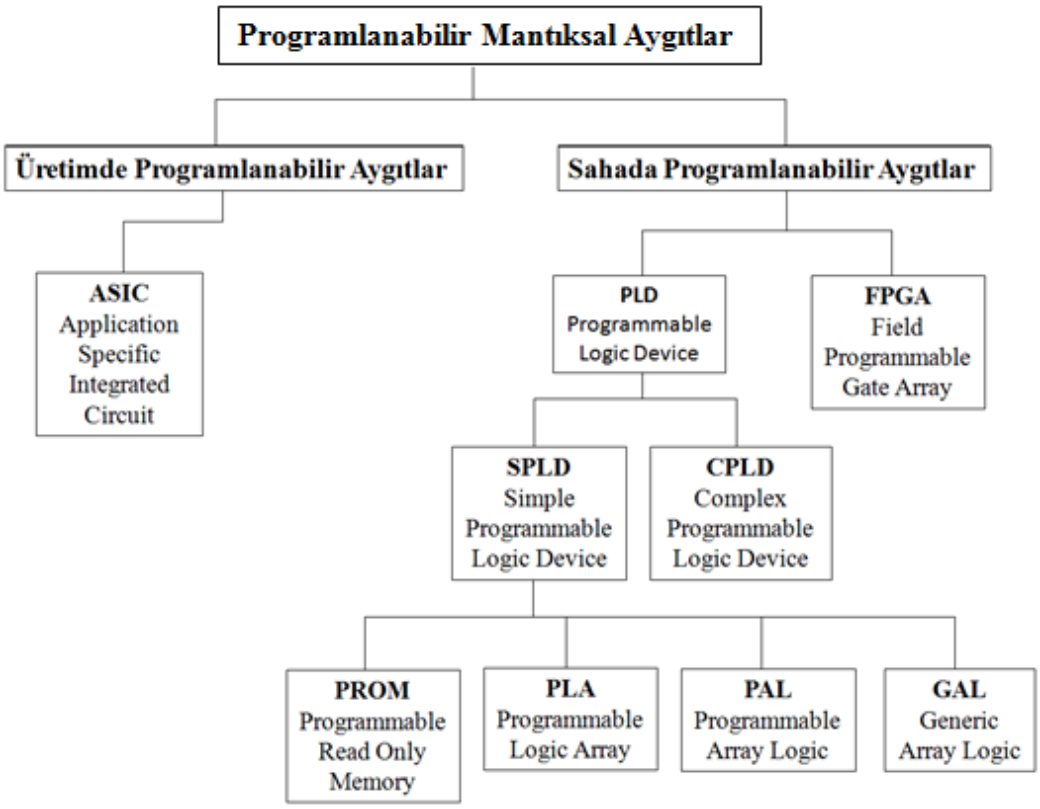
Joan Daemen ve Vincent Rijmen'ın geliştirdiği, Amerikan Ulusal Standartlar ve Teknoloji Enstitüsü (NIST-National Institute of Standards and Technology) tarafından, Gelişmiş Şifreleme Standardı (AES-Advanced Encryption Standard) olarak seçilen algoritma ileri düzey güvenlik sağlayan simetrik bir blok şifreleme algoritmasıdır.

Şifreleme algoritması ile birlikte şifrelemenin gerçekleştirileceği aygıtta çok büyük önem arz etmektedir. Çeşitli aygıtlar üzerinde, çeşitli şifreleme algoritmaları geliştirilmiştir. Hız ve güvenlik yönünden önde gelen aygıtlar, şifreleme işlemlerinin gerçekleştirilmesinde ön plana çıkmaktadır. Bu kapsamda paralel işlem yapabilme yeteneğine sahip olan FPGA'ların tercih edilmesi en doğru seçenek olacaktır.

FPGA üzerinde 128 bit giriş ve şifreleme anahtarı uzunluğuna sahip olan AES'nin (AES-128) uygulanması ve şifreleme süresinin tespit edilmesi amacıyla bu tez çalışması yapılmıştır. Ek olarak, AES-128'in bir bilgisayar üzerindeki şifreleme süresi de tespit edilerek, iki değerin karşılaştırılması ile hangi sistemin, şifreleme işlemini diğerine göre kaç kat daha hızlı gerçekleştirdiğinin tespit edilmesi hedeflenmiştir.

2. PROGRAMLANABİLİR MANTIKSAL AYGITLAR

Programlanabilir mantıksal aygıtları, üretimde programlanabilir aygıtlar ve sahada programlanabilir aygıtlar olmak üzere iki kısma ayırarak inceleyebiliriz. İlk programlanabilir mantıksal aygıt olarak PLD (Programmable Logic Device)'ler kabul edilmektedirler. Şekil 2.1'de görülen programlanabilen mantıksal aygıtların sınıflandırılmasına göre ilgili aygıtlar incelenecektir.

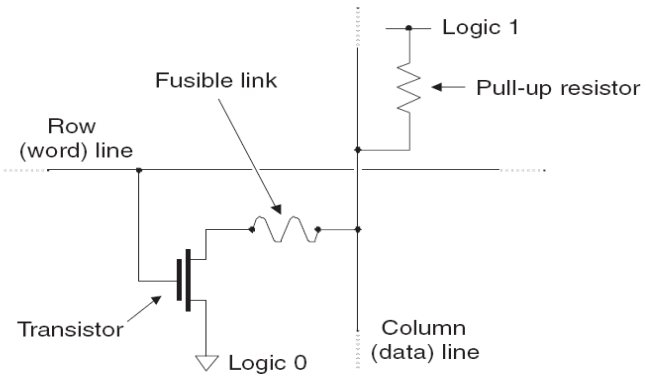


Şekil 2.1 Programlanabilir Mantıksal Aygıtların Sınıflandırması

2.1. PROM (Programmable Read Only Memory)

Basit PLD (Programmable Logic Device)'lerin ilki Harris Semiconductor tarafından 1970'te üretilen, kullanıcı tarafından programlanabilen yalnızca okunabilir bellek olan PROM'dur. Sabit VE (AND) kapısı fonksiyonlarının VEYA (OR) kapı

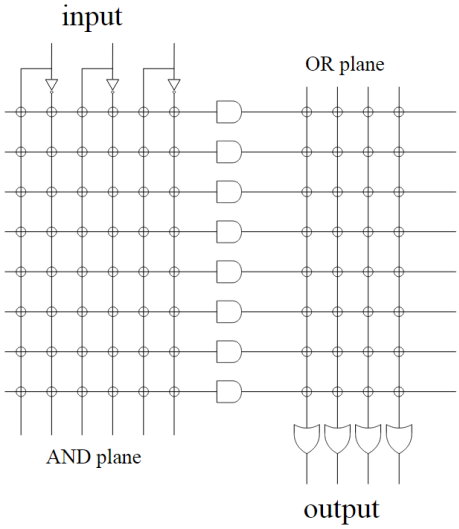
fonksiyonlarını sürmesi şeklinde oluşturulmuştur. PROM'un iç yapısı Şekil 2.2'de görülmektedir.



Şekil 2.2 Prom Yapısı[1]

2.2. PLA (Programmable Logic Array)

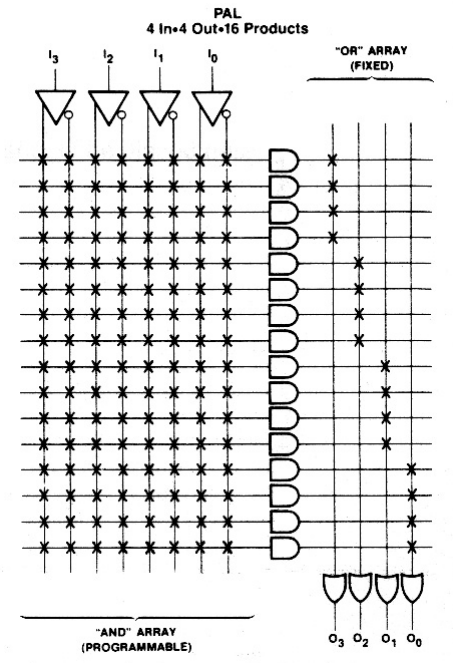
Programlanabilir mantıksal dizi (PLA-Programmable Logic Array), PROM'ların yapısından kaynaklanan sınırlamaların ortadan kaldırılması için 1975 yılında piyasaya sürülmüştür. PLA'larda kullanıcı tarafından, hem VE (AND) hem de VEYA (OR) dizileri programlanabilmektedir. Şekil 2.3'te de görüldüğü üzere iki adet programlanabilme alanı olduğu için her hangi bir VE/VEYA birleşimi gerçekleştirilebilir.



Şekil 2.3 PLA Yapısı[2]

2.3. PAL (Programmable Array Logic)

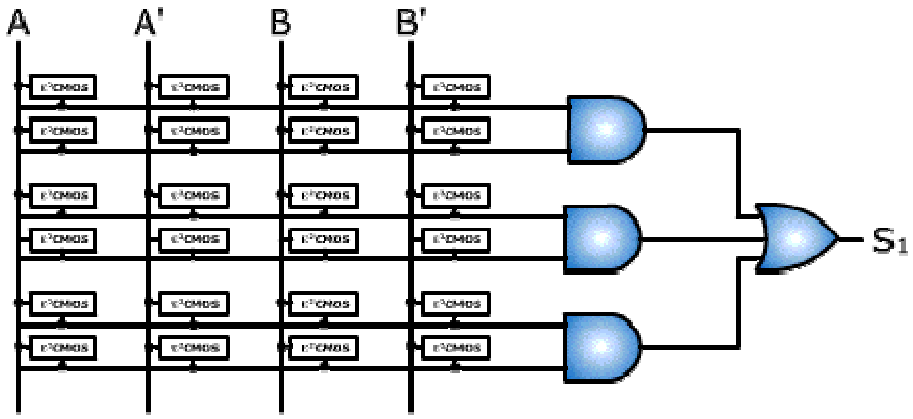
PLA'lardaki hız sorununu çözmek amacıyla 1978'te Monolithic Memories tarafından programlanabilir dizi mantıkları (PAL-Programmable Array Logic) piyasaya sürülmüştür. PAL, PROM'un tam tersidir. Çünkü programlanabilir VE (AND) dizisinden ve önceden tanımlanmış VEYA (OR) dizisinden oluşmaktadır. Önceden tanımlanmış VEYA dizisinin terim sayısının sınırlı olması nedeniyle, PLA'dan daha sınırlıdır. PAL yapısı Şekil 2.4'te görülmektedir.



Şekil 2.4 PAL Yapısı[3]

2.4. GAL (Generic Array Logic)

PAL ile aynı mantıksal özelliklere sahip olan GAL (Generic Array Logic), Bütünleyici Metal Oksit Yarı İletken-CMOS (Complementary Metal Oxide Semiconductor) teknolojisi ile birleştirilerek, elektriksel olarak silinebilen, düşük güç tüketen ve yüksek hızlı bir aygıt olarak, Lattice Semiconductor tarafından 1985 yılında geliştirilmiştir. GAL'ın yapısı Şekil 2.5'te görülmektedir.



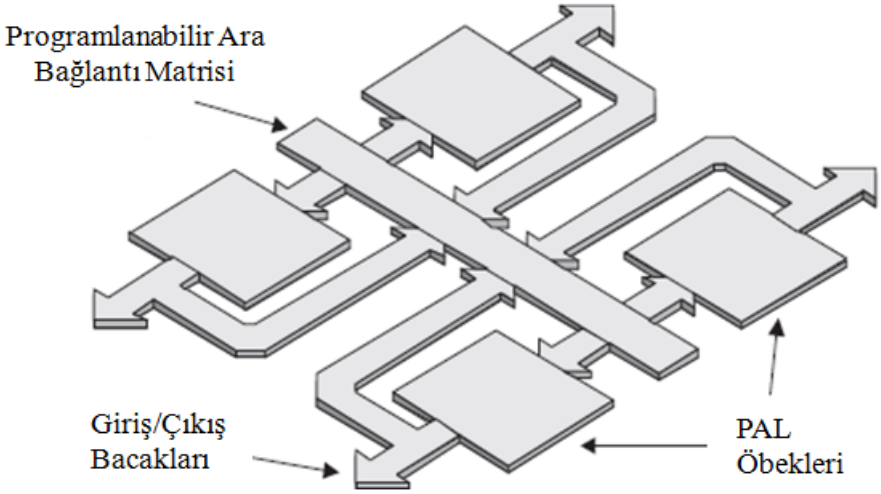
Şekil 2.5 GAL Yapısı[4]

2.5. CPLD (Complex Programmable Logic Device)

Karmaşık programlanabilir mantıksal aygıt olan CPLD (Complex Programmable Logic Device)'ler PAL'lara benzemektedirler. Fakat PAL'lardan daha üstün özelliklere sahiptirler.

PAL'ı üreten Monolithic Memories, 84 uçlu (pinli) ve aralarında bağlantılar olan 4 PAL'dan oluşan MegaPAL adını verdiği orantısız güç tüketen bir malzeme üretmiştir. 1984 yılında Altera, CMOS ve EPROM teknolojilerini birleştirerek kendi ürettiği CPLD'yi tanıtarak bu alanda büyük bir atılım yapılmasını sağlamıştır. Altera'nın CMOS kullanması, ürettiği CPLD'nin çok az güç tüketmesini sağlamıştır. Ayrıca EPROM ile programlanabilme özelliği de, bu malzemeleri geliştirmek için ve örnek hazırlamak için ideal hale gelmesini sağlamıştır. Şekil 2.6'da CPLD yapısı görülmektedir. [25]

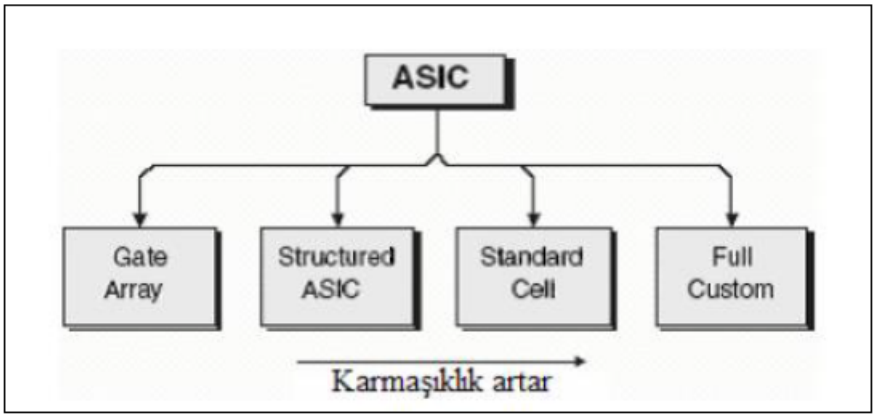
İlerleyen yıllarda, CPLD'nin programlanmasında karşılan zorlukların üstesinden gelinmesi amacıyla PALASM, ABEL ve CuPL gibi diller ve ilgili araçların geliştirilmesi nedeniyle, CPLD'nin kullanımı yaygınlaşmıştır. Ayrıca bu gelişmeler Verilog HDL ve VHDL (HDL-Hardware Description Language) gibi yüksek seviyeli donanım tanımlama dillerinin temeli olarak kabul edilmektedir.



Şekil 2.6 CPLD Yapısı

2.6. ASIC (Application Specific Integrated Circuit)

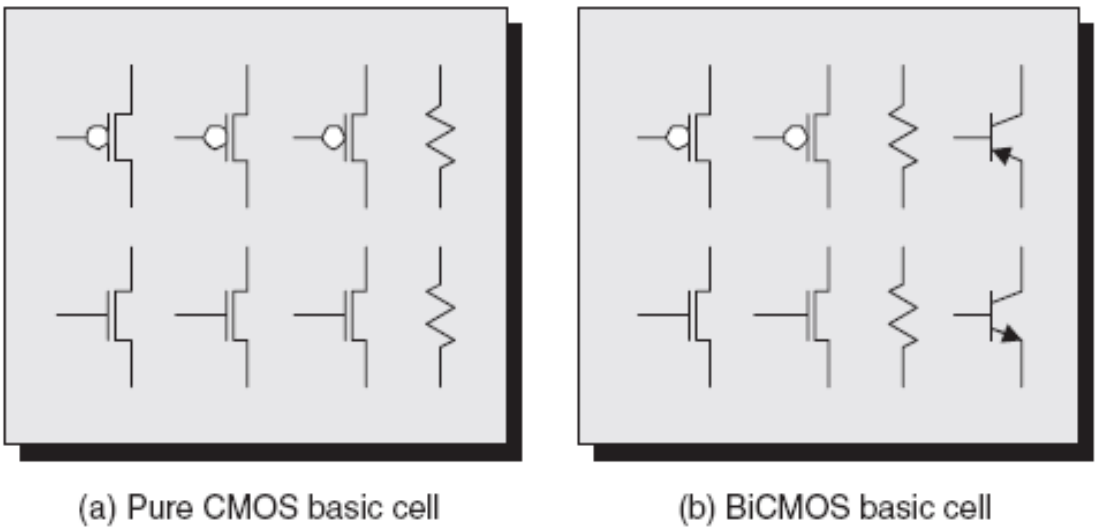
Uygulamaya Özel Tümleşik Devre (ASIC), büyük ve karmaşık işlemleri destekleyen bir aygıttır. İç devreleri mekanik olarak kalıcı bir şekilde tasarlandığından oldukça hızlı çalışmakta ve büyük çaplı seri üretimlerde maliyet açısından oldukça ucuza üretilmektedirler. Fakat üretim ve test aşamaları, sistemin geri dönüşü olmadığından oldukça uzun zaman almaktadır. Proje bazlı üretimlerde, üretim maliyeti çok yüksek olduğundan dolayı tercih edilmemektedirler. ASIC'lar Şekil 2.7'de görüldüğü gibi dört alt ana başlık altında incelenebilmektedirler.



Şekil 2.7 ASIC Tasarım Yöntemleri

2.6.1. Gate Array (Kapı Dizisi)

Kapı dizisi düşüncesi 1960'ların sonlarına doğru Fujitsu ve IBM tarafından ileri sürülmüştür. Kapı dizisi, bağlı olmayan transistör ve dirençler topluluğunu kapsayan temel hücre düşüncesine dayanmaktadır. Şekil 2.8'de basit kapı dizisinin temel hücre örneği görülmektedir. ASIC üreticileri bu temel hücrelerin dizilerini içeren silikon çipleri oluşturarak işe başlamışlardır.

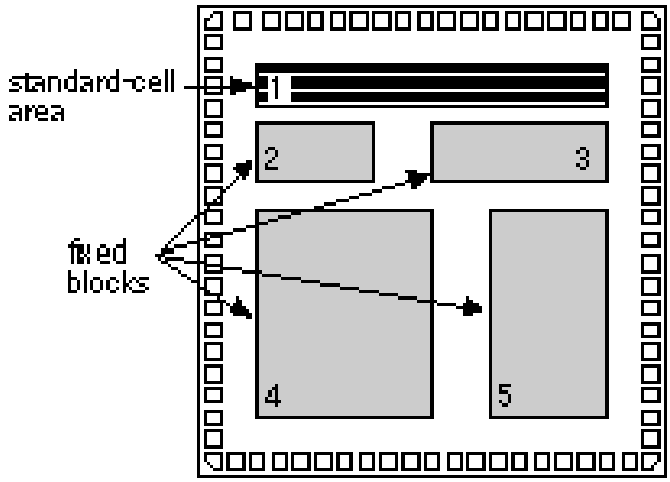


Şekil 2.8 Basit Kapı Dizisi Hücrelerinin Örneği

Kapı dizisindeki transistor ve diğer malzemelerin hazır olması, önemli derecede fiyat avantajı sağlamaktadır. Fakat, çoğu tasarımda önemli derecede iç kaynaktan yararlanılamaması, kapı yerleşiminin sınırlı olması ve iç yolların yerleşiminin en uygun (optimum) şekilde olmaması, fiyat avantajının sağladığı olumlu bakış açısının, olumsuz bir bakış açısına dönüşmesine neden olmaktadır.

2.6.2. Standart Cell (Standart Hücre)

Kapı dizilerindeki sorunları ortadan kaldırmak amacıyla 1980'lerin başında standart hücre entegreleri ortaya çıkmıştır. Kapı dizileri ile bir çok benzerlikleri bulunmaktadır. Fakat kapı dizilerinden farklı olarak temel hücre düşüncesini kullanmamaktadırlar. Her mantıksal kapının birer birer netliste yerleştirilmesi ve bu kapıların birbirine bağlanmasını sağlayan en uygun yolun belirlenmesi amacıyla özel araçlar kullanılmaktadır. Sonrasında, bu sonuçlar, entegrenin üretiminde kullanılan her tabaka için, özel foto-maskeler oluşturulması amacıyla kullanılmaktadır. Standart hücre düşüncesi her mantıksal fonksiyonun en az sayıda transistor kullanılarak oluşturulmasına dayanmaktadır. Kapı dizilerine göre en uygun seviyeye yakın olacak şekilde, silikon kullanılması sağlanmaktadır. Şekil 2.9'da standart hücre yapısı görülmektedir.

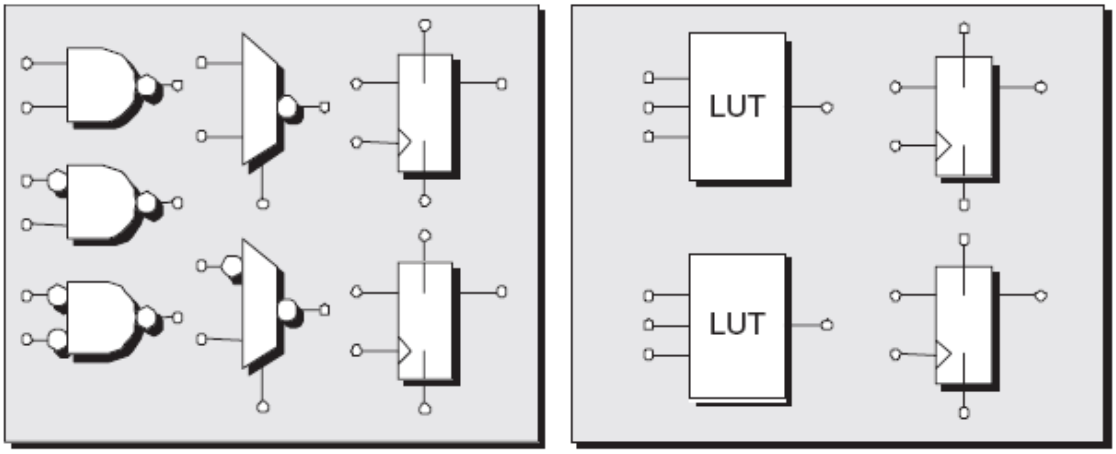


Şekil 2.9 Standart Hücre Yapısı

2.6.3. Structured ASIC (Yapısal ASIC)

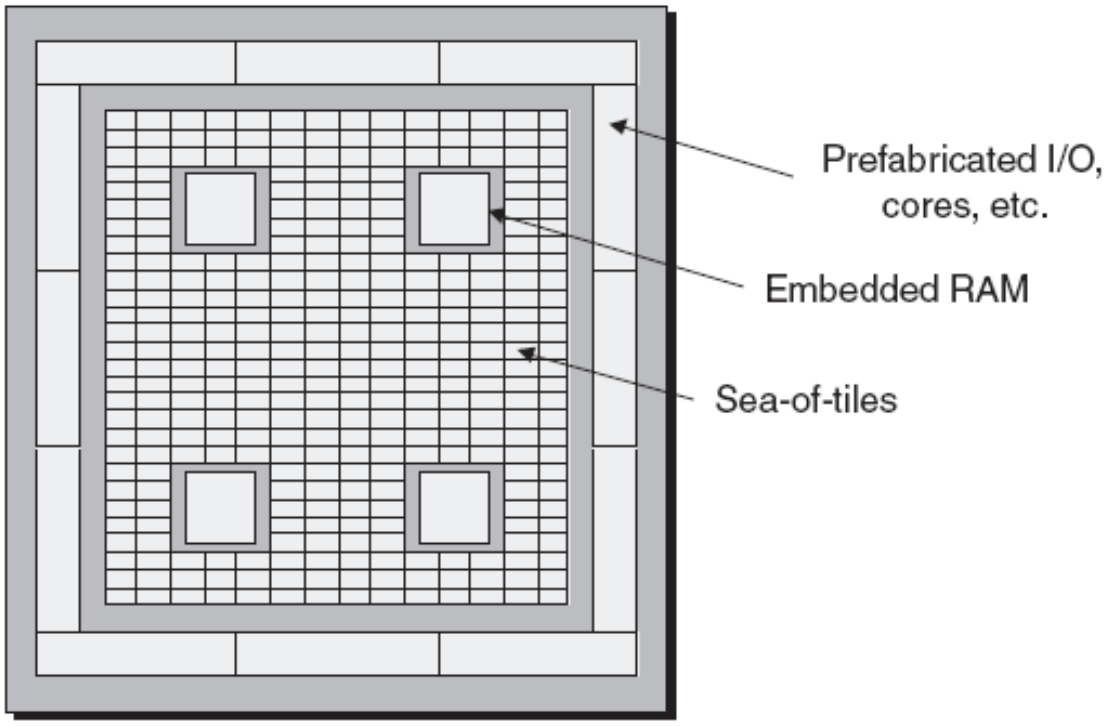
Yapısal ASIC düşüncesi, 1990'ların başlarında ortaya çıkmıştır. Fakat bir süre gündemde kalmış, sonrasında ise ilgi düşüşü yaşamıştır. Yaklaşık 10 yıl sonra, ASIC üreticisi bir grup tasarım maliyetini ve geliştirme süresini düşürmek amacıyla araştırma yapmaya başlamış ve sonrasında, 2003 yılının ortalarında yapısal ASIC'ları ileri sürmüşlerdir. Her üretilen entegrede olduğu gibi, her üretici firma kendi özel yapısını geliştirmiştir. Her entegre, bazı firmaların modül olarak adlandırdığı, bazılarının ise yapı taşı olarak adlandırdığı temel elemanla başlamaktadır. Şekil 2.10'da görülen bu eleman, kapı, multiplexer veya lookup table olarak gerçeklenen bir yada daha fazla kaydedicinin ve küçük bir yerel RAM'in karışımını içerebilmektedir. Yapısal ASIC'ın genel yapısı Şekil 2.11'de görülmektedir.

Yapısal ASIC'ların, standart hücre entegrelerinden 2-3 kat daha fazla elektrik tükettiği söylenmekte, fakat bu sonuçların mimariden mimariye değişebileceği göz önüne alındığında, kesin bir sonuç söylemenin doğru olmayacağı anlaşılmaktadır.



(a) Gate, mux, and flop-based (b) LUT and flop-based

Şekil 2.10 Yapısal ASIC Örnekleri



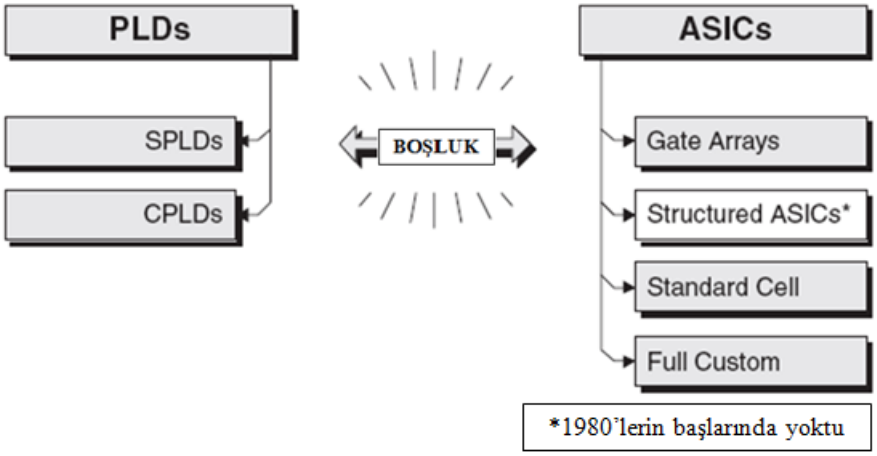
Şekil 2.11 Genel Yapısal ASIC

2.6.4. Full Custom (Tam Özel)

Sayısal entegre devreler, hafıza çipleri hariç tutulduğunda, ilk başlarda iki sınıfa ayrılmaktaydılar. Birinci sınıf oldukça basit yapı blokları (TI ve Fairchild şirketleri tarafından üretilen malzemeler), ikinci sınıf ise özel bir şirket kullanımı için tasarlanan mikroişlemciler gibi tam özel entegrelerdi. Tam özel ASIC'larda, biraz hızlı bir mantıksal kapıya ihtiyaç duyulduğu zaman, o kapıyı oluşturmak için kullanılan transistorların boyutları değiştirilebilmektedir. Tam özel ASIC için tasarım araçları genellikle mühendislerin kendileri tarafından işyerinde tasarlanmaktadır. Tam özel ASIC'ların tasarımı çok karmaşık ve zaman kaybettiricidir. Fakat elde edilen çiplerde en az düzeyde boşa harcanan silikon vardır ve en fazla miktarda lojik içermektedirler.

2.7. FPGA (SAHADA PROGRAMLANABİLİR KAPI DİZİSİ)

1980'li yıllarda PLD'ler ile ASIC'lar arasında bir boşluk görülmeye başlamıştır. Bir yanda SPLD ve CPLD gibi yüksek yapılandırılabilme ve hızlı tasarımı destekleyen fakat geniş ve çok karmaşık tasarımları desteklemeyen (CPLD belli düzeyde karmaşık tasarımları desteklemektedir. Fakat FPGA, CPLD'den daha karmaşık tasarımları da desteklemektedir) programlanabilir yongalar varken, diğer tarafta ise çok karmaşık tasarımları destekleyen fakat devreye özel üretimleri çok pahalı olan, tekrar tekrar programlama özelliğine sahip olmayan ASIC'ler bulunmaktaydı. Bu boşluk Şekil 2.12'de görülmektedir.



Şekil 2.12 PLD ve ASIC Arasındaki Boşluk

Bu boşluğu gidermek amacı ile Xilinx firmasının kurucuları olan Ross Freeman ve Bernard Vonderschmitt tarafından Sahada Programlanabilir Kapı Dizisi (FPGA-Field Programmable Gate Array) olarak isimlendirilen ilk ticari FPGA olan, XC2064 model FPGA, 1985 yılında piyasaya sürülmüştür. Bu modelde 64 yapılandırılabilir mantık bloğu (CLB-Configurable Logic Blocks) ve iki-üç girişli arama tabloları (LUT-Look Up Table) bulunmaktadır[5]. Şekil 2.13'te XC2064-50 FPGA görülmektedir.



Şekil 2.13 Xilinx XC2064-50 FPGA

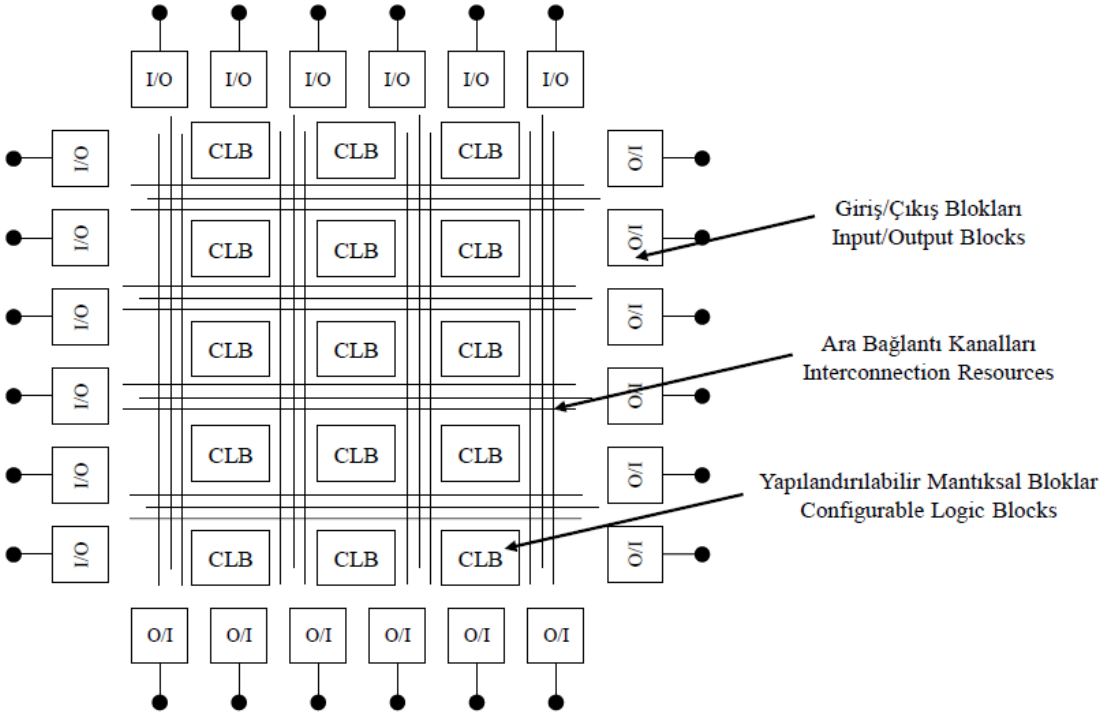
FPGA'lar, yapılandırılabilir veya programlanabilir olarak da adlandırılan mantık blokları (CLB) ve bu bloklar arasındaki ara bağlantılardan oluşan, uygulama alanlarının geniş olduğu sayısal tümleşik devrelerdir[6,7]. FPGA üretim aşamasından sonra, sahada kullanıcı tarafından programlanabilmektedir. FPGA'lar, alternatifi olan mikroişlemcilere oranla sağladığı güvenlik, yüksek işlem gücü ve ASIC'lara oranla sağladığı kullanım kolaylığı ve maliyet artıları ile yakın zamanda elektronik tasarımların vazgeçilmez parçası haline gelmiştirler. Günümüzde FPGA teknolojisi; uzay, işaret işleme, havacılık sistemleri, kriptografik uygulamalar savunma sanayi vb. gibi bir çok alanda yaygın olarak kullanılmaktadır. FPGA teknolojisi, ortaya çıktığından beri sağladığı üstün özellikleri nedeniyle talep akınına uğramıştır. [8]

Ayrıca, FPGA'ların en önemli özelliklerinden biri olan paralel işlem yapabilme yeteneği unutulmamalıdır.

2.7.1. FPGA Yapısı

FPGA temel olarak aşağıda sıralaması yapılan ve Şekil 2.14'te görülen üç bloktan oluşmaktadır.

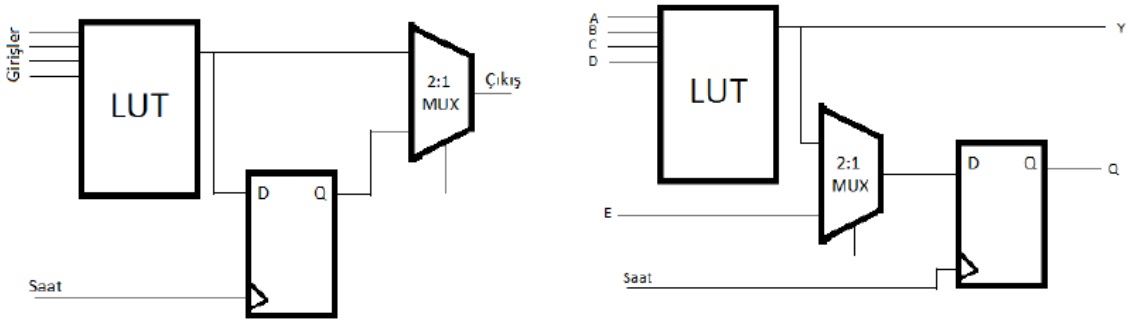
- Programlanabilir mantıksal bloklar (CLB-Configurable Logic Blocks)
- Giriş/Çıkış Blokları (I/O-Input/Output Blocks)
- Ara bağlantı kanalları (Interconnection Resources)



Şekil 2.14 FPGA Yapısı[9]

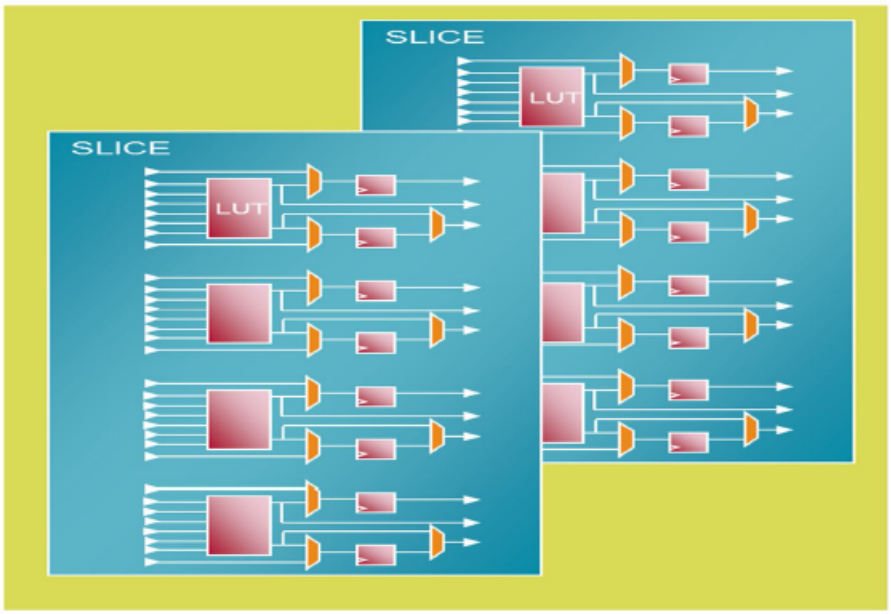
2.7.1.1. Programlanabilir Mantıksal Bloklar (Configurable Logic Blocks-CLB)

FPGA'nın temel yapısını oluşturan programlanabilir mantıksal blokların temelinde, Bakılan Tablo (LUT-Look Up Table), çoklayıcı (MUX-Multiplexer) ve flip flop bulunmaktadır. FPGA'nın kapasitesi genellikle bu mantık hücrelerinin sayısı ile belirlenmektedir. Şekil 2.13'te en temel yapıları gösterilmektedir[10].



Şekil 2.15 Programlanabilir Mantık Bloğunun Yapısı

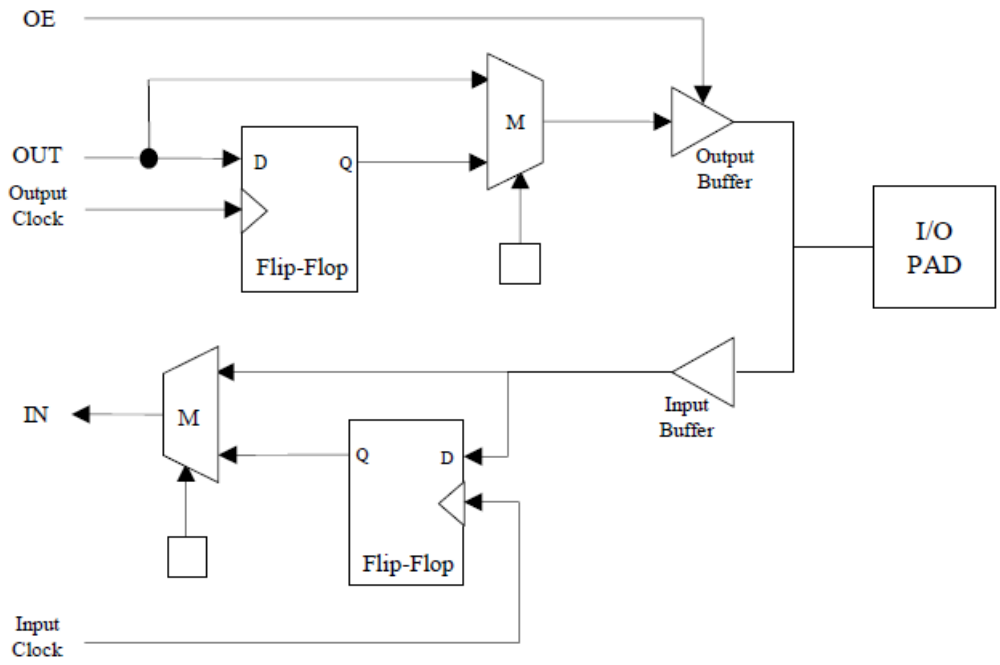
Mantık bloklarında en küçük mantıksal birim, CLB (Configurable Logic Block) olarak adlandırılmaktadır. Şekil 2.16'da Xilinx 7 serisine ait FPGA'ların CLB yapıları görülmektedir. Her bir CLB iki slice'den oluşur. Her bir slice'ın yapısında, altı girişli, dört adet LUT bulunur. Her bir LUT iki adet flip floba bağlıdır. Böylece bir ardışık tasarım oluşturulmaktadır [10].



Şekil 2.16 Programlanabilir Mantıksal Bloğun Yapısı[11]

2.7.1.2. Giriş/Çıkış Blokları (I/O Blocks)

Şekil 2.17'de görülen G/Ç blokları, çipin iç sinyal hatları ile uçları (pinleri) arasında programlanabilir arabirim görevi yapmaktadırlar. G/Ç blokları sayesinde FPGA'ların uçları (pinleri) giriş, çıkış veya çift yönlü olarak programlanabilmektedir. FPGA çipinin türüne göre bir çipteki G/Ç bloğu sayısı (pin sayısı) 1000'li sayılara kadar çıkabilmektedir[12].

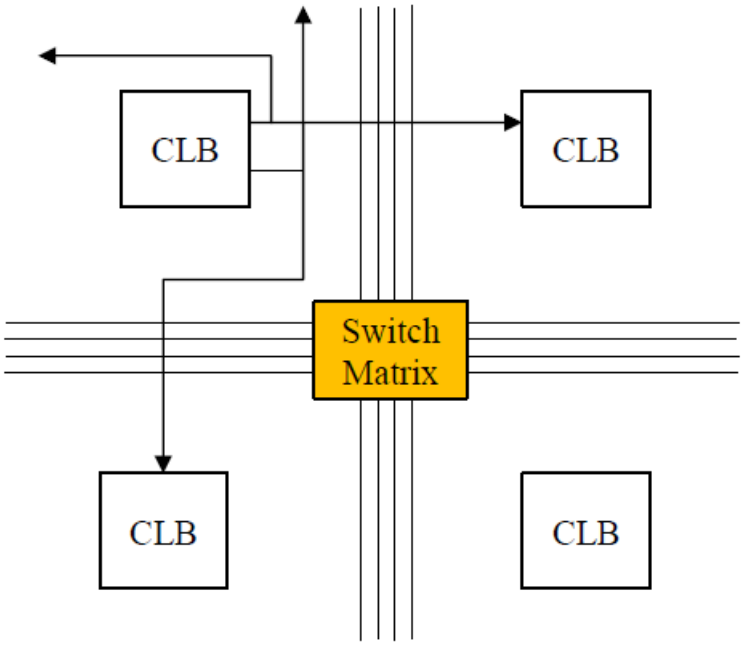


Şekil 2.17 FPGA Programlanabilir G/Ç Bloğu[13]

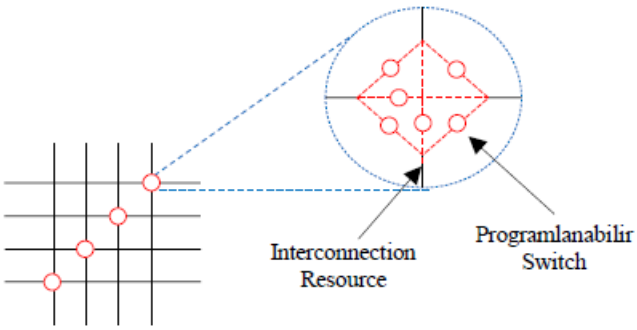
2.7.1.3. Ara Bağlantı Kanalları (Interconnection Resources)

Şekil 2.18'de açık şema ile gösterilen bu birimler hem CLB'ler arasında, hem de CLB'ler ile giriş/çıkış blokları arasında bağlantıları yapılandırmada kullanılmaktadırlar. Programlanabilir olduklarından çok esnek bir yapıya sahiptirler[14].

Şekil 2.19'da yatay ve dikey kanalların birleştiği yerlerde oluşan matris anahtarlar (switch matrix) görülmektedir. Bu anahtarların içinde altı transistorlu (pass transistor) yönlendirme mekanizması bulunmaktadır. Programlanabilme özelliğine sahip olan bu anahtarlar sayesinde, giriş yapılan taraftan kendisine komşu diğer üç tarafa yönlendirme sağlanabilmektedir[15].



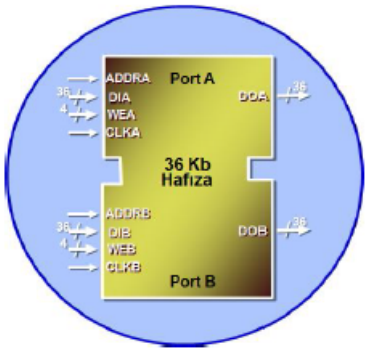
Şekil 2.18 FPGA Programlanabilir Bağlantı



Şekil 2.19 Switch Matrix

2.7.1.4. Blok RAM

Günümüzdeki FPGA tasarımlarda küçük veriler için mantık hücreleri arasındaki küçük bellekler kullanılırken, FPGA'nın büyük bellek ihtiyaçları için ise Blok RAM'ler kullanılmaktadır. Şekil 2.20'de Xilinx 7 serisine ait Blok RAM yapısı görülmektedir.



Şekil 2.20 Xilinx 7 serisi Blok RAM yapısı

2.7.2. FPGA Programlama Teknolojileri

FPGA'lar programlama yöntemlerine göre temel olarak aşağıda belirtildiği gibi iki sınıfa ayrılarak incelenebilmektedirler;

- Bir Kez Programlanabilir FPGA (OTP FPGA- One Time Programmable FPGA)
- Tekrar Programlanabilir FPGA (Re-Programmable FPGA)

2.7.2.1. Bir Kez Programlanabilir FPGA (OTP FPGA)

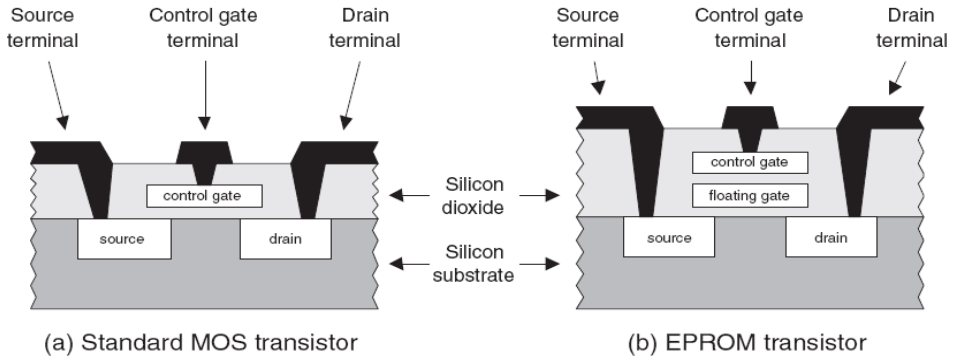
İsminden de anlaşıldığı üzere, bu tür FPGA bir kez programlanabilme özelliğine sahiptir. Yalnızca bir kez programlanabilme özelliği nedeni ile her hangi bir şekilde bir hata yapılması halinde, bu FPGA'nın geri dönüşü olmayacak şekilde kaybına neden olmaktadır.

Bir kez programlanabilir FPGA türlerine, Sigorta (Fuse), EPROM ve Karşıt Sigorta (Anti-Fuse) temelli programlama teknolojileri dahil edilebilmektedir.

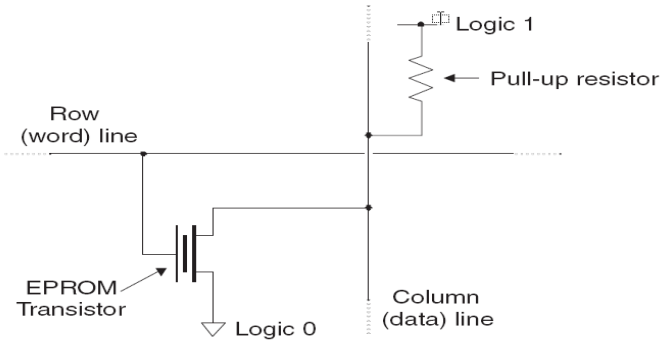
2.7.2.1.1. EPROM Teknolojisi

EPROM temelli FPGA, aslında tam olarak bir kez programlanabilir FPGA sınıfına dahil değildir. Bu tür FPGA, EPROM hücrelerinin yeniden programlanmasını sağlayan Ultraviyole ışınlarını geçirmeyecek şekilde kılıflanması ile bir kez programlanabilme özelliği kazanmaktadır[8].

EPROM transistorü temelde standart MOS transistorü ile aynı yapıya sahiptir. EPROM transistore sadece ikinci bir kapı olarak kayan kapı (floating gate) eklenmiştir. Programlanmamış durumda kayan kapı şarjlı değildir ve kontrol kapısının (control gate) normal çalışmasını etkilemez. EPROM transistorü programlamak için kontrol kapısı ile akıtıcı terminal (drain terminal) arasına 12 Voltluk nispeten yüksek bir gerilim uygulanmaktadır. Programlama sinyali kesildiğinde kayan kapıda negatif şarj kalmakta ve bu şarj normal çalışma ortamında 10 yıldan daha uzun süre kalabilmektedir. Programlanmamış durumda bütün EPROM transistorların kayan kapıları şarjlı değildir. EPROM kayan kapıdaki elektronların deşarj edilmesi ile silinebilmektedir. Bunun için de ultraviyole ışını gereklidir. EPROM'lardaki ana problem quartz camlarından dolayı pahalı olmaları ve ultraviyole ışın ile silinmelerinin uzun zaman almasıdır (yaklaşık 20 dakika). Şekil 2.21 ve Şekil 2.22'de bu yapıların detayları görülmektedir.



Şekil 2.21 Standart MOS Transistor ve EPROM Transistor

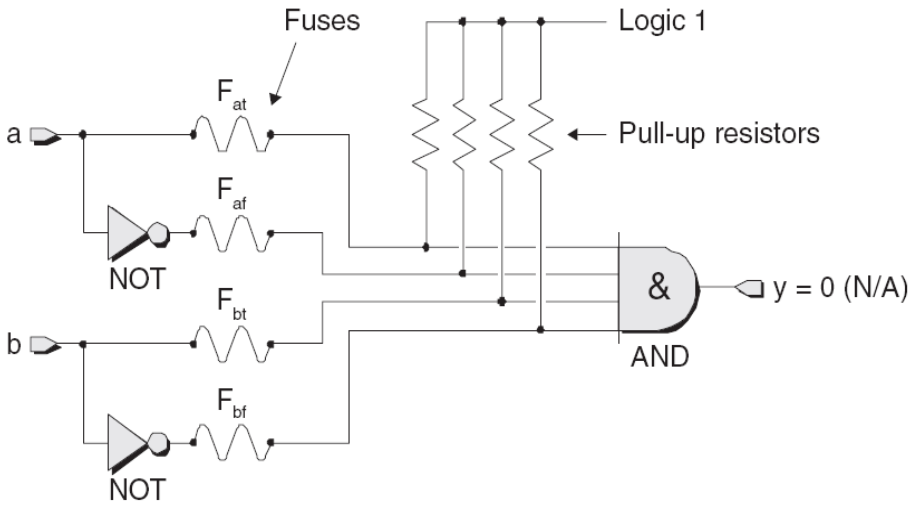


Şekil 2.22 EPROM Transistor Tabanlı Hafıza Hücresi

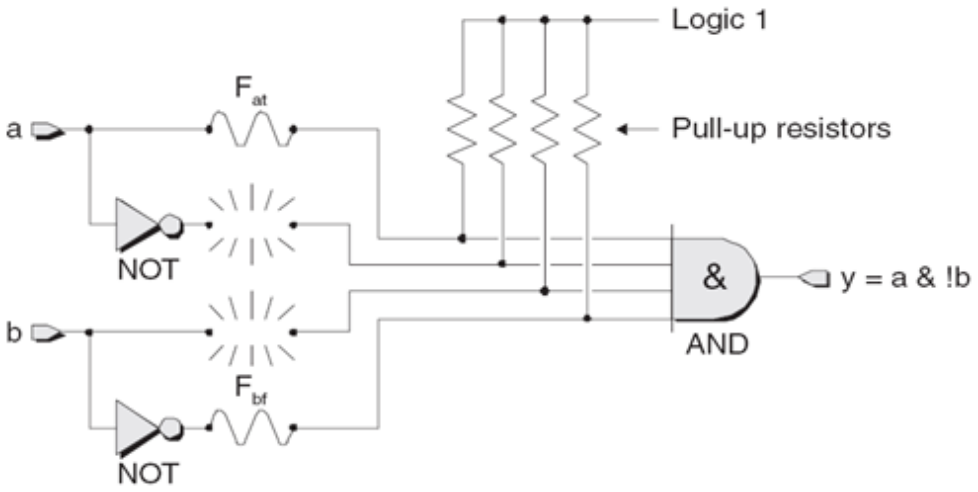
2.7.2.1.2. Sigorta (Fuse) Teknolojisi

Sigorta temelli FPGA programlama teknolojisi ise bipolar teknoloji temelli olması nedeniyle, üretilen çiplerde ihtiyaç duyulan boyut küçüklüğü ve hız ihtiyacını karşılamada yetersiz kalması nedeniyle terk edilmiştir[8].

Sigorta teknolojisi, tasarımcının kendi devresini programlamasına izin veren ilk teknolojilerdendir. Şekil 2.23'te görüldüğü gibi tüm bağlantılar evlerimizde bulunan sigortalar ile aynı çalışma mantığına sahip olan, gelişmiş bir sigorta teknolojisi ile üretilen, sigortalar ile doldurulmuş şekilde üretilmektedirler. Tasarlanması istenilen devrenin oluşturulması için, tasarımcı istediği girişlere yüksek gerilime sahip olan sinyaller göndermektedir. Yani ilgili bağlantıdaki sigortaların eritilerek, açık devre haline gelmesini sağlamaktadır. Şekil 2.24'te programlanmamış sigorta programlama teknolojisi görülmektedir. Programlama işlemi sadece bir kez yapılabilmektedir[7].



Şekil 2.23 Programlanmamış Sigorta Programlama Teknolojisi

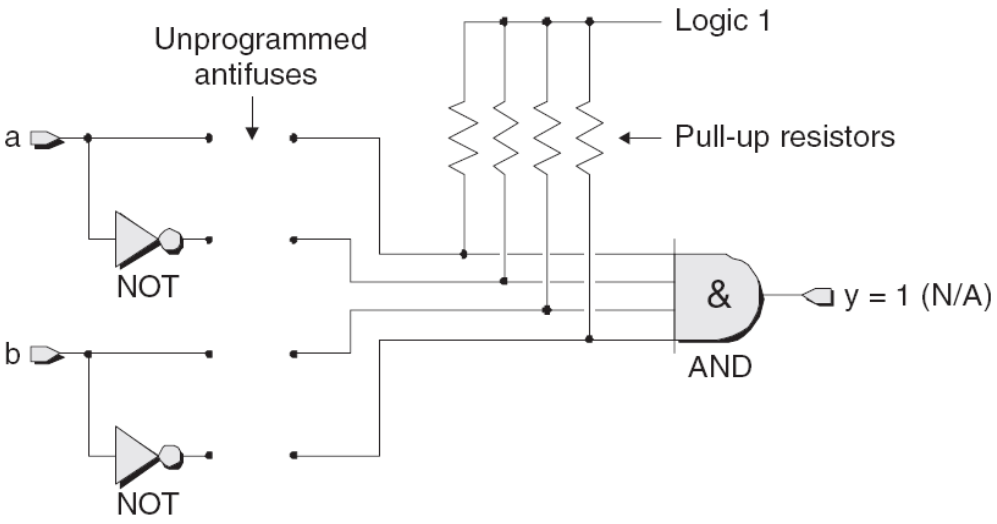


Şekil 2.24 Programlanmış Sigorta Programlama Teknolojisi

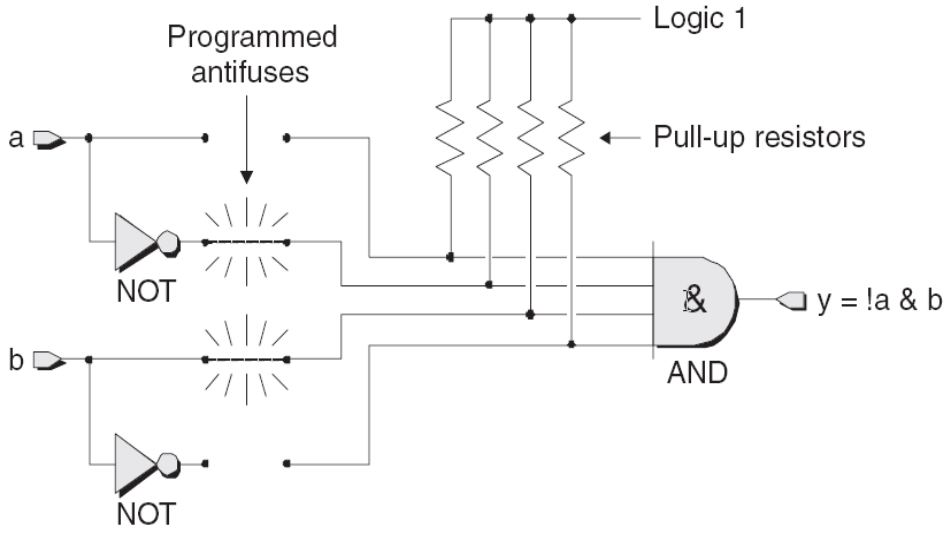
2.7.2.1.3. Karşıt Sigorta Teknolojisi

Bir kez programlanabilir FPGA sınıfının en çok kullanılan türü, karşıt sigorta programlama teknolojisidir. Karşıt sigorta teknolojisi ile programlanabilen FPGA, CMOS teknolojisi kullanması ve veri saklama süresi ile ilk sırada radyasyon olmak üzere çevresel şartlara karşı dayanıklı olduğundan yaygın olarak tercih edilmektedir[16].

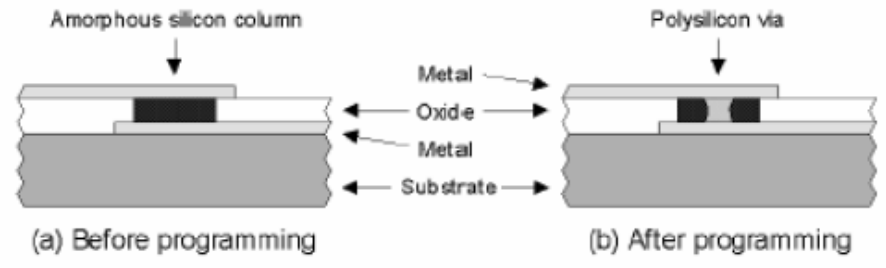
Şekil 2.25'te görüldüğü gibi,sigorta teknolojisinden farklı olarak karşıt sigorta teknolojisinde, var olan bütün bağlantılara karşıt sigortalar yerleştirilmiştir. Karşıt sigortaların yüksek dirençli olmaları nedeniyle programlanmamış halde iken açık devre gibi davranırlar. Yüksek gerilim verilerek, karşıt sigortalar programlanmaktadırlar. Böylece, Şekil 2.26'da görüldüğü gibi aygıtların kapalı devre gibi davranması sağlanmaktadır. Tasarımcı, istediği bağlantıları oluşturarak istenilen tasarımı elde edebilmektedir. Bu tür programlamaya izin veren teknoloji, amorf silikon sütununun yüksek gerilim altında iletken polisilikona dönüşmesidir. Bu teknoloji de sigorta teknolojisinde olduğu gibi bir kez programlanabilir özelliktedir[7]. Şekil 2.27'de karşıt sigorta programlama teknolojisinin, programlamadan önceki ve sonraki hali görülmektedir.



Şekil 2.25 Programlanmamış Karşıt Sigorta Teknolojisi



Şekil 2.26 Programlanmış Karşıt Sigorta Teknolojisi



Şekil 2.27 Karşıt Sigorta Teknolojisinin Programlamadan Önceki ve Sonraki Hali

2.7.2.2. Tekrar Programlanabilir FPGA (Re-Programmable FPGA)

Bu tür FPGA'lar, adından da anlaşıldığı üzere yeniden programlanabilen yapılardır. Bu özelliği nedeniyle tasarımcılar tarafından çok fazla tercih edilmektedirler. Tekrar programlanabilir FPGA'lar programlanan hücrelerinin yapılarına göre;

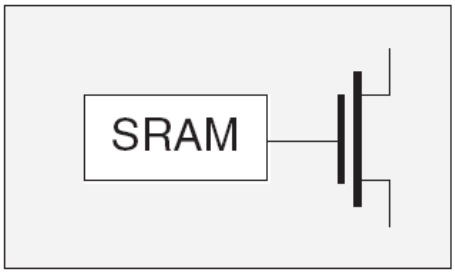
- SRAM Merkezli FPGA (SRAM-Based FPGA)
- Flash Merkezli FPGA (Flash-Based FPGA)

olmak üzere iki kısma ayrılmaktadırlar. Güç kesintisi olduğunda, yapılandırmalarını tutabilme özelliklerine göre uçucu (volatile) ve uçucu olmayan (non-volatile) özellikte olanları vardır[8].

2.7.2.2.1. SRAM Merkezli FPGA (SRAM-Based FPGA)

Güç kesintisi olduğunda yapılandırmalarını kaybetmeleri nedeniyle yapılandırma dosyaları, flash bellek içinde tutulmaktadır. Sistemin her açılışında bu hafızadan yapılandırma dosyaları yüklenmektedir. SRAM Merkezli FPGA'da yapılandırma dosyasının tutulduğu bellek çip dışında ise "Harici Hafızalı SRAM Merkezli FPGA" olarak isimlendirilmektedirler. Özellikle gelişmiş uygulamalarda kullanılan, yüksek işlem gücü ve yüksek hafıza sağlayan SRAM Merkezli FPGA'lar harici hafızalıdır. Eğer yapılandırma dosyasının tutulduğu bellek, FPGA ile aynı çipte yer alıyorsa "Dahili Hafızalı SRAM Merkezli FPGA" olarak isimlendirilmektedirler. Bu tür FPGA'larda her güç kesintisi olduktan sonra, FPGA yeniden başlatıldığında, tekrar

yapılandırma dosyasının yüklenmesine gereksinim duyulması nedeniyle, aşırı güç tüketimi ve zaman gecikmesi olmaktadır. Ek olarak kopyalama saldırısına karşı açıktırlar[8]. Şekil 2.28'de SRAM yapısı görülmektedir.



Şekil 2.28 SRAM Yapısı

2.7.2.2.2. Flash Merkezli FPGA (Flash-Based FPGA)

Flash teknolojisi ile programlanabilmektedirler. SRAM Merkezli FPGA, sistemin her açılışında SRAM hücrelerini yapılandırmak için, yapılandırma dosyasının tutulduğu flash hafızadan çekerek kullanırken, FLASH Merkezli FPGA ise, flash teknolojisini ana yapılandırma işlemi için kullanmaktadır. Bu nedenle elektriksel olarak silinmediği sürece üzerindeki yapılandırmayı kaybetmemektedir. Böyle bir artısının olması nedeniyle, SRAM Merkezli FPGA ile karşılaştırıldığında, çok daha kısa sürede çalışabilir hale gelmekte ve çok daha az güç tüketmektedir. FLASH Merkezli FPGA'yı yalnızca ACTEL üretmektedir. Bu ürünü, dahili hafızalı SRAM Merkezli FPGA'dan ayırmak için True FLASH Merkezli FPGA ifadesi yaygın olarak kullanılmaktadır[8].

2.7.2.3. FPGA Programlama Teknolojilerinin Karşılaştırılması

Piyasada en çok kullanılan FPGA'ların, programlama teknolojilerinin birbirilerine göre üstün özellikleri ve üstün olmayan özellikleri Tablo 2.1'de görülmektedir.

Tablo 2.1 FPGA Programlama Teknolojilerinin Karşılaştırılması[6]

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

2.7.3. Güvenlik Açısından FPGA

Bir tasarım gerçekleştirmek için kullanılacak FPGA türü seçilirken, bu seçimde önem arz eden hafıza hız vb. gibi çeşitli parametreler bulunmaktadır. Özellikle gizlilik açısından önemli verilerin işlendiği FPGA sistemlerinde, bu parametrelerin en başında güvenlik gelmektedir. Bu bölümünde elimizden geldiğince güvenlik açısından FPGA'ları incelemeye çalışacağız.

İlk olarak yapılabilir saldırı türleri ele alınmaya çalışılacak, sonrasında FPGA türleri için bu saldırıların gerçeklenebilirliği ve alınan önlemler üzerinde durulacaktır[8].

2.7.3.1. Kopyalamaya Karşı Koruma

FPGA üzerinde yapılması en kolay saldırı yöntemidir. Bu işlem, içeriğin ne olduğuna bakılmaksızın yapılandırma dosyalarının çeşitli yöntemlerle birebir olarak alınması ve saldırganın istediği şekilde bu yapılandırma dosyalarını kullanması olarak açıklanabilir. Kopyalamanın en yaygın gerçeklenme şekli yapılandırma dosyalarının tutulduğu hafızanın okunması ya da FPGA sistemi çalışmaya başlarken hafızadan yapılandırma dosyalarının alındığı anda veri ara yüzünün dinlenmesidir. Kopyalama işlemi ile elde edilen veriler kullanılarak, farklı bir FPGA yapılandırılabilmekte ve bu FPGA üzerinde analizler yapılabilmektedir. Bu işlem önemli derecede güvenlik kayıplarına neden olabilmektedir. Saldırı yöntemi incelendiğinde, SRAM Merkezli FPGA'nın bu saldırı yöntemine en fazla hedef olabileceği ortaya çıkmaktadır. Özellikle harici hafıza yapısını kullanmakta olan SRAM Merkezli FPGA bu saldırı yöntemine tam anlamıyla açıktır. Dahili hafıza yapısını kullanmakta olan FPGA ise, yapılacak dinlemeye karşı daha dayanıklıdır. Buna karşın, bu FPGA türü ek bir güvenlik önlemi olmadan uzun süre kullanılmıştır. Ancak güvenlik konusundaki kaygıların artması üzerine "bitstream encryption" yöntemi ile korunan bir model geliştirilmiştir[8].

Bu modelde yapılandırma dosyaları hafıza elemanları üzerinde şifreli olarak saklanmakta, sisteme güç verildiğinde FPGA öncelikle bu şifreli dosyaları alıp üzerinde sakladığı algoritma ve anahtarlar ile çözmekte ve bu şekilde kendini yapılandırmaktadır[8].

Kopyalama saldırılarının farklı bir yöntemi ise FPGA'nın programlandıktan sonra yapılandırma dosyalarının tekrar okunması ile ortaya çıkmaktadır. Çünkü bir çok FPGA hata ayıklama (debugging) işlemi açısından kolaylık sağlanması amacı ile yapılandırma dosyalarının JTAG ara yüzünden geri okunabilmesine imkan sağlamaktadır. Bazı SRAM Merkezli FPGA'larda güvenlik biti (security bit) önlemi alınmıştır. Tasarımcı tarafından güvenlik bitleri aktif hale getirilmezse geri okuma işlemi yapılmaktadır. Fakat tasarımcı güvenlik bitlerini aktif hale getirirse, geri okuma işlemi kapatılmış olmaktadır. Bunun için güvenlik bitlerinin FPGA'daki yerleri bulunarak aktif hale getirilmelidir. Bazı SRAM Merkezli FPGA'larda bu güvenlik önlemi artırılarak JTAG ara yüzleri hassas hale getirilmiştir. Geri okuma işlemi

tasarımcı tarafından engellenip, her hangi bir nedenle geri okuma işleminin yapılması denenirse, FPGA, içinde var olan önemli bilgileri otomatik olarak silebilmektedir. FLASH Merkezli FPGA yapısında bu işlem için farklı sayılarda bitlerden oluşan kilit (lock) adı verilen yapılar bulunmaktadır[8].

Örneğin bu tez kapsamında da kullanılan Xilinx Spartan 3 serisi FPGA'larda kopyalamaya karşı engel olması amacı ile, yapılandırma dosyasının tutulduğu hafıza FPGA'nın içine yerleştirilmiştir (dahili hafızalı SRAM-Based FPGA uygulaması). Fakat asıl önlem olarak, FPGA'nın içine yüklenen bilgilerin ilgili FPGA ile ilintilenmesi sayesinde, kopyalama işlemi olsa dahi, farklı FPGA'da çalışmasını engelleyen "Çip DNA" önlemi alınabilmektedir. FPGA'nın üretimi sırasında, FPGA içine 57 bit ID değeri yerleştirilmektedir. Tasarımcı, FPGA'ya istenilen tasarımı yüklerken, yapılandırma dosyasını bu 57 bitlik ID ile eşleştirebilmektedir. Bu önlemin daha ileri düzeyi, Spartan 3AN FPGA'da bulunan, Çip DNA'sına ek olarak Fabrika Bellek ID değerinin de sisteme eklendiği önlemdir. Kısaca 57 bitlik güvenlik düzeyinin 70 bayta kadar çıkabildiğini söyleyebiliriz[22,8]. Ek olarak, Spartan 3 serisi FPGA'larda kopyalama önlemi olarak, geri okuma ve tekrar yapılandırılabilme özellikleri pasif edilebilmektedir[8].

2.7.3.2. Tersine Mühendislik İçin Karşı Koruma

Bu yöntemde, yapılandırma dosyasının analiz edilerek FPGA'daki tasarımın anlaşılması için çalışılmaktadır. Bu çalışmalar, kopyalama yöntemi ile karşılaştırıldığında, çok daha fazla zaman ve teknoloji gerektiren çalışmalardır. Önceki aşamalarda bahsedilen her tür FPGA bu saldırı yönteminin hedefi haline gelebilmektedir. Özellikle, SRAM Merkezli FPGA'ların yapılandırmalarının tutulduğu hafızaların (PROM) bu saldırı yöntemi için en uygun FPGA türü olması nedeniyle, bu alanda en zayıf olarak görülmektedirler. Fakat Kopyalamaya Karşı Koruma alt başlığı altında bahsedilen güvenlik biti vb. gibi yöntemler bellekler üzerinden bu saldırıların yapılmasını engellemektedir[8].

ASIC tasarımlarındaki bağlantı ve kanallar, FPGA'lar ile karşılaştırıldığında daha belirgin olduğu için, bu tür saldırılara karşı ASIC'lar, FPGA'lara göre daha zayıftırlar. Karşıt Sigorta FPGA'da ise yapılandırma sırasında oluşturulan bağlantıların incelenerek anlam çıkarılması çok zordur. Bundan dolayı, Karşıt Sigorta FPGA'da böyle bir saldırı gerçekleştirebilmek neredeyse imkânsızdır. Bu saldırı yöntemine karşı en fazla dayanıklılık sağlayan FPGA türü ise FLASH Merkezli FPGA'dır. Çünkü FLASH Merkezli FPGA programlanmasında eleman üzerinde yer alan anahtarlar fiziksel olarak durum değiştirmemektedirler. Yalnızca akış kapısı (floating gate) bölgelerindeki elektronların yoğunluğu değişmektedir[8,18].

2.7.3.3. Değiştirmeye Karşı Koruma

Bu yöntemde FPGA üzerindeki yapılandırmanın değiştirilmesi hedeflenmektedir. Bu değiştirme işlemi ile FPGA'nın çalışma sisteminin bozulması ve gizli bilgilerin ortaya çıkmasının sağlanması mümkün olabilmektedir. Karşıt Sigorta FPGA programlama teknolojisinden dolayı bu tür saldırılara karşı dirençlidir. SRAM Merkezli FPGA önlem alınmaksızın kullanıldığında bu saldırı yöntemine açık olarak görünmektedir. Fakat bit akışı şifreleme (bit stream encryption) adı verilen yöntem ile dosyaların şifreli olarak FPGA'ya yüklenmesi ve FPGA üzerinde gizli anahtarla açılarak kullanılması nedeniyle bu saldırı yöntemine karşı önlem alınmış olunmaktadır. Saldırgan tarafından değiştirilmiş bir tasarım, FPGA'ya yüklense dahi şifre çözme aşaması nedeni ile bu tasarım anlamsız hale gelerek FPGA üzerinde çalışmamaktadır[8].

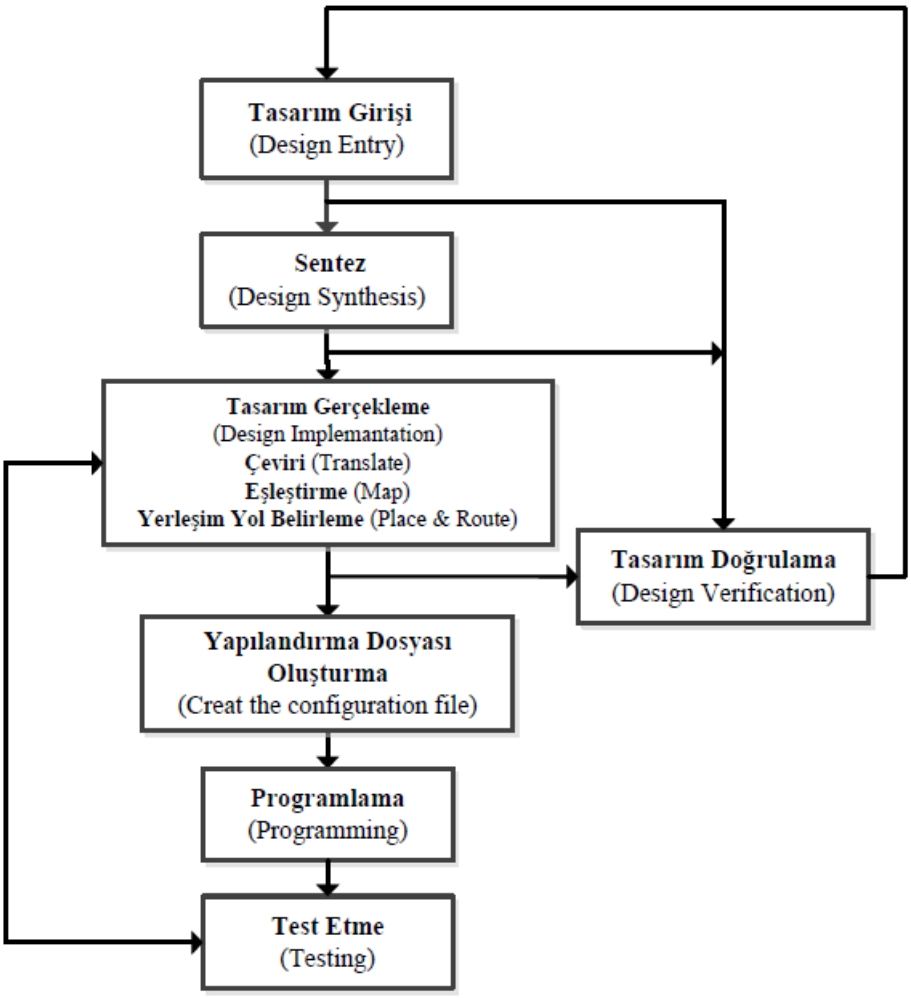
FLASH Merkezli FPGA, tekrar programlanabilme özelliğinden dolayı bu tür saldırı yöntemine açıktır. Fakat, kilit (lock) önlemiyle bu tür saldırıların engellenmesi hedeflenmektedir[8,17].

2.7.3.4. Sonuç

FPGA seçiminde hız, güvenlik, kullanım kolaylığı, bellek kapasitesi gibi dikkate alınması gereken bir çok özellik bulunmaktadır. Yalnızca güvenlik özelliği ön plana alınarak belirli bir model veya türde FPGA üzerinde karar kılmak doğru değildir. Bu alt bölüm içeriğinde verilen bilgiler incelendiğinde en fazla saldırılara maruz kalan FPGA türünün SRAM Merkezli FPGA olduğu anlaşılmaktadır. Böyle olmasına rağmen en fazla kullanılan FPGA türünün de bu tür olduğu sonucuna (FPGA piyasasının yaklaşık %80'ine sahip olan Xilinx ve Altera'nın neredeyse yalnızca SRAM Merkezli FPGA üretmesinden) ulaşılmaktadır. Buradan SRAM Merkezli FPGA kullanımının ne derece yaygın olduğu anlaşılmaktadır. Bunun nedeni ise, SRAM Merkezli FPGA'nın güvenlik dışındaki işlem gücü, kapasite, kullanım kolaylığı vb. gibi özellikleri açısından tasarımcıya ve kullanıcıya sağladığı artı özellikleridir. Bu artı özellikleri nedeni ile SRAM Merkezli FPGA, veri gizliliğinin çok önemli olduğu tasarımlarda dahi tercih edilmektedir[8].

2.7.4. FPGA Tasarım Akışı

Bu kısım, tasarım girişinden itibaren FPGA'ya yüklenecek kodun üretilmesine kadar olan basamakları içermektedir. Şekil 2.29'da görülen tasarım akışındaki önemli basamaklar ele alınacaktır.



Şekil 2.29 FPGA Tasarım Süreci[12]

2.7.4.1. Tasarım Girişi (Design Entry)

Tasarlanmak istenilen uygulama belirlendikten sonra, uygulamanın sistem tasarımı yüksek seviyeli donanım tanımlama dillerinden (HDL) biri ile yani VHDL veya Verilog HDL kullanılarak veya şematik tasarım yapılarak tasarlanabilmektedir. Şekil 2.30'da Verilog HDL ile Xilinx ISE'de yazılmış basit bir kod parçası görülmektedir.

```
21 module Test(
22     input In0,
23     input In1,
24     output Out0
25     );
26
27 assign Out0 = In0 ^ In1;
28
29 endmodule
```

Şekil 2.30 Basit Verilog HDL Kod Parçası

2.7.4.2. Sentez (Synthesis)

Bu basamak, tasarım girişini devre bağlantılarına dönüştürür. Sentez araçları, donanım tanımlama dili kodunu ve seçilen üreticinin FPGA modelinin bilgilerini alarak, donanım tanımlama dili ile oluşturulan tasarımı gerçekleştirmek için FPGA modelinin dahili bağlantılarının nasıl olmasını gerektiğini gösteren mantıksal kapı seviyesinde bağlantı listesi (Gate Level Netlist) oluşturmaktadır.

Ayrıca sentezleme işlemi, devreye ait lojik kısıtlamalar (Giriş/Çıkış uçları, zamanlama, yerleştirme, saat frekansı, kritik yollar gibi) kullanıcı kısıtlama dosyası (user constraints file, Xilinx için) dikkate alınarak yapılmaktadır. Yine bu aşamada, istenilen fonksiyonların en iyi şekilde gerçeklenebilmesi için gerekli lojik indirgemeler (logic optimization) de yapılmaktadır[12].

2.7.4.3. Eşleştirme (Map)

Bu adımda, yapılan tasarım ile hedef FPGA'nın kaynakları arasında eşleştirme işlemi yapılmaktadır. Yani tasarımımızın bölümlerinin FPGA üzerinde kullanacağı (gerçekleneceği) bölümler belirlenmektedir.

2.7.4.4. Yerleşim ve Yol Belirleme (Place & Route)

Bu aşamayı bir nevi optimizasyon aşaması olarak adlandırabiliriz. Bu basamağın, yerleşim (place) aşamasında sentezlenmiş ağ listesi alınarak, gerçeklenmesi istenilen tasarımın FPGA'daki CLB'lere en uygun şekilde yerleşeceği yerler

seçilmektedir. Yol belirleme (Route) aşamasında ise bloklar arasındaki bağlantı yolu, olabilecek en kısa seviyeye indirgenerek bağlantı oluşturulmaktadır. Ayrıca tasarımın çalıştırabileceği en yüksek saat frekansı da tasarımda oluşturulan kritik yola göre belirlenmektedir.

2.7.4.5. Benzetim (Simulation)

Benzetim işlemi tasarım girişinden sonra veya yerleşim işleminden sonra da yapılabilmektedir. Eğer tasarım işleminden sonra yazılan kodun doğru işlemi yapıp yapmadığını test etmek istenilerek benzetim yapılırsa, buna fonksiyonel benzetim, yerleşim basamağından sonra yapılırsa, buna zamanlama benzetimi denilmektedir. Xilinx ISE'de benzetim yapılabileceği gibi, diğer firmalar tarafından sunulan programlarda da (Modelsim vb.) benzetim yapılabilmektedir.

2.7.4.6. Kaynak Kod Oluşturma (Bitstream Generation)

En son aşamada FPGA'ya yüklenecek biçimde olan yapılandırma dosyası (bitstream) oluşturulmaktadır.

FPGA'ya yüklenecek yapılandırma dosyası oluşturulduktan sonra, JTAG (Joint Test Action Group (Ortak Test Eylem Grubu)) ara yüzü ile bu dosya FPGA'ya yüklenebilmektedir. SRAM tabanlı FPGA entegreleri bu dosyayı her açılışta uçucu olmayan bir bellekten okuyup yapılandırmalarını sağladıkları gibi, Flash tabanlı FPGA entegreleri bu kodu içlerinde de tutabilmektedirler. FPGA'lar JTAG ara yüzleri sayesinde karta lehimli iken bile tekrar tekrar programlanabilmektedirler[12].

2.7.5. FPGA'nın Donanım Tanımlama Dili ile Programlanması

FPGA üreten başlıca firmalar Xilinx, Altera, Lattice, Microsemi, Quicklogic ve SiliconBlue'dur. FPGA üreten hemen hemen her firma kendi tasarım araçlarını da sunmaktadır. FPGA piyasasının yaklaşık %80'ine Xilinx ve Altera firmaları hakim olduğu için bu kısımda Xilinx ve Altera'dan kısaca bahsedilecektir[23,24]**.** Altera'nın ürettiği FPGA'larda proje tasarımı yapmak için "Quartus II Design Software" yazılımı kullanılmaktadır. Xilinx'in ürettiği FPGA'larda proje tasarlamak için ise "ISE Design

Suite" veya daha iyi sentezleme yapabilmesi amacıyla geliştirilen "Vivado Desing Suite" kullanılmaktadır.

Bu belirtilen tasarım araçlarında istenilen tasarımın yapılabilmesi için, şematik tasarım veya donanım tanımlama dili (HDL-Hardware Description Language) ile gerçekleştirilen tasarım kullanılabilmektedir. Şematik tasarım, esnek olmaması ve karmaşık devrelerin tasarlanmasının çok zaman alması nedeniyle profesyonel tasarımlarda neredeyse hiç tercih edilmemektedir.

İlk modern Donanım Tanımlama Dili (DTD), Verilog 1985 yılında "Gateway Design Automation" tarafından geliştirilmiştir. Daha sonra Cadence Design Systems Verilog-XL'nin haklarını satın almıştır. Verilog, yapılan çalışmalar sonucunda aralık 1995'te IEEE standardı haline getirilmiştir. Amerika Savunma Dairesi'nin geliştirilmesi isteğiyle 1987 yılında VHDL (Very High Speed Integrated Circuit Hardware Description Language) ortaya çıkmıştır. Sentezleme araçları, DTD kaynak dosyalarını derleyerek RTL (Register Transfer Level) seviyesinde üretime hazır netlist tanımlamalarını oluşturmaktadır.

Verilog HDL ve VHDL donanım tanımlama dilleri yıllardır ön planda yer alan DTD'lerdir. İkisi de birbirine yakındır. Fakat Verilog HDL'in kodları C diline yakınlığı ile bilinmektedir. Tez kapsamında, Verilog HDL kullanıldığı için, genel yapısına kısaca değinilmiştir.

2.7.5.1. Verilog HDL

Verilog HDL'de tasarlanan her sistem modüller ve bu modüller arasındaki bağlantılar ile tanımlanmaktadır. Modüller giriş/çıkış uçları (portları) ve bu giriş sinyalleri ile çıkış sinyallerinin belirlendiği içerikten oluşur. Verilog'un 1995 sürümünde portların adları modül başlığında, portların yönleri ve boyutları ise modülün içinde tanımlanmaktaydı. Fakat Verilog'un 2001 sürümünde portların yön ve boyutlarına ilişkin tanımlamalar modül başlığında yapılabilmektedir. Örnek bir modül aşağıdaki gibi tanımlanabilmektedir.

```
module andgate(
                input a,
                input b,
```

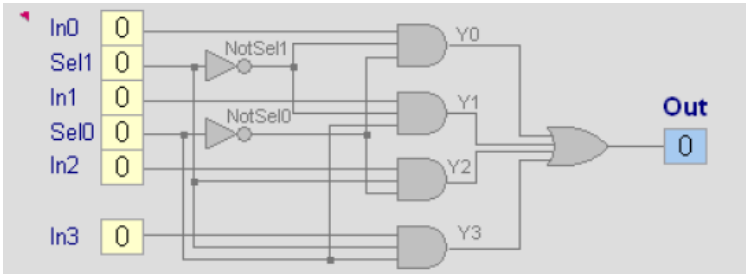
```
                output y
            );
assign y = a & b;
endmodule
```

Her modül, endmodule kodu ile sonlandırılmaktadır. Verilog büyük küçük harfe duyarlıdır. Bu nedenle kod kelimeleri küçük harf ile yazılmalıdır. Verilog'da bir modülün içeriği üç farklı şekilde tasarlanabilmektedir. Bunlar;

- Yapısal Biçim (Structural Style)
- Veri Akışı Biçimi (Dataflow Style)
- Davranışsal Biçim (Behavioral Style)'dir.

2.7.5.1.1. Yapısal Biçim (Structural Style)

Ön tanımlı ilkel kapı (Ör: Not, And, Or kapısı) türünden veya alt modüllerden oluşmaktadır. Ön tanımlı ilkel kapı kullanılırken ilk yazılan değişken genellikle çıkış olarak tanımlanır. Şekil 2.31'de Verilog HDL'in yapısal biçimi görülmektedir.




```
module mux_4_to_1 (Out,In0,In1,In2,In3,Sel1,Sel0);
output Out;
input In0, In1, In2, In3, Sel0, Sel1;

wire NotSel0, NotSel1;
wire Y0, Y1, Y2, Y3;

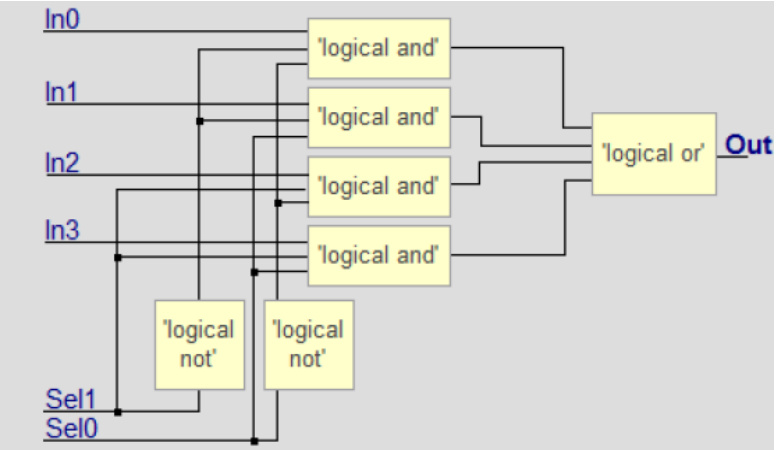
not (NotSel0, Sel0);
not (NotSel1, Sel1);
and (Y0, In0, NotSel1, NotSel0);
and (Y1, In1, NotSel1, Sel0);
and (Y2, In2, Sel1, NotSel0);
and (Y3, In3, Sel1, Sel0);
or (Out, Y0, Y1, Y2, Y3);

endmodule
```

Şekil 2.31 Verilog HDL'in Yapısal Biçimi[25]

2.7.5.1.2. Veri Akışı Biçimi (Dataflow Style)

Bu tasarımda, giriş sinyallerinin bir fonksiyonu olarak çıkış sinyalleri yazılmaktadır. Çok büyük ve karmaşık tasarımlarda anlaşılabilirliği zor olduğu için hata çözme işlemi çok uzun zaman alabilmektedir. Şekil 2.32'de Verilog HDL'in veri akış biçimi görülmektedir.




```
module mux_4_to_1 (Out, In0, In1, In2, In3, Sel1, Sel0);

output Out;
input In0, In1, In2, In3, Sel0, Sel1;

assign Out = (~Sel1 & ~Sel0 & In0) | (~Sel1 & Sel0 & In1)
             | (Sel1 & ~Sel0 & In2)| (Sel1 & Sel0 & In3);

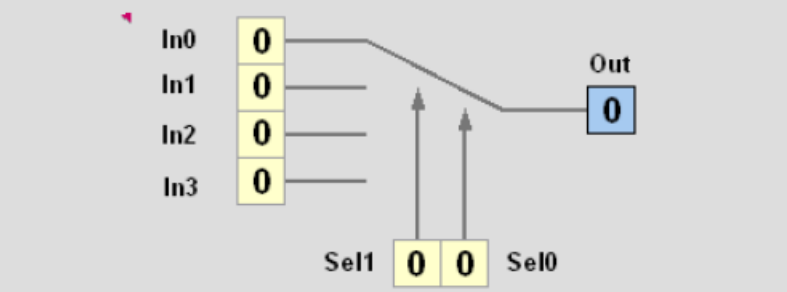
endmodule
```

Şekil 2.32 Verilog HDL'in Veri Akış Biçimi[25]

2.7.5.1.3. Davranışsal Biçim (Behavioral Style)

Bu tasarımda, always kodları içinde case veya if-else yapıları kullanılarak tasarım yapılmaktadır. Fakat always kodları içersinde wire (hat) veri tipinin kullanılamayacağı, reg (yazmaç) ve integer (tamsayı) veri tipinin sinyallere kullanılabileceği unutulmamalıdır. always kod başlığında yer alan değişkenlerin durumlarında bir değişiklik olduğunda, always içerisindeki kodlar tekrar çalıştırılmaktadırlar. Ayrıca, always kodu ile iki tür atama yapılabilmektedir. "<=" işareti ile yapılan atama işlemleri paralel bir şekilde yani aynı anda yapılırlar. "=" işareti

ile yapılan atama işlemleri ise C programlama dilinde olduğu gibi sıralı bir şekilde yapılmaktadır. Şekil 2.33'te Verilog HDL'de davranışsal biçim görülmektedir.




```
module mux_4_to_1 (Out,In0,In1,In2,In3,Sel1,Sel0);
output Out;
input In0, In1, In2, In3, Sel0, Sel1;
reg Out;

always @(Sel1 or Sel0 or In0 or In1 or In2 or In3)
  begin
    case ({Sel1, Sel0})
    2'b00 : Out = In0;
    2'b01 : Out = In1;
    2'b10 : Out = In2;
    2'b11 : Out = In3;
    default : Out = 1'bx;
    endcase
  end

endmodule
```

Şekil 2.33 Verilog HDL'de Davranışsal Biçim[25]

3. GELİŞMİŞ ŞİFRELEME STANDARDI (AES)

Kriptoloji, Yunanca krypto's (saklı) ve lo'gos (kelime) kelimelerinin birleşimi sonucu oluşturulmuştur ve iletişimde gizlilik bilimi olarak değerlendirilmektedir[26].

İki nokta arasındaki haberleşmede veya sistemler arasındaki bağlantılarda verinin karşı tarafa güvenli bir şekilde ulaştığından emin olmak gereklidir. Bu gönderilen verinin şifrelenmesi ile sağlanabilmektedir. Bu şekilde, açık haberleşme kanalları kullanılarak, verinin güvenli bir şekilde hedefe ulaştırılması sağlanmaktadır. İletişimde açık haberleşme kanalı kullanılıyorsa, gizli tutulmak istenilen bilginin yetkisiz kişi veya kişiler tarafından dinlenebileceği veya haberleşme kanalına girilerek (araya girme) verinin bozulabileceği veya değiştirilebileceği (yanlış verinin gönderilmesi) düşüncesi önemli bir sorun oluşturmaktadır[27].

Şifreleme, askeri ve diplomatik haberleşmede güvenliği sağlamak için yüzyıllardır kullanılmaktadır. Şifrelemeye günümüzde özel sektörde de ihtiyaç duyulmaktadır. Finansal işler, sağlık hizmetleri gibi konularda bilgisayarlar arasındaki haberleşme açık kanallar kullanılarak yapılmaktadır. Bu açık kanallar kullanılırken işlemlerin güvenli ve gizli bir şekilde yapılabilmesi için şifrelemeye gerek duyulmaktadır[28].

Şifreleme algoritmalarını yapısal olarak simetrik ve asimetrik olmak üzere iki gruba ayırabiliriz. Simetrik şifreleme algoritmalarında, verinin şifrelenmesi ve şifrenin çözülmesinde bir anahtar, asimetrik şifreleme algoritmalarında ise şifreleme ve şifre çözmede iki farklı anahtar kullanılmaktadır.

Amerika Birleşik Devletleri'nde (ABD) kurulan Ulusal Standartlar ve Teknoloji Enstitüsü (NIST-National Institute of Standards and Technology), 1977 yılında simetrik şifreleme algoritması olan DES'i (Data Encryption Standard) bir veri şifreleme standardı olarak belirlemiştir. Teknolojinin gelişmesiyle birlikte güçlü donanımsal özelliklere sahip bilgisayarların üretilmesi ve kriptanalistlerin uzun süren çalışmaları sonucunda DES kırılmıştır. Sonrasında DES algoritmasını tekrarlayarak işlem yapan TDES (Triple

DES) ortaya atılmış, fakat o da kırılarak güvenirliğini kaybetmiştir. Bu nedenlerle NIST, DES'in yerine kullanılması planlanan, Gelişmiş Şifreleme Standardının (AES-Advanced Encryption Standard) geliştirilmesi amacıyla bir yarışma düzenlemiştir. Yarışmanın birinci turunda on beş aday algoritma değerlendirmeye alınmış, ikinci turunda ise algoritma sayısı beşe indirgenmiştir. Seçilen bu beş algoritmanın adları ve geliştiricileri aşağıda görülmektedir.

- MARS - IBM tarafından geliştirilmiştir.
- RC6 - Ronald Rivest tarafından geliştirilmiştir.
- Rijndael - Vincent Rijmen ve Joan Daemen tarafından geliştirilmiştir.
- Serpent - İngiltere, İsrail ve Norveç'ten üyeleri olan bir takım tarafından geliştirilmiştir.
- Twofish - Bruce Schneier tarafından geliştirilmiştir.

Dört yıl gibi uzun bir zaman alan değerlendirmeler sonrasında Vincent Rijmen ve Joan Daemen tarafından geliştirilen Rijndael algoritması Gelişmiş Şifreleme Standardı (AES) olarak belirlenmiş ve NIST tarafından basılan FIPS 197 (Federal Information Processing Standard)[29] belgesinde AES'nin tüm özellikleri açıklanmıştır[29,30].

Son elemeye kalan bu beş algoritmanın hangi açılardan incelenemeye tabii tutuldukları ve aldıkları puanlar Tablo 3.1'de görülmektedir.

Tablo 3.1 Son Elemeye Kalan Beş Algoritmanın Hangi Açıdan İncelendikleri[31]

	Algoritma				
Kategori	**MARS**	**RC6**	**Rijndael**	**Serpent**	**Twofish**
Genel güvenlik	3	2	2	3	3
Uygulama güvenliği	1	1	3	3	2
Yazılım performansı	2	2	3	1	1
Akıllı kart performansı	1	1	3	3	2
Donanım performansı	1	2	3	3	2
Tasarım özellikleri	2	1	2	1	3

3.1. AES'nin Genel Yapısı

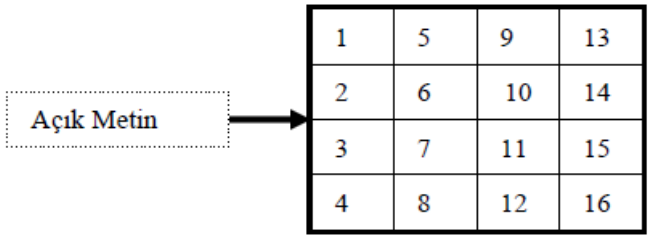
AES algoritması, 128 bitlik veri bloklarını, 128, 192 ve 256 bitlik anahtarlar ile sırasıyla 10, 12 ve 14 adımda şifreleyen simetrik bir blok şifreleme algoritmasıdır. Tur sayısı, şifreleme işlemi için kullanılan anahtar uzunluğuna bağlıdır. Tablo 3.2'de AES anahtar boyutuna göre, blok boyutu, anahtar uzunluğu ve tur sayısının karşılaştırılması görülmektedir. 128 bit anahtar uzunluğu için 10 tur gereklidir(Nr=10)[32].

Tablo 3.2 AES Anahtar Boyutuna Göre, Blok Boyutu, Anahtar Uzunluğu ve Tur Sayısının Karşılaştırılması

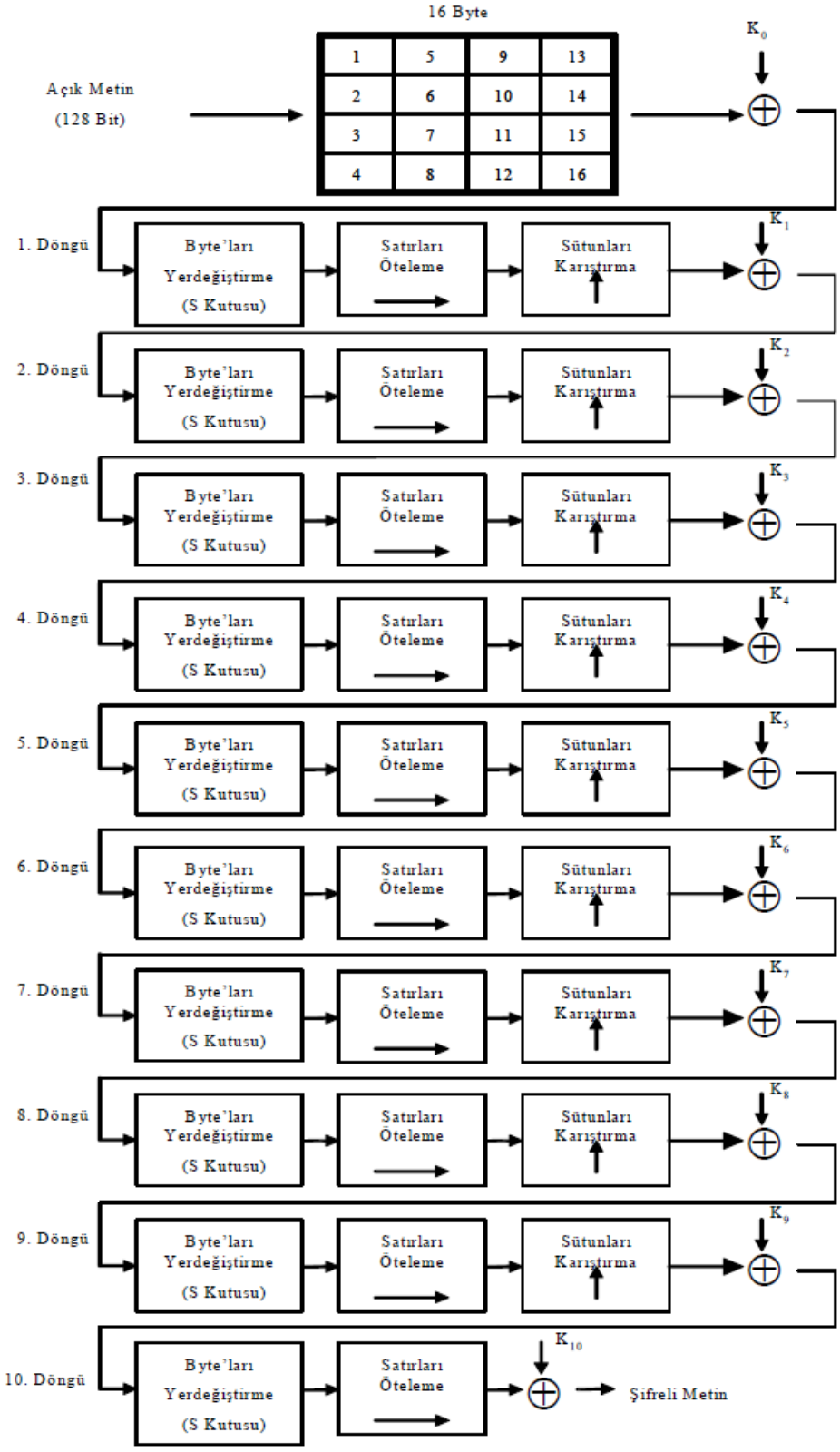
Type	**Block Size N_b Words**	**Key Length N_k Words**	**Number of Rounds N_r**
AES-128 bits key	4	4	10
AES-192 bits key	4	6	12
AES-256 bits key	4	8	14

Tez çalışması kapsamında 128 bit anahtar uzunluğuna sahip AES (AES-128) algoritması incelendiği için 128 bitlik anahtar uzunluğuna sahip AES algoritması üzerinde durulmuştur.

128 bitlik (16 baytlık) veri 4x4'lük bir matris ile ifade edilmektedir. Algoritma içerisindeki dönüşüm işlemleri bu matrise uygulanmakta ve Şekil 3.1'de görülen bu matrise AES durum matrisi (AES State Array) denilmektedir.



Şekil 3.1 AES Durum Matrisi



Şekil 3.2 Tüm AES Algoritması[33]

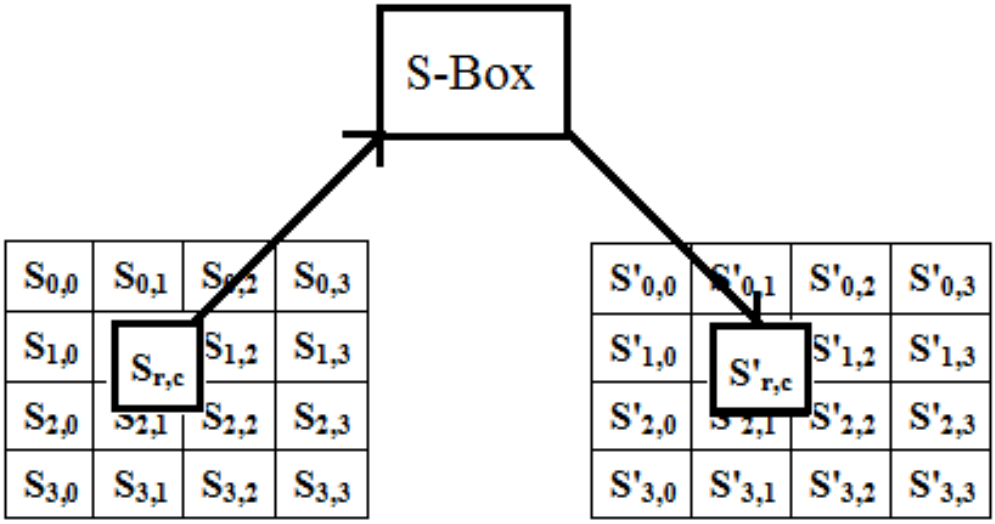
AES algoritmasının genel yapısı Şekil 3.2'de görülmektedir. İlk aşamada Tur Anahtarı eklenmekte ve sonrasında son tur hariç olmak üzere her tur için, Bayt'ları Yer Değiştirme, Satırları Öteleme, Sütunları Karıştırma ve Tur Anahtarı Ekleme, son tur için ise Bayt'ları Yer Değiştirme, Satırları Öteleme ve Tur Anahtarı Ekleme işlemleri yapılmaktadır.

3.1.1. Bayt'ları Yer Değiştirme

Bu yer değiştirme işleminde S Kutusu (S-Box) olarak adlandırılan 16 özdeş 256 baytlık Tablo 3.3'te görülen tablo kullanılmaktadır. Şekil 3.3'te Baytları Yer Değiştirme işlemi görülmektedir.

Tablo 3.3 S-Kutusu

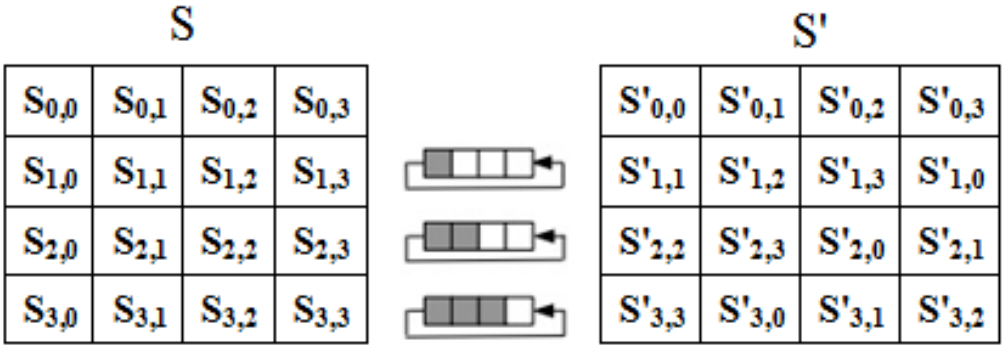
	x0	**x1**	**x2**	**x3**	**x4**	**x5**	**x6**	**x7**	**x8**	**x9**	**xa**	**xb**	**xc**	**xd**	**xe**	**xf**
0x	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1x	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2x	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3x	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb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



Şekil 3.3 Baytları Yer Değiştirme İşlemi

3.1.2. Satırları Öteleme

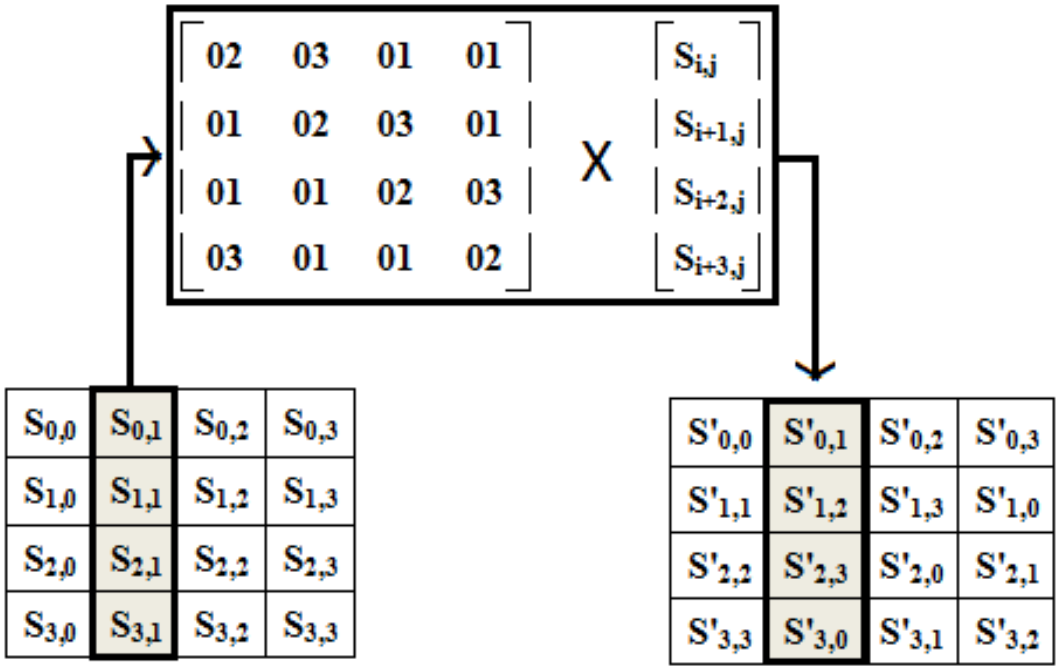
Bu aşamada ilk satırdaki baytlar olduğu gibi bırakılarak, son 3 satırdaki baytlar üzerinde işlem yapılmaktadır. Şekil 3.4'te Satırları Öteleme işlemi görülmektedir.



Şekil 3.4 Satırları Öteleme İşlemi

3.1.3. Sütunları Karıştırma

Bu işlem durum matrisinin her bir sütununa ayrı ayrı uygulanmaktadır. Şekil 3.5'te Sütunları Karıştırma işlemi görülmektedir.

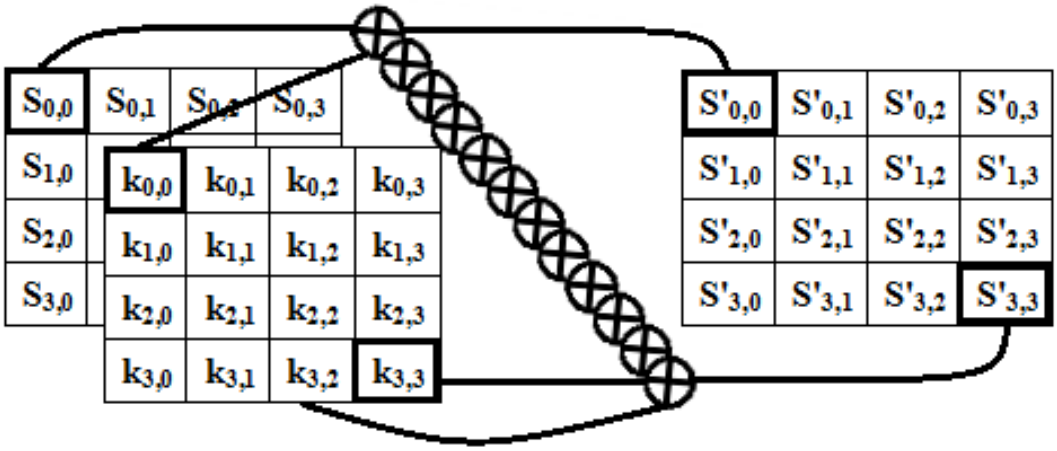


Şekil 3.5 Sütunları Karıştırma İşlemi

3.1.4. Tur Anahtarı Ekleme

Şekil 3.6'da görüldüğü üzere, her turun sonunda elde edilen matris, Anahtar Genişletme (Key Expansion) tarafından o tur için üretilen tur anahtarı ile Özel Veya (XOR) işlemine tabi tutulmaktadır. Özel Veya (XOR) işleminin doğruluk tablosu Tablo 3.4'te görülmektedir.

AES'nin anahtar uzunluğuna göre tur sayıları, Tablo 3.2'de görülebilmektedir.



Şekil 3.6 Tur Anahtarı Ekleme İşlemi

Tablo 3.4 Özel Veya (XOR) İşlemi Doğruluk Tablosu

A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

3.1.4.1. Anahtar Genişletme

128 bit sabit veri bloklarına sahip AES algoritmasının Anahtar Genişletme (Key Expansion) fonksiyonu, 128, 192 ve 256 bit uzunluğundaki farklı anahtarlar ile çalışırken ürettiği tur anahtarlarının uzunluğu da sabit bir şekilde 128 bittir. Nr, döngü sayısıdır. Buna göre AES-128 için (Nr=10) Nr+1=11, AES-192 için (Nr=12) Nr+1=13, AES-256 için (Nr=14) Nr+1=15 alt anahtar dizisi oluşturulmaktadır.

Anahtar genişletme işlemi döngü anahtarlarını word word (32 bit 32 bit) oluşturmaktadır. Bu işlemi aşağıdaki gibi tanımlanan 4×(Nr +1) word oluşturmaktadır.

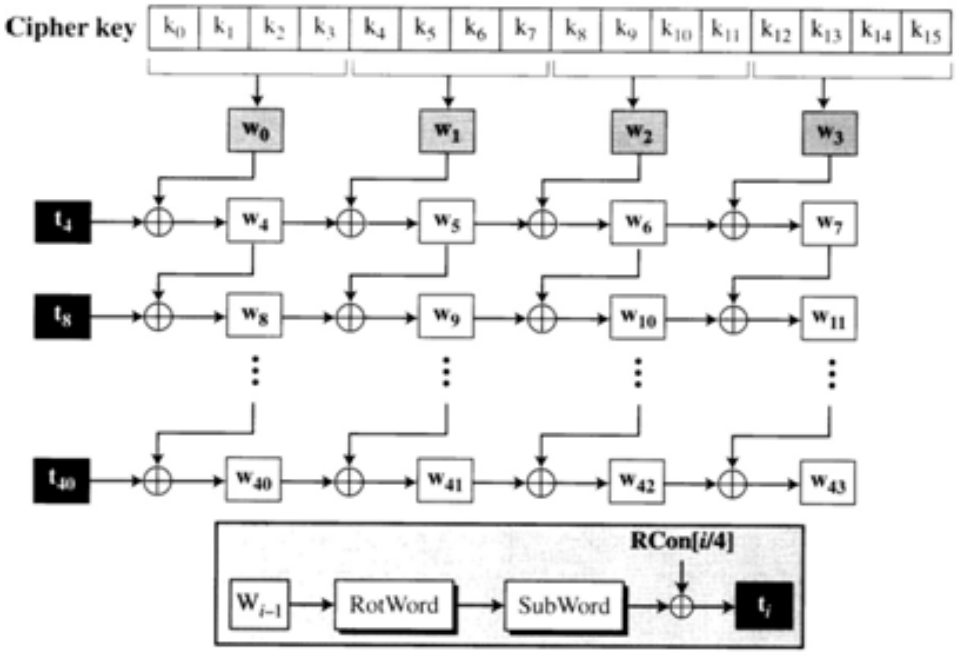
- $W_0, W_1, W_2, \ldots, W_{4(Nr+1)-1}$

Farklı bir söylemle, AES-128'de 44 word, AES-192'de 52 word ve AES-256'da 60 word vardır. Tablo 3.5'te döngüler ve word'ler arasındaki ilişki gösterilmektedir:[34]

Tablo 3.5 Döngüler ve word'ler Arasındaki İlişki[34]

Round	*Words*			
Pre-round	$\mathbf{w}_0$	$\mathbf{w}_1$	$\mathbf{w}_2$	$\mathbf{w}_3$
1	$\mathbf{w}_4$	$\mathbf{w}_5$	$\mathbf{w}_6$	$\mathbf{w}_7$
2	$\mathbf{w}_8$	$\mathbf{w}_9$	$\mathbf{w}_{10}$	$\mathbf{w}_{11}$
. . .	. . .			
N_r	$\mathbf{w}_{4N_r}$	$\mathbf{w}_{4N_r+1}$	$\mathbf{w}_{4N_r+2}$	$\mathbf{w}_{4N_r+3}$

Şekil 3.7'de AES-128 algoritması için orijinal anahtardan 44 word'ün nasıl oluşturulduğu gösterilmektedir.



Şekil 3.7 AES-128 için 44 word'ün oluşturulması[34]

Anahtar genişletme işlemi aşağıdaki gibidir:[34]

İlk 4 word (w_0, w_1, w_2, w_3) şifre anahtarından elde edilmktedir. Şifre anahtarı k_0 dan k_{15}'e kadar 16 baytlık bir dizi olarak düşünülmektedir. İlk 4 byte (k_0 dan k_3'e) w_0, ikinci 4 byte (k_4'ten k_7'ye) w_1 ve bu şekilde diğer word'ler w_2 ve w_3'te şifre anahtarının wordler şeklinde yan yana konması ile elde edilmektedir.

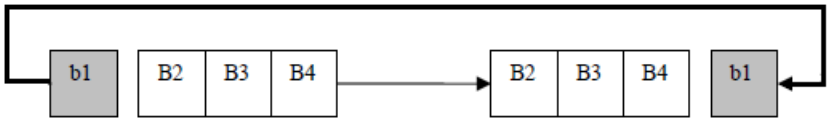
Diğer wordler (w_i i=4 den 43'e kadar) aşağıdaki şekilde elde edilir:

Eğer i (mod 4) $\neq$ 0 ise $w_i = w_{i-1} \oplus w_{i-4}$ şeklinde Şekil 3.7'de de görüldüğü gibi soldan ve üstten bir değerden elde edilmektedir.

Eğer i (mod 4) = 0 ise $w_i = t \oplus w_{i-4}$ şeklinde elde edilmektedir. Burada t geçici bir bellek ve iki rutinin w_{i-1} üzerindeki uygulama sonucudur. Bunlar, SubWord ve RotWord'tür. t'nin elde edilme süreci bir döngü sabiti RCon ile XOR'lama işlemi ile sonlanmaktadır. Diğer bir deyişle; $t = SubWord(RotWord(w_{i-1})) \oplus RCon_{i/4}$.'dir.

3.1.4.1.1. Kelime Öteleme (Rotword)

Kelime öteleme (Rotword) işlemi, Satırları Öteleme işlemine benzemektedir. Fakat sadece bir satıra uygulanmaktadır. Şekil 3.8'de görülen bu işlem bir word'ü dört baytın bir dizisi olarak almakta ve her bayt'ı sola ötelemektedir.



Şekil 3.8 Kelime Öteleme İşlemi (Rotword)

3.1.4.1.2. Kelime Yer Değiştirme (Subword)

Baytları Yer Değiştirme (Sub Bytes) işlemine benzemektedir. Ancak sadece 4 bayta uygulanmaktadır. Bu döngü, word'deki her bayt değerini almakta ve S-Kutusu yardımıyla diğer bir bayt ile yer değiştirmektedir[34].

3.1.4.1.3. Döngü Sabitleri (RCon - Round Constants)

Her döngü sabiti yani RCon, 4 bayt değerinde ve en sağdaki 3 bayt'ı 0 olan bir değerdir. Tablo 3.6'da AES-128 için (on döngü) değerleri görülmektedir[34]

Tablo 3.6 AES-128 için Döngü Sabitleri

Round	*Constant (RCon)*	*Round*	*Constant (RCon)*
1	$(\underline{\mathbf{01}}\ 00\ 00\ 00)_{16}$	6	$(\underline{\mathbf{20}}\ 00\ 00\ 00)_{16}$
2	$(\underline{\mathbf{02}}\ 00\ 00\ 00)_{16}$	7	$(\underline{\mathbf{40}}\ 00\ 00\ 00)_{16}$
3	$(\underline{\mathbf{04}}\ 00\ 00\ 00)_{16}$	8	$(\underline{\mathbf{80}}\ 00\ 00\ 00)_{16}$
4	$(\underline{\mathbf{08}}\ 00\ 00\ 00)_{16}$	9	$(\underline{\mathbf{1B}}\ 00\ 00\ 00)_{16}$
5	$(\underline{\mathbf{10}}\ 00\ 00\ 00)_{16}$	10	$(\underline{\mathbf{36}}\ 00\ 00\ 00)_{16}$

Anahtar genişletme işlemi word değerlerini hesaplarken, yani Tablo 3.6'yı elde ederken ya da en soldaki baytı dinamik olarak hesaplamak için GF(2^8) cismi kullanılmaktadır. Tablo 3.7'de görülmektedir[34].

Tablo 3.7 AES-128 için Döngü Sabitlerinin Hesaplanması

RC_1	$\rightarrow x^{1-1}$	$=x^0$	mod *prime*	$=1$	$\rightarrow 00000001$	$\rightarrow 01_{16}$
RC_2	$\rightarrow x^{2-1}$	$=x^1$	mod *prime*	$=x$	$\rightarrow 00000010$	$\rightarrow 02_{16}$
RC_3	$\rightarrow x^{3-1}$	$=x^2$	mod *prime*	$=x^2$	$\rightarrow 00000100$	$\rightarrow 04_{16}$
RC_4	$\rightarrow x^{4-1}$	$=x^3$	mod *prime*	$=x^3$	$\rightarrow 00001000$	$\rightarrow 08_{16}$
RC_5	$\rightarrow x^{5-1}$	$=x^4$	mod *prime*	$=x^4$	$\rightarrow 00010000$	$\rightarrow 10_{16}$
RC_6	$\rightarrow x^{6-1}$	$=x^5$	mod *prime*	$=x^5$	$\rightarrow 00100000$	$\rightarrow 20_{16}$
RC_7	$\rightarrow x^{7-1}$	$=x^6$	mod *prime*	$=x^6$	$\rightarrow 01000000$	$\rightarrow 40_{16}$
RC_8	$\rightarrow x^{8-1}$	$=x^7$	mod *prime*	$=x^7$	$\rightarrow 10000000$	$\rightarrow 80_{16}$
RC_9	$\rightarrow x^{9-1}$	$=x^8$	mod *prime*	$=x^4+x^3+x+1$	$\rightarrow 00011011$	$\rightarrow 1B_{16}$
RC_{10}	$\rightarrow x^{10-1}$	$=x^9$	mod *prime*	$=x^5+x^4+x^2+x$	$\rightarrow 00110110$	$\rightarrow 36_{16}$

En soldaki bayt, RC_i, x^{i-1}'dir ve i değeri döngü sayısıdır. AES, $x^8+x^4+x^3+x+1$ indirgenemez polinomunu kullanmaktadır.

Örnek: Alice ve Bob 128 bit şifre anahtarı için on altılık sayı sisteminde (hexadecimal) 24 75 A2 B3 34 75 56 88 31 E2 12 00 13 AA 54 87 değeri üzerinde anlaşmış olduğu düşünülürse, diğer döngüler için anahtarların elde edilmesi Tablo 3.8 'deki gibi gösterilebilmektedir[34].

Tablo 3.8 AES-128 için Anahtarların Elde Edilmesi

Round	*Values of t's*	*First word in the round*	*Second word in the round*	*Third word in the round*	*Fourth word in the round*
—		w_{00} = 2475A2B3	w_{01} = 34755688	w_{02} = 31E21200	w_{03} = 13AA5487
1	AD20177D	w_{04} = 8955B5CE	w_{05} = BD20E346	w_{06} = 8CC2F146	w_{07} = 9F68A5C1
2	470678DB	w_{08} = CE53CD15	w_{09} = 73732E53	w_{10} = FFB1DF15	w_{11} = 60D97AD4
3	31DA48D0	w_{12} = FF8985C5	w_{13} = 8CFAAB96	w_{14} = 734B7483	w_{15} = 2475A2B3
4	47AB5B7D	w_{16} = B822deb8	w_{17} = 34D8752E	w_{18} = 479301AD	w_{19} = 54010FFA
5	6C762D20	w_{20} = D454F398	w_{21} = E08C86B6	w_{22} = A71F871B	w_{23} = F31E88E1
6	52C4F80D	w_{24} = 86900B95	w_{25} = 661C8D23	w_{26} = C1030A38	w_{27} = 321D82D9
7	E4133523	w_{28} = 62833EB6	w_{29} = 049FB395	w_{30} = C59CB9AD	w_{31} = F7813B74
8	8CE29268	w_{32} = EE61ACDE	w_{33} = EAFE1F4B	w_{34} = 2F62A6E6	w_{35} = D8E39D92
9	0A5E4F61	w_{36} = E43FE3BF	w_{37} = 0EC1FCF4	w_{38} = 21A35A12	w_{39} = F940C780
10	3FC6CD99	w_{40} = DBF92E26	w_{41} = D538D2D2	w_{42} = F49B88C0	w_{43} = 0DDB4F40

Her döngüde son üç word'ün hesabı oldukça basittir. İlk word'ün hesabı için ilk önce geçici word (t) nin hesaplanması gerekmektedir. Örneğin ilk t (döngü 1 için) aşağıdaki gibi hesaplanabilmektedir[34].

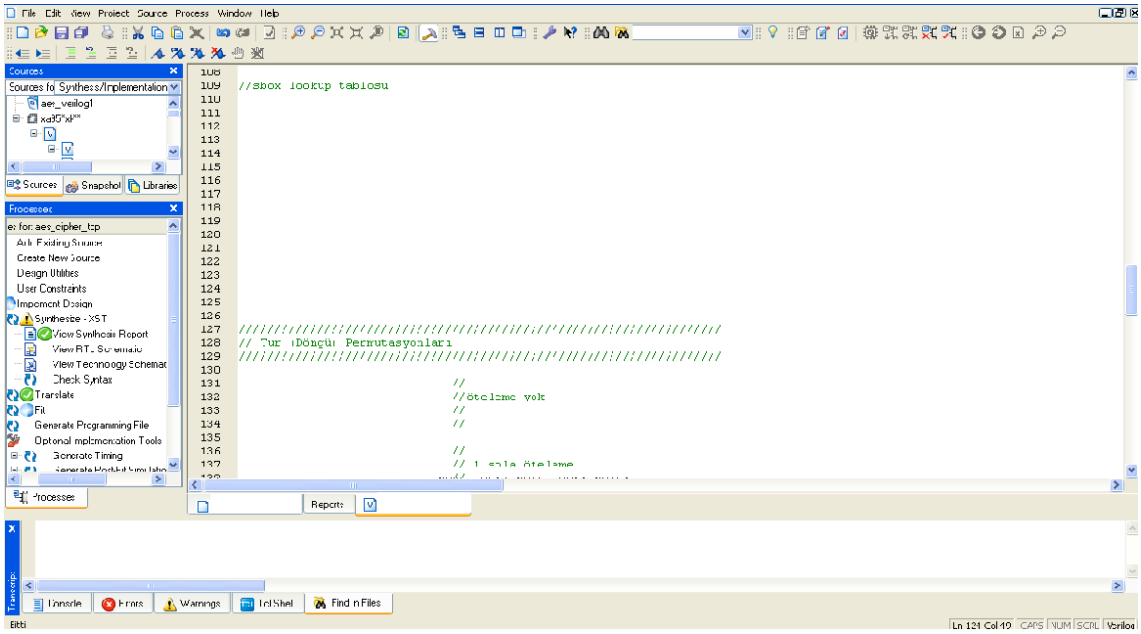
RotWord (13AA5487) = AA548713
SubWord (AA548713) = AC20177D
t = AC20177D $\oplus$ $RCon_1$ = AC20177D $\oplus$ 01000000 = AD20177D

AES şifresinde her döngü anahtarı bir önceki döngü anahtarına bağlıdır. Bununla beraber bu bağımlılık Kelime Yer Değiştirme (Sub Word) dönüşümünden dolayı doğrusal değildir. Döngü sabitlerinin toplanması da her döngü anahtarının bir öncekinden farklı olmasını garanti etmektedir[34].

4. AES'NİN FPGA'DA UYGULANMASI VE ŞİFRELEME SÜRESİ ÖLÇÜMÜ

FPGA'lar önemli donanımsal ortamlarda kriptografik algoritma gerçeklemelerinin ayrılmaz bir parçası olarak kabul edilmektedirler. FPGA'lar kriptografik algoritmaların uygulanması için kolay ve oldukça ucuz bir çözüm sunmaktadırlar.

Şekil 4.1'de Xilinx ISE Design Suite programının ekran görüntüsü yer almaktadır.



Şekil 4.1 Xilinx ISE Desing Suite Programının Ekran Görüntüsü

Xilinx ISE Design Suite 13.3 programında Verilog HDL dili kullanılarak tasarlanan 128 bitlik Gelişmiş Şifreleme Standardı (AES) algoritmasının doğruluğunun teyit edilmesi için Federal Information Processing Standards Publication 197 (FIPS-197) ([29] numaralı kaynak) kaynağında yer alan Giriş Bloğu (Text in) ve Şifreleme

Anahtarı (Cipher Key) değerlerinin aynısı tasarıma girilmiştir. Girilen değerler ve elde edilmesi beklenen şifrelenmiş metin (Text out) Tablo 4.1'de görülmektedir.

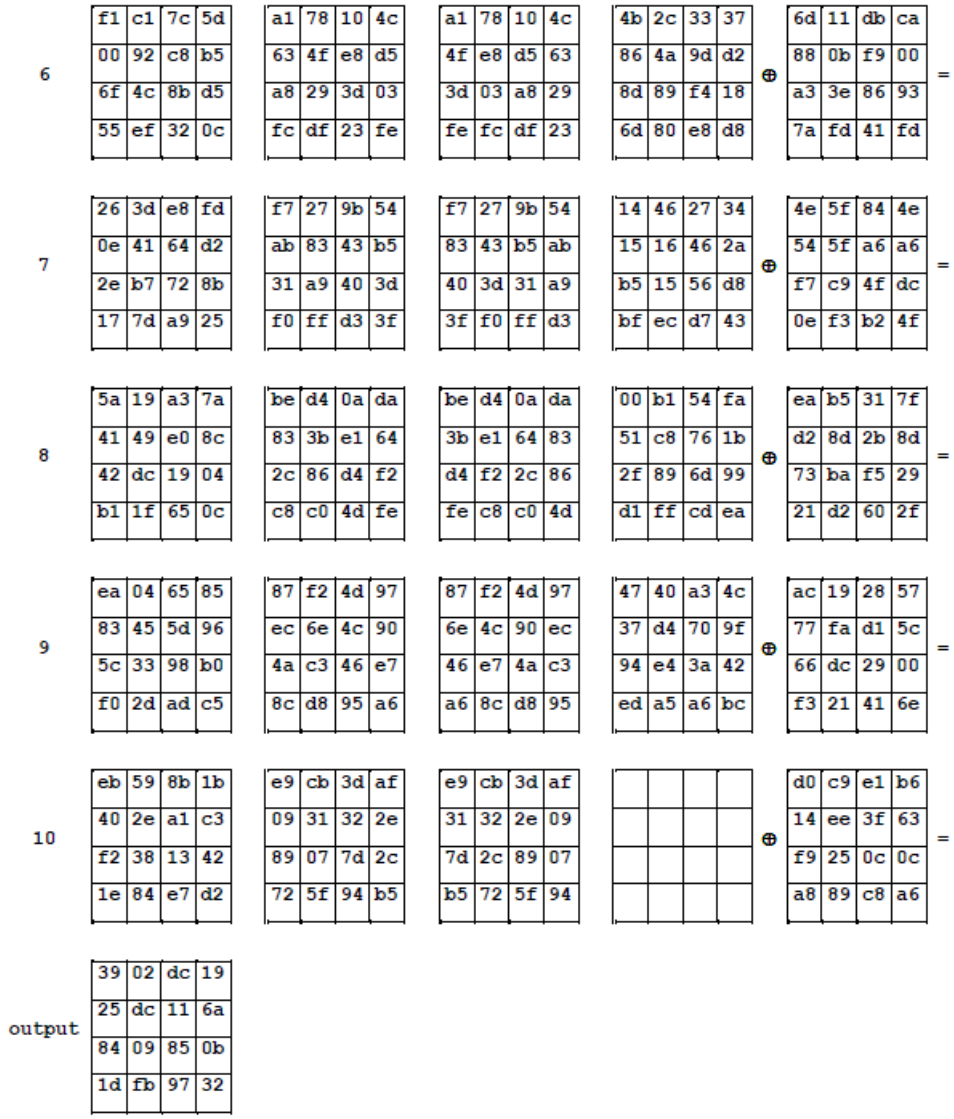
Tablo 4.1 Giriş Bloğu, Şifreleme Anahtarı ve Şifrelenmiş Metin Değerleri

	Değerler (Values)
Giriş Bloğu (Text in)	3243f6a8885a308d313198a2e0370734
Şifreleme Anahtarı (Cipher Key)	2b7e151628aed2a6abf7158809cf4f3c
Şifrelenmiş Metin (Text out)	3925841d02dc09fbdc118597196a0b32

Girilen bu değerlere göre Gelişmiş Şifreleme Standardının (AES) Tablo 4.2'deki işlemleri yaparak şifrelenmiş metni oluşturması beklenmektedir.

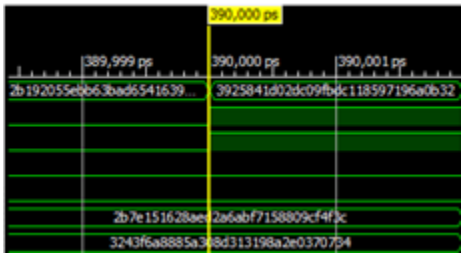
Tablo 4.2 128 bitlik Gelişmiş Şifreleme Standardı (AES-128) İşlem Sonuçları[29]

Round Number	Start of Round	After SubBytes	After ShiftRows	After MixColumns	Round Key Value	
input	32 88 31 e0 43 5a 31 37 f6 30 98 07 a8 8d a2 34				2b 28 ab 09 7e ae f7 cf 15 d2 15 4f 16 a6 88 3c	⊕ =
1	19 a0 9a e9 3d f4 c6 f8 e3 e2 8d 48 be 2b 2a 08	d4 e0 b8 1e 27 bf b4 41 11 98 5d 52 ae f1 e5 30	d4 e0 b8 1e bf b4 41 27 5d 52 11 98 30 ae f1 e5	04 e0 48 28 66 cb f8 06 81 19 d3 26 e5 9a 7a 4c	a0 88 23 2a fa 54 a3 6c fe 2c 39 76 17 b1 39 05	⊕ =
2	a4 68 6b 02 9c 9f 5b 6a 7f 35 ea 50 f2 2b 43 49	49 45 7f 77 de db 39 02 d2 96 87 53 89 f1 1a 3b	49 45 7f 77 db 39 02 de 87 53 d2 96 3b 89 f1 1a	58 1b db 1b 4d 4b e7 6b ca 5a ca b0 f1 ac a8 e5	f2 7a 59 73 c2 96 35 59 95 b9 80 f6 f2 43 7a 7f	⊕ =
3	aa 61 82 68 8f dd d2 32 5f e3 4a 46 03 ef d2 9a	ac ef 13 45 73 c1 b5 23 cf 11 d6 5a 7b df b5 b8	ac ef 13 45 c1 b5 23 73 d6 5a cf 11 b8 7b df b5	75 20 53 bb ec 0b c0 25 09 63 cf d0 93 33 7c dc	3d 47 1e 6d 80 16 23 7a 47 fe 7e 88 7d 3e 44 3b	⊕ =
4	48 67 4d d6 6c 1d e3 5f 4e 9d b1 58 ee 0d 38 e7	52 85 e3 f6 50 a4 11 cf 2f 5e c8 6a 28 d7 07 94	52 85 e3 f6 a4 11 cf 50 c8 6a 2f 5e 94 28 d7 07	0f 60 6f 5e d6 31 c0 b3 da 38 10 13 a9 bf 6b 01	ef a8 b6 db 44 52 71 0b a5 5b 25 ad 41 7f 3b 00	⊕ =
5	e0 c8 d9 85 92 63 b1 b8 7f 63 35 be e8 c0 50 01	e1 e8 35 97 4f fb c8 6c d2 fb 96 ae 9b ba 53 7c	e1 e8 35 97 fb c8 6c 4f 96 ae d2 fb 7c 9b ba 53	25 bd b6 4c d1 11 3a 4c a9 d1 33 c0 ad 68 8e b0	d4 7c ca 11 d1 83 f2 f9 c6 9d b8 15 f8 87 bc bc	⊕ =



Tasarım Xilinx ISE Design Suite 13.3 programında sentezlenerek, benzetimi oluşturulmuştur. Tablo 4.1'de belirtilen Giriş Bloğu (Text in) ve Şifreleme Anahtarı (Cipher Key) değerleri girilerek, aynı tabloda belirtilen Şifrelenmiş Metin (Text out) değerinin aynısının oluşturulduğu gözlenmiştir.

Şekil 4.2'de benzetim sonuçlarına ilişkin alınan ekran görüntüsünde, birinci sırada Şifrelenmiş Metin (Text out), ikinci sırada Şifreleme Anahtarı (Cipher Key), üçüncü sırada ise Giriş Bloğu (Text in) değerleri ve Gelişmiş Şifreleme Standardının şifreleme zamanı 390.000ps (390ns) olarak görülmektedir.



Şekil 4.2 AES-128 Şifreleme İşlemi Benzetim Sonucu

Ayrıca tez kapsamında AES-128'in bilgisayar üzerindeki şifreleme süresi de tespit edilmeye çalışılmıştır.

5. AES'NİN BİLGİSAYAR ÜZERİNDE ŞİFRELEME SÜRESİ ÖLÇÜMÜ

Bilgisayar üzerinde, bir çok kriptoloji algoritması, çeşitli yazılım dillerinde kodlanmıştır[32]. Bu çalışmada AES-128 algoritması tasarımı için kodlama kolaylığı, görsel programlama dilleri ile karşılaştırıldığında daha az yer kaplaması ve daha sağlıklı şifreleme süresi ölçümü yapılacağı düşüncesi ile C yazılım dili tercih edilmiştir.

Şifreleme süresi ölçümü yaparken, Windows türevi işletim sistemlerine göre sistem kaynaklarını daha az kullanan ve şifreleme süresinin daha sağlıklı ölçülmesini sağlayan Linux türevi olan Ubuntu 12.04 LTS işletim sistemi tercih edilmiştir.

Şifreleme süresi ölçümünün yapıldığı bilgisayarın özellikleri Tablo 5.1'de görülmektedir.

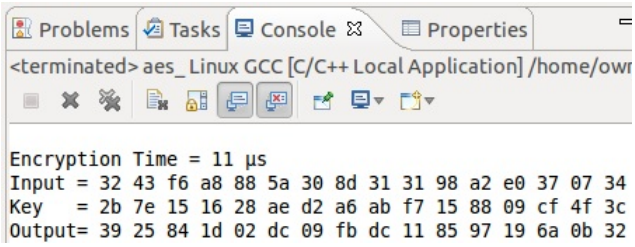
Tablo 5.1 AES Şifreleme Süresinin Ölçüldüğü Bilgisayarın Özellikleri

	Özellikler (Qualifications)
İşlemci (Processor)	Intel Core i7-2670 (6MB Cache, up to 3.1 GHz)
İşletim Sistemi (Operating System)	Ubuntu 12.04 LTS
Bellek (Memory)	6GB DDR3 SDRAM at 1333Mhz
Sabit Disk (Hard Drive)	5400 Rpm SATA (650GB)

Ubuntu 12.04 LTS işletim sistemi üzerinde C kodlarının yazılması için Eclipse editörü tercih edilmiştir. Yazılan kodları derlemek için ise GCC (GNU Compiler Collection) derleyicisi tercih edilerek, Eclipse ile uyumlu çalışması sağlanmıştır.

Bu çalışmada da bölüm dörtte yer alan Tablo 4.1'deki Giriş Bloğu (Text in) ve Şifreleme Anahtarı (Cipher Key) değerleri girilerek, aynı tabloda belirtilen Şifrelenmiş Metin (Text out) değerinin aynısının oluşturulduğu gözlenmiştir. Bu değerler ve

bilgisayar üzerindeki şifreleme süresi ölçümüne ilişkin alınan ekran görüntüsü Şekil 5.1'de görülmektedir.



Şekil 5.1 AES-128'in Bilgisayar Üzerindeki Şifreleme Süresi

6. SONUÇLAR VE TARTIŞMA

Gelişmiş Şifreleme Standardının (AES) FPGA üzerinde uygulanması ve FPGA'daki şifreleme süresinin ölçülmesi amacıyla yapılan tez çalışması kapsamında Programlanabilir Mantıksal Aygıtların gelişim süreci PROM'dan başlanarak, sırasıyla PLA, PAL, GAL, SPLD, CPLD, ASIC olacak şekilde detaylıca incelenmiştir. Son olarak da FPGA'ya değinilmiştir.

FPGA'lar incelenirken, yapısı, programlama teknolojileri, güvenlik açısından durumları, tasarım akışı, Verilog DTD ile tasarımlarının yapılması konularına detaylıca değinilmiştir.

Gelişmiş Şifreleme Standardının (AES) ortaya çıkışından başlanarak, genel yapısı ve tüm işlem adımları bu algoritma üzerinde çalışmayı düşünenlere yol gösterebilmesi amacı ile çok detaylı bir şekilde ele alınarak, tüm işlem basamakları örneklendirilerek açıklanmaya çalışılmıştır.

Gelişmiş Şifreleme Standardının (AES-128), paralel işlem yapabilme yeteneğine sahip olan FPGA üzerindeki şifreleme süresinin tespit edilmesi amacı ile Xilinx ISE Design Suite 13.3 programında tasarım sentezlenerek, benzetimi oluşturulmuş ve 390ns'de şifreleme işlemini yaptığı tespit edilmiştir.

Bu çalışmaya ek olarak, AES-128'in bilgisayar üzerindeki şifreleme süresinin ölçülmesi ve FPGA'nın şifreleme süresi ile karşılaştırılması amacı ile Ubuntu İşletim Sistemi üzerinde, C dili ile kodlanarak, GCC derleyicisi kullanılarak derlenen, AES-128'in, 11µs'de şifreleme işlemini yaptığı tespit edilmiştir.

FPGA ve bilgisayarın şifreleme süreleri karşılaştırıldığında, şifreleme süresi ölçümünde kullanılan bilgisayardan, FPGA'nın 28,21 kat (11000/390=28,21) daha hızlı şifreleme işlemi yaptığı sonucuna ulaşılmıştır. Gelişen teknoloji ile birlikte bilgisayar teknolojisi ile FPGA arasındaki bu farkın kapatılamayacağının garantisi yoktur. Fakat günümüz için bu fark göz ardı edilemeyecek derecede büyüktür.

Ayrıca FPGA ile bilgisayar maliyet açısından karşılaştırıldığında, Spartan 3E FPGA'nın, şifreleme süresi ölçülen bilgisayardan daha ucuza mal edildiği ortaya çıkmaktadır. Bu nedenle maliyet ve hızlı işlem yapabilme özellikleri ile de FPGA ön plana çıkmaktadır.

ÖNERİLER

Verilog DTD, C diline benzerliği nedeniyle, C programlama dili bilgisine sahip tasarımcılar tarafından çok daha kolay anlaşılabilmektedir. Fakat VHDL'de de rahatlıkla istenilen tasarımlar yapılabilmektedir.

AES veri şifreleme algoritmasının bitsel düzeydeki işlemleri nasıl yaptığını anlamakta zorlanan araştırmacılara yardımcı olması amacıyla, tüm işlemlerin detaylıca anlatıldığı 35 numaralı kaynağı incelemelerini önermekteyim.

KAYNAKLAR

[1] http://www.mouloudrahmani.com/ElectricalEngineering/Embedded/MemoryARM (Ziyaret tarihi: Kasım 2014)

[2] http://upload.wikimedia.org/wikipedia/commons/0/0e/Programmable_logic_array_(schematic_drawing_example).svg (Ziyaret tarihi: Kasım 2014)

[3] http://folk.uio.no/poe/317www/fig2-14.jpg (Ziyaret tarihi: Kasım 2014)

[4] http://www.virtual.unal.edu.co/cursos/ingenieria/2000477/lecciones/040401.htm (Ziyaret tarihi: Kasım 2014)

[5] http://en.wikipedia.org/wiki/Field-programmable_gate_array (Ziyaret tarihi: Kasım 2014)

[6] ERYILMAZ E. ***Kablosuz Algılayıcı Ağlar İçin Skipjack Şifreleme Algoritmasının Fpga ile Gerçeklenmesi***, Yüksek Lisans Tezi, Kocaeli Üniversitesi, Fen Bilimleri Enstitüsü, Kocaeli, Türkiye, 2009

[7] AKTAŞ M., ***FPGA Mimarisi,*** İstanbul Teknik Üniversitesi, Bilişim Enstitüsü, Bilgisayar Mimarisinde Yeni Yaklaşımlar Dersi Raporu

[8] ÖZTEMÜR M., ***FPGA Güvenliği***, TÜBİTAK BİLGEM, 2011 https://www.bilgiguvenligi.gov.tr/donanim-guvenligi/fpga-guvenligi.html (Ziyaret tarihi: Kasım 2014)

[9] JOSEPH P., ***Designing For Reuse Of Configurable Logic***, Teknik Rapor, Carnegie Mellon Üniversitesi, Yazılım Mühendisliği Enstitüsü, Pitsburg, ABD, 2005

[10] YILMAZ A.R., ***FPGA Üzerinde Diferansiyel Gelişim Algoritması İle Yapay Sinir Ağı Eğitimi***, Yüksek Lisans Tezi, Yıldız Teknik Üniversitesi, Fen Bilimleri Enstitüsü, İstanbul, Türkiye, 2014

[11] ***Xilinx All Programmable***, 7 Series FPGA Overview

[12] TEMUR G., ***Yapay Sinir Ağlarının Otomatik Olarak FPGA Çipine Uygulanması İçin Denetleyici Tasarım Aracı,*** Yüksek Lisans Tezi, Düzce Üniversitesi, Fen Bilimleri Enstitüsü, Düzce, Türkiye, 2013

[13] BAUMANN C., ***Field Programmable Gate Array (FPGA)***, Gömülü Sistem Mimarisi Seminer Çalışması, Innsbruck Üniversitesi, Avusturya, 2010

[14] KOYUNCU İ., ***A Matrix Multiplication Engine for Graphic Systems Designed to run on FPGA Devices***, Yüksek Lisans Tezi, Düzce Üniversitesi, Fen Bilimleri Enstitüsü , Düzce, Türkiye, 2008

[15] SARITEKİN K. N.,***Yapay Sinir Ağlarının Otomatik Olarak FPGA'ya Uygulanması İçin Veri Yolu Tasarım Aracı***, Yüksek Lisans Tezi, Düzce Üniversitesi, Fen Bilimleri Enstitüsü, Düzce, Türkiye, 2011

[16] ***Design Security in Nonvolatile Flash and Antifuse FPGAs***, Güvenlik Arka Planı, http://www.actel.com/documents/DesignSecurity_WP.pdf (Ziyaret tarihi: Kasım 2014)

[17] http://www.actel.com/products/solutions/security/default.aspx#flashlock (Ziyaret tarihi: Kasım 2014)

[18] http://www.design-reuse.com/articles/6529/design-security-in-nonvolatile-flash-and-antifuse-fpgas.html (Ziyaret tarihi: Kasım 2014)

[19] CASTILLO J., HUERTA P., Martinez J.I., ***Secure IP downloading for SRAM FPGAs***, http://www.sciencedirect.com/science/article/pii/S0141933106000251 (Ziyaret tarihi: Kasım 2014)

[20] AMR EL-ASHMAWI, ***Designers can Protect IP With FPGAs and Bitstream Encryption,***http://mae.pennnet.com/display_article/289152/32/ARTCL/none/none/1/Designers-can-protect-IP-with-FPGAs-and-bitstream-encryption/ (Ziyaret tarihi: Kasım 2014)

[21] http://embedded-computing.com/fpgas-built-in-aes-key-secure-system-designs (Ziyaret tarihi: Kasım 2014)

[22] SMERDON M.,***Security Solutions Using Spartan-3 Generation FPGAs,*** http://www.xilinx.com/support/documentation/white_papers/wp266.pdf (Ziyaret tarihi: Kasım 2014)

[23] http://bass.gmu.edu/courses/ECE448/viewgraphs_S08/lecture17_market.ppt (Ziyaret tarihi: Kasım 2014)

[24] http://i.cmpnet.com/eetimes/news/09/07/1565table_pg23.gif (Ziyaret tarihi: Kasım 2014)

[25] BAŞAK S***. Verilog HDL'e Giriş,*** Sunum Dosyası

[26] SCHNEIER B., ***Applied Cryptology, Second Edition: Protocols, Algorithms, and Source Code in C***, *Wiley Publishing, 1996*

[27] KOLTUKSUZ A., ***Elektronik Ticarette Güvenlik, Özgürlük Denetimi, Doğruluk-Bütünlük ve Sayısal İmza***, 4. Türkiye İnternet Konferansı, İstanbul, Türkiye, 1999

[28] STALLİNGS W., ***Cryptography and Network Security: Principles and Practise,*** ISBN: 0-13-869017-0, Prentice Hall, 1998

[29] FIPS 197: ***Advanced Encryption Standard***, Ulusal Standartlar ve Teknoloji Enstitüsü, ABD, 2001, http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf (Ziyaret tarihi: Kasım 2014)

[30] DAEMEN J. ve RIJMEN V., ***The Design of Rijndael,*** Springer Berlin-Heidelberg, 2002

[31] BAŞKÖK M. D., ***AES Şifreleme Algoritmasının Modellenmesi***, Yüksek Lisans Tezi, Gazi Üniversitesi, Fen Bilimleri Enstitüsü, Ankara, Türkiye, 2007

[32] AKMAN Y. ve YERLİKAYA T., ***Encryption Time Comparison of AES on FPGA and Computer,*** Advances in Computational Science, Engineering and Information Technology, Springer, ISBN: 978-3-319-00950-6, 317-324, 2013

[33] SAKALLI M. T., ***Modern Şifreleme Yöntemlerinin Gücünün İncelenmesi***, Doktora Tezi, Trakya Üniversitesi, Fen Bilimleri Enstitüsü, Edirne, Türkiye, 2006

[34] SAKALLI M. T., ***Şifreleme Teknikleri-1 Dersi 10. Ders Notları***, Trakya Üniversitesi, Bilgisayar Mühendisliği, Türkiye

[35] CHUNG R, PHAN W., ***Mini Advanced Encryption Standart (Mini-AES): A tested for Cryptanalysis Students Cryptologia***, XXVI(4), 2002

ÖZ GEÇMİŞ

Kocaeli Üniversitesi, Teknik Eğitim Fakültesi, Bilgisayar Öğretmenliği bölümünden 2008 yılında mezun olan Yasin AKMAN, mezuniyet sonrasında çeşitli özel okullarda Bilgisayar Öğretmenliği ve Bölüm Başkanlığı yapmıştır. Özel bir şirkette Bilgi İşlem Müdürlüğü görevini yerine getirirken 2011 yılında Selçuk Üniversitesi Huğlu Meslek Yüksekokulu'na Öğretim Görevlisi olarak atanmış. Halen Selçuk Üniversitesi'nde Öğretim Görevlisi olarak çalışmaktadır.

TEZ ÖĞRENCİSİNE AİT TEZ İLE İLGİLİ BİLİMSEL FAALİYETLER

AKMAN Y., YERLİKAYA T. ***Encryption Time Comparison of AES on FPGA and Computer,*** Advances in Computational Science, Engineering and Information Technology, Springer, ISBN: 978-3-319-00950-6, 317-324, 2013