İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

ELİPTİK EĞRİ KRİPTOSİSTEMİNİN FPGA ÜZERİNDE GERÇEKLENMESİ

YÜKSEK LİSANS TEZİ

Müh. İlker YAVUZ

Anabilim Dalı : ELEKTRONİK ve HABERLEŞME MÜHENDİSLİĞİ

Programı : ELEKTRONİK MÜHENDİSLİĞİ

OCAK 2008

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

ELİPTİK EĞRİ KRİPTOSİSTEMİNİN FPGA ÜZERİNDE GERÇEKLENMESİ

YÜKSEK LİSANS TEZİ

Müh. İlker YAVUZ
(504041211)

Tezin Enstitüye Verildiği Tarih : 24 Aralık 2007
Tezin Savunulduğu Tarih : 29 Ocak 2008

Tez Danışmanı :	**Yrd.Doç.Dr. Sıddıka Berna ÖRS YALÇIN**
Diğer Jüri Üyeleri	**Prof.Dr. Emre HARMANCI (İ.T.Ü.)**
	Doç.Dr. Mürvet KIRCI (İ.T.Ü.)

OCAK 2008

ÖNSÖZ

Bu tez çalışmasının hazırlanmasında yardımlarını esirgemeyen, sabır ve anlayışla bana yol gösteren, destek olan değerli danışman hocam Yrd. Doç. Dr. Sıddıka Berna Örs Yalçın'a teşekkürü bir borç bilirim.

Ayrıca yardım ve gösterdikleri anlayıştan dolayı Sn.Ümit Göğüsgeren ve Sn. Tevfik Nur başta olmak üzere TÜBİTAK-UEKAE bünyesindeki çalışma arkadaşlarıma teşekkür ederim.

Son olarak, uzakta olsa da varlığını hep yanımda hissettiğim, yaşamım boyunca benden sevgi ve desteğini esirgemeyen, zor zamanlardaki en büyük desteğim ailem, Ayten Yavuz, Mustafa Yavuz ve H.Meliz Yavuz'a, sonsuz sevgi ve teşekkürlerimi sunarım.

Aralık 2007 İlker YAVUZ

İÇİNDEKİLER

KISALTMALAR

AES	: Advanced Encryption Standard
ALP	: Ayrık Logaritma Problemi
CMOS	: Complementary Metal Oxide Semiconductor
DES	: Data Encryption Standard
DSA	: Digital Signature Algorithm
EEK	: Eliptik Eğri Kriptografisi
ECC	: Elliptic Curve Cryptography
MMÇ	: Montgomery Modüler Çarpması
FPGA	: Field Programmable Gate Array
NESSIE	: New European Schemes for Signature, Integrity and Encryption
NIST	: National Institute of Standards and Technology
RSA	: Rivest, Shamir, and Adleman

TABLO LİSTESİ

Sayfa No

ŞEKİL LİSTESİ

Sayfa No

ELİPTİK EĞRİ KRİPTOSİSTEMİNİN FPGA ÜZERİNDE GERÇEKLENMESİ

ÖZET

Eliptik eğri kriptosistemi (EEK), bir açık anahtar kriptosistemi olup 1985 yılında Rivest, Shamir, ve Adleman (RSA) kriptosistemine alternatif olarak öne sürülmüştür. EEK güvenliği yarı üstel zamanda henüz çözülemeyen ayrık logaritma problemine(ALP) dayanmaktadır. NESSIE (New European Schemes for Signature, Integrity and Encryption) raporuna göre EEK daha kısa anahtar uzunlukları ile RSA kriptosistemine eş güvenlik sağlayabilmektedir. Aynı zamanda, aynı anahtar uzunluğu için EEK, RSA kriptosisteminden daha hızlıdır. Daha kısa şifreleme anahtarı ile EEK daha az bellek ihtiyacı ve daha az güç tüketimi anlamına gelmektedir.

Bu tezde $GF(p^m)$ sonlu alanında tanımlı bir EEK sahada programlanabilir kapı dizisi (FPGA : Field Programmable Gate Array) ile gerçeklenmiştir. Bu gerçeklemede eliptik eğri performansını önemli ölçüde etkileyen modüler çarpma işlemi için Montgomery modüler çarpma (MMÇ) algoritması kullanılmıştır. Bu amaçla çarpma algoritması $GF(p^m)$ sonlu alanına uyarlanmıştır. Ayrıca eliptik eğri kriptosisteminin tamamını oluşturan tüm alt bloklar $GF(p^m)$ sonlu alanına uyarlanıp tasarlanmıştır. Bunlara ek olarak tüm devreler için girişleri uygun formlara dönüştüren dönüşüm devreleri tasarlanmıştır. Son olarak tüm alt bloklar eliptik eğri algoritmasını gerçekleyecek şekilde uygun bir sonlu durum makinası ile kontrol edilerek EEK gerçeklenmiştir.

Bu tezde ilk olarak temel kriptografik bilgiler verilmiş daha sonra EEK anlaşılabilmesi için gerekli matematiksel ifadelerden bahsedilmiştir. Dördüncü bölümde daha önceki matematiksel ifadeler kullanılarak eliptik eğri temelleri anlatılmış, EEK uygulaması için gerekli işlemler, ve algoritmalar verilmiştir. Beşinci bölümde eliptik eğri uygulamalarından bahsedilmiştir. Altıncı bölümde EEK gerçeklenmesi detaylı şekilde anlatılmıştır. Son olarak, sonuçlardan bahsedilmiş ve literatürdeki çalışmalarla karşılaştırılmıştır.

FPGA IMPLEMENTATION OF AN ELLIPTIC CURVE CRYPTOSYSTEM

SUMMARY

Elliptic Curve Cryptosytem(ECC) is a public key cryptosytem and it is proposed instead of Rivest, Shamir, and Adleman (RSA) in 1985. Security of ECC is based on discrete logarithm problem which has not solved yet on sub-exponential domain. According to NESSIE (New European Schemes for Signature, Integrity and Encryption) report, ECC using shortest encryption key can provide equivalent security level with RSA. Also, ECC implementation is faster than RSA cryptosystem for equal key lenghts. With shortest encryption key, ECC means less memory need and less power consumption.

In this thesis, an elliptic curve cryptosystem defined over GF(p^m) finite field is implemented on an FPGA (Field Programmable Gate Array). In this implementation, Montgomery modular multiplication algorithm is used for modular multiplication operation which has considerable effect on performance of an elliptic curve. For this purpose, the algorithm is adapted to $GF(p^m)$ finite field. Also, other subblocks which form whole ECC are adapted to $GF(p^m)$ finite field and implemented. In addition, transformation circuits are implemented that converts normal inputs to appropriate input forms for all circuits. Finally, all subblocks are controlled to implement elliptic curve algorithm using an appropriate state machine and elliptic curve cryptosystem is realized.

In the first part of this thesis, cryptographic fundamentals are given and then mathematical backgrounds for ECC is given. In the fourth chapter, elliptic curve basics are given using previous mathematical background information and then necessary ECC operations and algorithms are presented for ECC. In the fifth chapter, ECC protocols are given. In the sixth chapter, ECC implementation is given with all details. Finally, results are given and compared with studies in the literature.

1. GİRİŞ

Eliptik eğri kriptosistemi (EEK) [1][2] bir açık anahtar kriptosistemi olup RSA [3] açık anahtar kriptosistemine alternatif olarak 1985 yılında öne sürülmüştür.[3] RSA kriptosisteminin güvenliği, büyük tam sayıların asal çarpanlarına ayrılmasındaki matematiksel zorluğa dayanırken EEK güvenilirliği RSA' e göre daha yeni bir matematiksel problem olan ve günümüzde yarı üstel zamanda hala çözülemeyen ayrık logaritma problemine (ALP) dayanmaktadır. NESSIE (New European Schemes for Signature, Integrity and Encryption) tarafından yayımlanan rapora [4] göre EEK daha kısa anahtar ile RSA kriptosistemi ile aynı güvenlik seviyesi sağlayabilmektedir. Örneğin 160-bitlik bir EEK ile 1536-bitlik bir RSA kriptosisteminin birbirine güvenlik seviyesi anlamında denktir.[4] Ayrıca aynı anahtar uzunluğu için EEK, RSA kriptosisteminden daha hızlı şifreleme yapabilmektedir.[4] Bu özellikleri ile EEK, RSA kriptosisteminin yerini almak için adaydır çünkü ihtiyaç duyulan güvenlik seviyesi EEK daha küçük anahtarlar ile karşılayabilecektir. Küçük anahtar uzunluğu ile gerçeklenen EEK, daha az işlem maliyeti ve daha az bellek tüketimi demektir ya da başka bir bakış açısıyla aynı anahtar uzunluğu kullanıldığında EEK daha hızlı bir kriptosistemdir.[5]

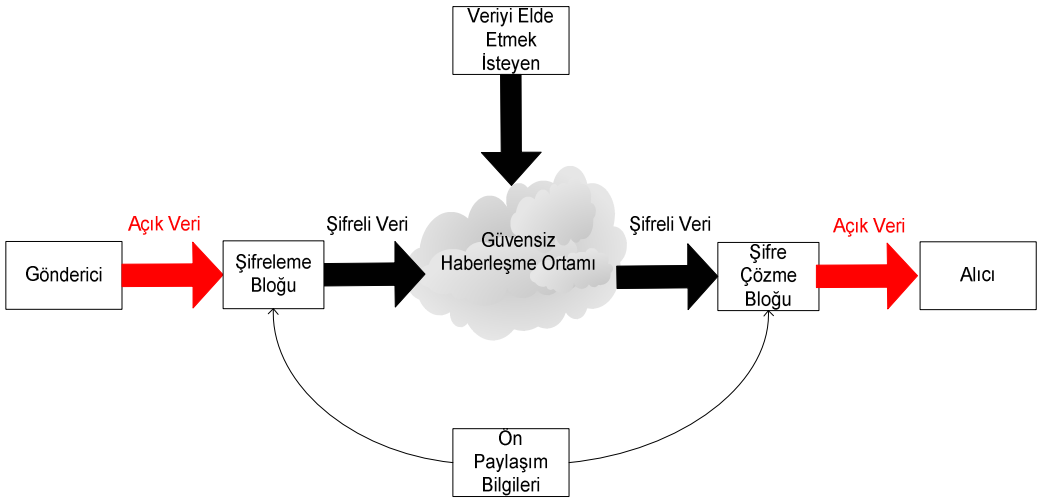
EEK gerçeklemesinde bir çok seçenek bulunmaktadır. Bunlar eliptik eğrinin üzerinde tanımlı olduğu farklı sonlu alanlar, bu sonlu alanlarda oluşturulan farklı eğriler, sonlu alan elemanlarının işlemler sırasında farklı ifade şekilleri, farklı modüler çarpma ve toplama yöntemleridir. Tüm bu seçenekler EEK için çok farklı tasarımlar yapma imkanı sağlamaktadır.

Bu çalışmada günümüze kadar diğer seçeneklere oranla çok fazla üzerinde çalışılmayan $GF(p^m)$ [6] sonlu alanında tanımlı bir EEK sahada programlanabilir kapı dizisi (FPGA : Field Programmable Gate Array) üzerinde gerçeklenecektir. Bu gerçeklemede eliptik eğri işlemlerinde en önemli alt blok olan modüler çarpma için Montgomery modüler çarpma (MMÇ) algoritması[7] kullanılarak bu algoritmanın sağladığı avantajlardan yararlanılacaktır. Bu amaçla MMÇ algoritması $GF(p^m)$ sonlu alanı için uyarlanacak ve giriş parametreleri devre içerisinde

kullanılabilir biçimlere dönüştürülecektir. Tüm bu bahsedilen çalışmalar için MMÇ devresi başta olmak üzere elemanların farklı gösterilimleri arasında geçişleri sağlayan dönüşüm sağlayan devreler tasarlanacaktır. Daha sonra bu tüm alt devreleri uygun bir akışta kullanacak bir kontrol devresi tasarlanacak ve EEK gerçeklenecektir.

2. KRİPTOGRAFİK SİSTEMLER

Kriptoloji kısaca şifre bilimidir ve amacı haberleşmek isteyen noktaların bulunduğu güvensiz ortam içerisinde, güvenli bir kanal oluşturarak iletişimin güvenliğini sağlamaktır.[8] Bu kanal sayesinde ortamda bulunan üçüncü kişiler haberleşen iki nokta arasında gidip gelen verileri elde etseler bile anlamlandıramazlar. Bu kanal telefon ya da bilgisayar ağlarından oluşan bir ortam olabilir.



Şekil 2.1 : Güvensiz Bir Ortamda Şifreli Haberleşme

Göndericinin yolladığı veriye açık veri adı verilir. Açık verinin şifreleme bloğundan geçtikten sonraki hali şifreli veri adını alır. Şifreleme bloğunda daha önceden paylaşılmış parametrelerle (anahtar ya da anahtar elde etmek için kullanılacak fonksiyonlar, değerler,...) açık veri şifrelenir ve güvensiz haberleşme ortamı üzerinden alıcıya ulaştırılır. Şifrelenmiş verinin güvensiz ortamda üçüncü kişiler tarafından ele geçirilmesi hiçbir şey ifade etmez çünkü artık şifreli veri sadece alıcının bildiği parametrelerle çözüldükten sonra anlamlı hale gelecektir.

Yukarıda anlatılan kriptosistemin sağlaması gereken görevler ve servisler şunlardır ; [9][10]

Gizlilik : Bu servis iletilen verinin sadece yetkisi olan kullanıcı tarafından erişilebilir olmasıdır. Veri diğer tüm ortam için gizli ve özeldir.

Bütünlük Sağlama : İletilen verinin sadece yetkili kullanıcılar tarafından değiştirilebilmesini bunun dışında verinin bütünlüğünün bozulmamasını sağlayan özelliktir. Değiştirme yetkisi, veriye yeni bilgiler ekleme, olan verinin bir kısmını ya da tamamını değiştirme, durumunu değiştirme, silme, yeni bir veri yaratma, geciktirme ve tekrarlama özelliklerini kapsamaktadır. Bütünlük koruması aktif saldırılar ile ilgili bir özelliktir ve dolayısıyla bütünlüğün bozulduğunu tespit etmek bütünlüğünü sağlamaktan daha önemlidir.

Asıllama : Bu özellik veri kaynağının doğruluğunu sınamakdır. Güvensiz ortamdan gönderilen verinin kaynağı, kaynağın saati, verinin içeriği, verinin gönderildiği saat gibi parametreler asıllanır. Bu özellik iki ana alt dala ayrılmaktadır bunlar, bütünlük asıllaması ve veri kaynağı asıllamasıdır. İkinci grup zaten bütünlük sağlama özelliği ile eştir.

İnkar Etmeme : Bu özellik haberleşen uçların daha önceden gönderdikleri verileri ve yaptıkları istekleri inkar edememelerini sağlar.

Yukarıda bahsedilen özellikler kriptografik algoritmalar ile sağlanmaktadır. Temel olarak iki tip kriptografik algoritma vardır. Bunlar simetrik (gizli) anahtarlı algoritmalar ve açık anahtarlı algoritmalardır.[8]

2.1. Simetrik Anahtarlı Kriptosistemler

Simetrik anahtarlı kriptosistemlerde gönderilen verinin şifrelenmesi ve çözülmesi için aynı anahtarı kullanılmaktadır ya da bir anahtardan diğer anahtar kolayca elde edilebilmektedir.[8] Simetrik anahtarlı kriptosistemlerin amacı verinin hızlı bir şekilde şifrelenmesidir. Gönderici ve alıcı haberleşme işleminden önce gizli anahtarları paylaşmalıdır. Bu kriptosistemin güvenliği algoritmanın gizliliğinden değil, anahtarın gizliliğinden gelmektedir.

Blok şifreleyiciler ve dizi şifreleyiciler olmak üzere iki tür simetrik anahtarlı kriptosistem vardır.[11] Blok şifreleyicilerde şifrelenecek veriyi sabit bloklara ayrılıp her turda bir blok şifrelenir. En temel örnekler Veri Şifreleme Standardı (DES: Data Encryption Standard) [12] ve Gelişmiş Kodlama Standardı (AES: Advanced Encryption Standard) [13] şifreleme algoritmalarıdır.

Dizi şifreleyiciler şifrelenecek verinin her bir adımda sadece bir bitini şifreler.[8] Bazı durumlarda blok şifreleyicilerde blok uzunluğu bir seçilip dizi şifreleyicilerle aynı duruma gelebilmektedir. Bu şifreleyicilerde verinin her bir biti için farklı bir şifreleme metodu ya da anahtar kullanılabilmektedir. Ayrıca iletim sırasında hata oluşma olasılığı yüksek durumlarda kullanılabilmektedirler çünkü hata gecikmesi yoktur, şifre çözme sırasında hata oluştuğu anda başka herhangi bir bit işleme sokulmaz. Bunlara ek olarak donanımsal yetersizlik nedeniyle düşük performanslı aynı anda çok fazla işlem yapamayan sistemlerde her bir turda bir sembol işlenmesi gerekiyorsa dizi şifreleme kullanılmalıdırlar.

Simetrik anahtarlı şifrelemede en temel sorun hangi simetrik anahtarın kullanılacağı konusunda tarafların anlaşma yöntemidir. Tarafların birbiri ile anlaşması ve gizli anahtarı en etkili ve güvenli şekilde değişmesi gerekmektedir.[9] Bu anahtar dağıtım problemi 1977 yılında Diffie ve Hellman [14] tarafından önerilmiş olan yöntemle aşılmıştır. Diffie ve Hellman bu problemin çözümü için açık anahtar kriptosistemini önermiştir.

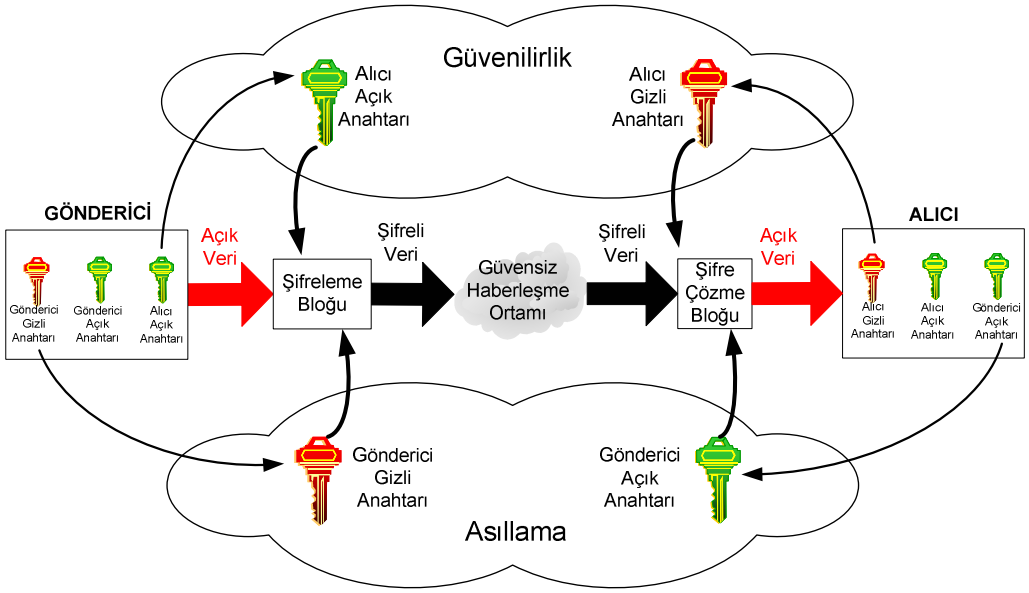
2.2. Açık Anahtarlı Kriptosistemler

Açık anahtarlı kriptosistemler şifreleme ve şifre çözme için kullanılan anahtarların birbirinden farklı olması temeline dayanmaktadırlar. Açık anahtarlı kriptosistemlerde her kullanıcının açık ve gizli olmak üzere iki anahtarı vardır.[8] Bu anahtarlardan açık anahtar haberleşme ortamındaki herkes tarafından bilinmektedir. Gizli anahtar ise kişiye özel olup sadece ait olduğu kullanıcı tarafından bilinir. Anahtarlar özel seçilmiş olup kriptosisteme göre değişiklik boyutlardadır. Örneğin RSA için büyük sayılardır ve özel matematiksel temellere dayanarak üretilmişleridir. Ayrıca teorik olarak açık anahtar biliniyor olsa bile gizli anahtarın hesaplanması zordur. Anahtarlar algoritma içinde yer alan çok özel işlevler tarafından kullanılmaktadır ve her kullanıcının açık ve gizli anahtarı bir çift oluşturur. Herkes kendi açık ve gizli anahtar çifti ile diğer tüm kullanıcıların açık anahtarına sahiptir.

Açık anahtar sistemleri iki amaçla kullanılabilir. Bunlar güvenilirlik ve asılllamadır. Anahtar paylaşımı, haberleşmek isteyen iki uç arasında güvenilirlik için kullanıldığında, veriyi gönderen şifreleme işlemini herkes tarafından bilinen alıcının açık anahtarı ile yapar. Şifrelenmiş bu veri sadece alıcının gizli anahtarı ile çözülebilmektedir. Eğer şifreli veri açılamazsa haberleşme esnasında bozulmuş

olarak kabul edilir. Şifreli veri sadece alıcının özel anahtarı ile açılabildiğinden ve bu anahtar da alıcıda güvenli bir şekilde tutulduğundan şifreli veri diğer tüm kullanıcılar için anlamsızdır ve çözülemez.

Haberleşme isteğinde bulunan uçlar asılama için açık anahtar kriptosistemini kullanacaksa bu durumda veriyi imzalayan uç kendi gizli anahtarını kullanır. Alıcı, imzayı göndericinin açık anahtarı ile doğrular. Eğer başarabilirse veri o açık anahtara ait gönderici tarafından imzalanmıştır, göndericiyi asıllamış olur.



Şekil 2.2 : Açık Anahtar Kirptografisi Kullanımı

Uygulamada açık anahtar algoritmalar 3 gruba ayrılır[8];

- *Bir TamsayınınÇarpanlarına Ayrılmasına Dayanan Algoritmalar* : Verilen bir n pozitif tamsayının asal çarpanlarını bulmaya dayanmaktadır. n sayısı çok büyük ve özel seçilmektedir. RSA [3] en çok kullanılan açık anahtar algoritması olup bu problemin çözümünün zorluğuna dayanmaktadır.
- *Ayrık Logaritma Problemine Dayanan Algoritmalar* : Verilen bir α, β ve p için öyle bir x değeri bulunmalıdır ki $\beta = \alpha^x \bmod p$ sağlamalıdır. Diffie - Hellman anahtar değişim protokolü, ElGamal algoritması bu probleme dayanmaktadır.

- *Eliptik Eğri Ayrık Logaritma Problemine Dayanan Algoritmalar* : Bu algoritmalar da ALP tabanlıdır ancak burada ALP bir eliptik eğri üzerinde tanımlıdır. Eliptik eğri Diffie-Hellman protokolü ve eliptik eğri sayısal imza algoritması bu matematiksel temele dayanmaktadır. Diğer grupta yer alan algoritmalarla 1024-2048 bit uzunluğunda sağlanan güvenlik seviyesi 160-256 bit uzunluğu ile sağlanabilmektedir. Sonlu alan aritmetiğine dayanmaktadır.

Matematiksel problemler arasında farklılıklar olmasına rağmen hepsi büyük sayılar üzerinde karmaşık işlemler yapmaktadır. Örneğin RSA'de 1024-2048 bit uzunluklu işlemler yapılırken eliptik eğri gibi ayrık logaritma problemi tabanlı algoritmalarda 160-512 bit uzunluklu sayılar üzerinde işlemler yapılmaktadır. Çalışılan anahtar uzunluklarının farklı olmasının nedeni algoritmaların matematiksel temellerindeki farklılıktan kaynaklanan karmaşıklıklarıdır.[4]

Açık anahtar kriptosistemleri şifreleme hızı konusunda gizli anahtarlı kriptosistemlerle karşılaştırıldıklarında yavaş kalmaktadırlar. Bunun nedeni açık anahtar kriptosistemlerinde çok yoğun aritmetik işlemlerin varlığıdır. Performans gerektiren şifreleme uygulamalarında (örneğin yoğun bir ağda çalışan bir ağ güvenlik cihazında) şifreleme amacıyla açık anahtar kriptosisteminin kullanılması mümkün değildir.[10] Dolayısıyla günümüz sistemleri açık anahtar ve gizli anahtar kriptosistemlerinin birleşiminden oluşmaktadır. Genel kullanım açık anahtar kriptosistemler anahtar kurulumu ve sayısal imzayla asıllama amacıyla kullanılmaktadırlar. Şifreleme işlemleri ise açık anahtar mekanizmasının oluşturduğu anahtarlarla gizli anahtar algoritmaları ile yüksek hızda şifreleme işlemlerini gerçekleştirir.[9]

2.3. Eliptik Eğri Kriptosistemi

EEK ilk olarak 1985 yılında Neal Koblitz [1] ve Victor Miller [2] tarafından önerilmiştir. EEK güvenilirliği eliptik eğri ayrık logaritma probleminin çözümünün zorluğuna dayanmaktadır. Bu problemin çözümü için henüz bir yarı üstel algoritma geliştirilememiştir.[15] Ayrıca EEK çok kullanılan RSA açık anahtar kriptosistemi ile karşılaştırıldığında aynı güvenlik seviyesi için daha kısa anahtar uzunluğu kullanır. Örneğin 1536-bit anahtarlı RSA ile oluşturulan güvenlik, 160-bit anahtarlı eliptik eğri ile sağlanabilmektedir.[4]

EEK detaylarına girmeden önce sonlu alanlar, galois alanları, alanların gösterilimi ve alanlar üzerinde işlemler gibi bu çalışma boyunca kullanılacak matematiksel kavramlardan bahsedilecektir.

3. ELİPTİK EĞRİ KRİPTOSİSTEMİ İÇİN MATEMATİKSEL İFADELER

3.1. Sonlu Alan Kavramı

Sonlu alanlar [16] ve sonlu alanların alt kümesi olan Galois alanları bir çok kriptografik algoritmanın temelini oluşturmaktadır. Bazı uygulama alanları ALP tabanlı anahtar değişim protokolleri [14], ve ElGamal [17] gibi açık anahtar kriptosistemleridir.

Sonlu alan teorisi için aşağıdaki tanımlar yapılmıştır.

Tanım 3-1 Grup

G kümesinin ve $\Diamond$ ikili işleminin oluşturduğu $< G, \Diamond >$ cebirsel yapısı aşağıdaki aksiyomları sağladığı taktirde grup adını alır.

Kapalılık : $\forall x, y \in G$ için $x \Diamond y \in G$ olmalıdır.

Birleşme : $\forall x, y, z \in G$ için $(x \Diamond y) \Diamond z = x \Diamond (y \Diamond z) \in G$ olmalıdır.

Etkisiz Eleman : $\forall x \in G$ için $x \Diamond e = e \Diamond x = x$ olan $e \in G$ vardır.

Ters Eleman : $\forall x \in G$ için $x \Diamond y = y \Diamond x = e$ olan $y \in G$ vardır.

Ayrıca ikili işlem değişme özelliği de sağlandığı taktirde $< G, \Diamond >$, değişmeli grup olarak isimlendirilir. 'Abelian Grup' olarak adlandırılır.

Değişme Özelliği : $\forall$ x, y $\in$ G için x $\Diamond$ y = y $\Diamond$ x

Örneğin tamsayılar kümesi ve toplama işlemi için $< Z, + >$, bir değişmeli gruptur. Aynı şekilde 0' dan $n-1$' e kadar olan tamsayıların oluşturduğu kümede modülo n toplama işlemi, $< Z_n, + >$ bir değişmeli gruptur.

Tanım 3-2 Halka

Halka, $< R, +, * >$, bir R kümesi ve bu kümenin elemanları üzerinde tanımlanmış olan '+' ve '*' şeklinde iki adet işlemden olşmaktadır.

R kümesinin ve üzerinde tanımlanan işlemin bir halka olması için aşağıdaki özellikleri sağlaması gerekir.

$< R,+ >$ bir değişmeli grup olmalıdır.

‘*’ işlemi R üzerinde, kapalılık ve birleşme özelliklerini sağlamalıdır. ‘*’ işlemi için bir etkisiz eleman tanımlanabilmelidir.

$\forall a,b,c \in R : (a+b)*c = (a*c)+(b*c)$ olmalıdır.

Eğer ‘*’ işleminin değişme özelliği varsa $< R,+,* >$ halkası ‘Değişmeli Halka’ dır.

Tanım 3-3 Alan

F, sayı kümesi üzerinde tanımlanmış ‘+’ ve ‘*’ işlemleriyle birlikte aşağıdaki aksiyomları sağlıyorsa bir ‘Alan’ oluşturur.

$< F,+ >$ bir değişmeli grup olmalıdır.

$< F,* >$ bir değişmeli grup olmalıdır. Ancak sadece toplama işleminin etkisisiz elemanı için bir ters elemanı bulunmayabilir.

$< F,+,* >$ bir halka olmalıdır. Yani ilk iki koşula ek olarak ‘*’ işleminin ‘+’ üzerinde dağılma özelliği olmalıdır.

Yukarıdaki duruma bir örnek olarak gerçel sayılar kümesi toplama ve çarpma işlemleri ile birlikte bir ‘Alan’ oluşturur.

Tanım 3-4 Sonlu Alan

Yukarıda tanımı verilen alanlar sonlu sayıda eleman içermesi durumunda ‘Sonlu Alan’ olarak adlandırılır.

Sonlu alanın eleman sayısı o alanın derecesidir. Aynı sonlu sayıda elemana sahip, derecesi aynı, alanlar eş yapıdadır yani aynı matematiksel yapıdadırlar sadece elemanların gösterilişleri farklıdır.

Bir sonlu alanın var olabilmesi için o sonlu alanın derecesinin, p ve m sırasıyla asal ve tamsayı olmak üzere, $q = p^m$ şeklinde bir asalın kuvveti olması gerekir ve bu alan F_q şeklinde gösterilir.

3.2. Galois Alanı Kavramı

Tanım 3-5 $GF(p)$ [18]

p asal sayı olmak koşuluyla, eleman sayısı p olan $Z_p = \{0,1,2...,p-1\}$ üzerinde modülo p toplama ve modülo p çarpma işlemlerinin tanımlanmasıyla p sayıda elemana sahip bir sonlu alan oluşur ve p karakteristikli $GF(p)$ olarak adlandırılır.

Alanın elemanlarına örnek verilirse; 3, 11, 9803, 1003039 gibi asal sayılar.

Tanım 3-6 $GF(p^m)$ Genişletilmiş Alan

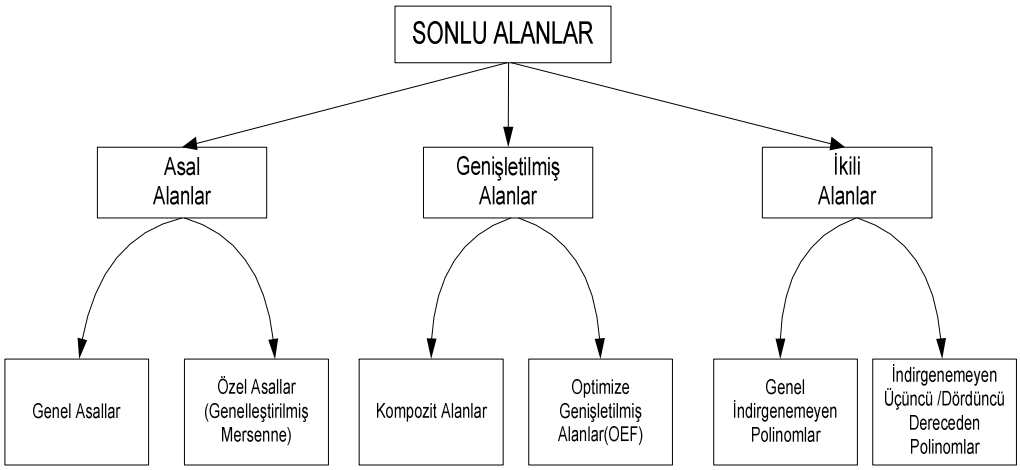
$GF(p)$ üzerinde $q = p^m$ elemanlı $GF(p^m)$ alanı oluşturulabilir. Bu yeni alana $GF(p)$ alanının genişletilmiş alanı denir. p, $GF(p^m)$ alanının karakteristiği, m ise alanın genişleme derecesi adını alır.

$GF(p^m)$ alanının p^m adet elemandan oluşur ve Galois alanı olabilmesi için m. dereceden bir indirgeme değeri kullanılır. Toplama ve çarpma işlemleri sonucunda oluşan sonucun $GF(p^m)$ sonlu alanı içerisinde kalması indirgeme değeri ile sağlanır. İndirgeme değerini aşan sonuçlar bu değerle indirgenir.

Elemanların ifade edilebilmesi için polinom şeklinde bir gösterilim kullanılabilir. Bu hem gösterilim hem de işlemlerde kolaylık sağlayacaktır. Elemanların gösterilimi ileriki bölümde detaylı olarak anlatılacaktır ancak burada bir örnek verilirse GF(3^5)'de $2x^4+x^2+2$ polinomsal gösterilimle ifade edilen bir alan elemanıdır. GF(3^5) alanı için örnek bir indirgeme polinomu da $2x^5+x^2+1$ olabilir.

Tanım 3-7 $GF(2^m)$ [19]

$GF(p^m)$ alanında p kakrakteristiğinin 2 seçilmesiyle oluşturulur. İkili sistemde çalışmak matematiksel işlemlerin donanım ve yazılımsal olarak gerçeklenmesini kolaylaştırır. $GF(2^m)$ sonlu alanları ikili sonlu alanlar olarak adlandırılır ve elemanları katsayıları $\{0,1\}$ den oluşan polinomlardır. Alan elemanlarının gösteriliminde çeşitli yöntemler kullanılarak gösterim ve hesaplama kolaylıkları sağlanmıştır. Bu gösterimlerden en çok kullanılanı polinomsal baz ve ikili baz gösterimleridir.



Şekil 3.1 : Sonlu Alanların Sınıflandırılması

Tablo 3-1 : Sonlu Alanların Özelliklerinin Karşılaştırılması

GF(p)	**GF(2^m)**	**GF(p^m)**
Alan Elemanları modülo p (0,1..p-1) 'de tamsayıdır	Elemanlar, derecesi m'den küçük ve katsayıları mod 2 de olan polinomlardır.	Elemanlar, derecesi m'den küçük ve katsayıları mod p de olan polinomlardır
İşlemler modülo p de gerçekleşir.	İşlemler derecesi k olan bir indirgeme polinomunda gerçekleşir.	İşlemler derecesi k olan bir indirgeme polinomunda gerçekleşir.
Asal sayı p, GF(p) alanı için indirgemede kullanılır..	İndirgeme polinomu, GF(2^m) de tanımlı indirgenemez polinomdur	İndirgeme polinomu, GF(p^m) de tanımlı indirgenemez polinomdur

3.3. Galois Alanında Elemanların Gösterilimi

Sonlu alanların elemanlarının gösterilmesi için farklı yolları vardır.[20] Standart polinomsal baz gösterilimi en çok kullanılan gösterilim biçimidir.

$GF(p)$ üzerinde tanımlı m. dereceden $F(x)$ polinomu, aynı alan üzerinde tanımlı başka polinomların çarpımı şeklinde gösterilemiyorsa indirgenemezdir. $F(x)$ indirgeme polinomu adını alır. İndirgenemez polinomların seçiminde kullanılan yöntemler için [21]'den yararlanılabilir. En genel halde $g_i \in F_p$ olmak üzere indirgenemez polinom eşitlik 3.1' deki gibi ifade edilir.

$$F(x) = x^m + G(x) = x^m + \sum_{i=0}^{m-1} g_i x^i \qquad \textbf{(3.1)}$$

En genel sonlu alan olan $GF(p^m)$ sonlu alanında polinomsal bazda gösterimi tanımına göre aşağıdaki şekildedir.[20]

$GF(p^m)$ için polinomsal baz $\{x^{m-1}, x^{m-2}, ..., x^2, x^1, 1\}$ kümesinden oluşmaktadır. Polinomsal baz vektörünün her bir elemanının $GF(p)$ ye ait bir elemanla çarpılmasıyla $GF(p^m)$'nin ilgili elemanının polinomsal gösterimi elde edilir.

$a \in GF(p^m)$'nin polinomsal baz gösterimi aşağıdaki gibidir;

$$a(x) = \sum a_i x^i = a_{m-1}x^{m-1} + a_{m-2}x^{m-2} + ... + a_1 x + a_0 \,, a_i \in \{0,1,..., p-1\} \quad \textbf{(3.2)}$$

Kelime düzeyinde gösterimde $a_i \in GF(p)$ katsayıları s bloğa ayrılır. Her bloğun uzunluğu w olduğu bir durumda vektörün toplam uzunluğu $m = s.w$ olur. Her $A_i(\alpha)$'nın w uzunluklu blokları ifade ettiği durumdaki gösterim aşağıda verilmiştir.

$$A(\alpha) = \sum_{i=0}^{s-1} A_i(\alpha)\alpha^{iw} \,, A_i(\alpha) = a_{iw+w-1}\alpha^{w-1} + ... + a_{iw+1}\alpha^1 + a_{iw} \quad \textbf{(3.3)}$$

Bir diğer gösterim ise normal bazdır. Normal bazlar için gösterim şekli $\{\beta, \beta^p, \beta^{p^2} ..., \beta^{p^{m-2}}\}$ şeklindedir. Bu gösterimde $\beta \in GF(p^m)$ dir. Bu gösterimle $B \in GF(p^m)$ aşağıdaki şekilde gösterilir.

$$B(\beta) = b_{m-1}\beta^{p^{m-1}} + b_{m-2}\beta^{p^{m-2}} + ... + b_1\beta^p + b_0\beta \,, b_i \in GF(p) \quad \textbf{(3.4)}$$

3.4. Galois Alanında Matematiksel İşlemler

Galois alanlarının tanımında da bahsedildiği gibi pozitif tamsayılar kümesinde modülo çarpma ve toplama işlemleri tanımlıdır. Bu işlemler genelde aynı sistematiğe dayanmakla birlikte, kullanılan alana göre küçük farklılıklar ve özel yöntemler içerebilmektedir. Bu tezde $GF(p^m)$ sonlu alanı üzerinde çalışıldığı için $GF(p^m)$ sonlu alanı üzerinde toplama ve çarpma işlemlerinden bahsedilecektir.

3.4.1. $GF(p^m)$ Sonlu Alanında Matematiksel İşlemler

bu bölümde $GF(p^m)$ sonlu alanında polinomsal baz kullanarak matematiksel işlemlerden bahsedilecektir.

Tanım 3-6 da belirtildiği gibi $GF(p^m)$ alanında bir eleman $m-1$. dereceden ve katsayıları $GF(p)$'de olan bir polinomla ifade edilir.

Alanların sınıflandırılmasında $GF(p^m)$ alanı optimum genişletilmiş alan sınıfına dahil edilmiştir. Bu alan için aşağıdaki özellikler mevcuttur;

p asal sayıdır.

$GF(p)$ üzerinde $P(x) = x^m - \omega$ indirgenemez polinomu bulunmaktadır.

x, $GF(p)$ alanında indirgenemez olan $P(x)$ polinomunun kökü olmak üzere;

$$a(x) = \sum a_i x^i = a_{m-1}x^{m-1} + a_{m-2}x^{m-2} + ... + a_1 x + a_0 \text{ ,} a_i \in \{0,1,..., p-1\} \quad \textbf{(3.5)}$$

şeklinde tanımlanır. Optimize genişletilmiş alan oluşturulmasında gerekli olan $GF(p)$'de tanımlı, $P(x) = x^m - \omega$ indirgenemez polinomu seçimi için [22]'den yararlanılabilir.

Bu alanda çalışmanın sağladığı en büyük avantaj küçük m uzaltılmış derece değerleriyle GF(p) alanı ile aynı işlem karmaşıklığı ve dolayısıyla güvenlik sağlanabilmektedir.[6]

3.4.1.1 Toplama ve Çıkarma

Toplama işlemi $A, B \in GF(p^m)$ polinomlarının aynı dereceden terimlerinin $GF(p)$ alanında toplanmasıyla elde edilir. [16]

$$c(x) = a(x) + b(x) = \sum_{i=1}^{n-1} c_i x^i \text{ , } c_i = a_i + b_i \bmod p \quad \textbf{(3.6)}$$

Toplama(çıkarma için de aynı durum geçerlidir) işleminde p ile indirgemenin nedeni a_i ve b_i katsayılarının toplama sonrasında $GF(p)$ içinde tutarak alan tanımını gerçeklemektir.

Burada bahsedilmesi gereken en önemli noktalardan birisi, iki polinomun toplanması sırasında basamaklar arasında elde değeri taşınmamaktadır. Yani iki katsayının toplanması işlemi sonucunda oluşan değer eğer p 'ye eşit ya da büyükse elde değeri

oluşturulup bir üst dereceli katsayıya eklenmez. Bu özellik algoritmanın gerçeklenmesi sırasında büyük kolaylık sağlamaktadır.

3.4.1.2 Çarpma

$GF(p^m)$ alanında çarpma işlemi, $GF(p^m)$ alanı üzerinde polinom çarpımı gerçeklenmesi ve çarpımın daha sonra indirgenemez polinoma bölünüp kalan bulunamasıyla gerçekleşmektedir. $m-1$. dereceden iki polinomun çarpım sonucu $2m-2$ olur.

$$c(x) = a(x)b(x) = \left(\sum_{i=0}^{m-1} a_i x^i\right)\left(\sum_{j=0}^{m-1} b_i x^j\right) = \sum_{i=0}^{m-1}\sum_{i=0}^{m-1} (a_i b_j \bmod p).x^{(i+j)} = \sum_{k=0}^{2m-2} c_k x^k \qquad \textbf{(3.7)}$$

Modülo çarpmanın uzun bir işlem olmasından dolayı, bu genel formül dışında birçok algoritma da geliştirilmiştir.[23][24] Standart algoritmada, ilk polinomun katsayıları ikinci polinom üzerinde gezdirilerek tek tek ara toplamlar elde edilir ve bu değerler birer adım kaydırılıp toplanarak işleme devam edilir. Bu tekniğe genel olarak operatör kaydırma tekniği adı verilir. Performans olarak herhangi bir iyileştirme söz konusu değildir. $a_i, b_i \in GF(p)$ katsayılarıyla m^2 tane çarpma işlemi yapılmaktadır.

Geliştirilmiş modülo çarpma algoritmalarından en çok kullanılanı çarpım kaydırma metodu [21] ve Karatsuba Algoritmasıdır.[25] İlk algoritma yine aynı sayıda çarpma işlemi yapmaktadır ancak bilgisayarda gerçekleme sırasında daha az belleğe ihtiyaç duymaktadır.[21] Karatsuba ise çok daha az katsayı çarpma işlemi gerçekleştirmektedir. [26]

Yukarıda bahsedilen algoritmalar çarpma işlemini gerçeklerken sağladığı avantajlar gözardı edilememekle birlikte modülo işleminde hala bir darboğaz yaşanmasını engelleyememektedir. Modülo işleminin gerek yazılım tabanlı gerekse donanım tabanlı sistemlerde gerçekleme zorluğunu aşmak için MMÇ algoritması [7] önerilmektedir. Bu algoritmada çarpma işlemi farklı bir yolla yapılıp algoritma sonucunda modülo işlemine gerek kalmamaktadır. Bu tez çalışmasında Montgomery tekniği kullanılmış olduğu için algoritmadan ilerleyen bölümlerde daha detaylı bahsedilecektir.

4. ELİPTİK EĞRİ TEMELLERİ

Tanım 4-1 EliptikEğri

K gibi bir alan üzerinde tanımlı bir Eliptik Eğri, aşağıda verilen Weierstrass denkleminin çözüm kümesi ve sonsuz Ó noktalarının birleşiminden oluşmaktadır. [27]

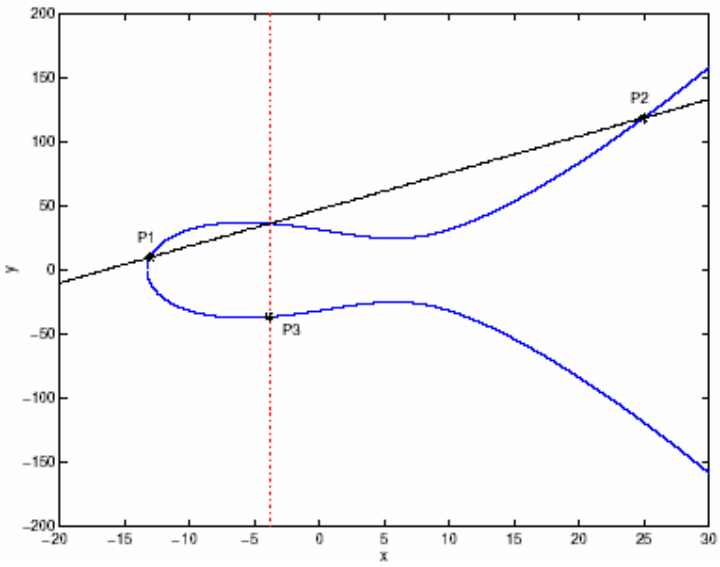
$$y^2 + a_1xy + a_5{}^3y = x^3 + a_2x^2 + a_4x + a_6 \qquad \textbf{(4.1)}$$

Bu çözüm kümesi nokta toplama işlemi ile bir Abelian Grup oluşturmaktadır. Daha önce bahsedildiği gibi oluşan grup, yani eliptik eğriyi oluşturan noktalar topluluğu, kriptografik sistemde kullanılabilir.[28] Oluşturulan grubun birim elemanı y ekseni üzerinde olduğu varsayılan sonsuzdaki Ó noktasıdır.

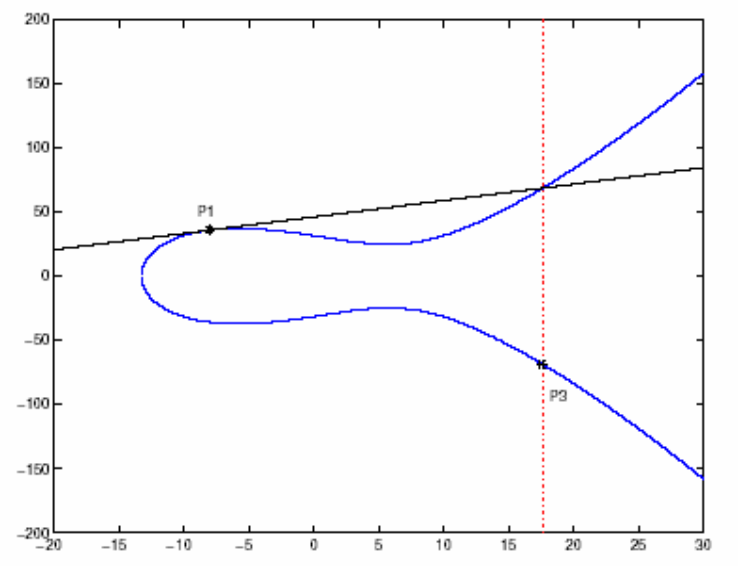
Çözüm kümesi bir grup oluşturduğuna göre, eliptik eğri üzerinde nokta toplama ve nokta çarpma işlemleri gerçekleştirilebilmektedir. P_1 ve P_2, E eliptik eğrisi üzerinde tanımlı iki nokta olmak üzere bu noktaların toplamı aşağıda verilen yöntemle bulunur;

P_1 ve P_2 noktaları bir doğruyla birleştirilir. Bu iki noktayı birleştiren doğru eliptik eğriyi Q gibi bir üçüncü noktada daha keser. Bu Q noktasının x eksenine göre simetriğinin alınmasıyla $P_1 + P_2$ değeri bulunur. Özel durumlar ele alınacak olursa , $P_1 = P_2$ gibi bir seçimde noktaları kesen doğru P_1 noktasından geçen eğriye teğet geçen doğrudur.

Yukarıda bahsedilen nokta toplama ve nokta ikileme işlemleri anlatımı kolaylaştırmak amacıyla aşağıda $\Re$'de tanımlı bir eliptik eğri üzerinde görsel olarak verilmiştir.



Şekil 4.1 : $\Re$'de Denklemi $y^2 = x^3 - 100x + 6000$ Olan Eliptik Eğride Toplama



Şekil 4.2 : $\Re$'de Denklemi $y^2 = x^3 - 100x + 6000$ Olan Eliptik Eğride İkileme

Yukarıda $\Re$'de geometrik olarak verilen nokta toplama ve nokta çarpma işlemleri matematiksel olarak formüle edilebilmektedir. Bu formülasyon sayesinde eliptik eğrinin çalışıldığı sonlu alandan bağımsız olarak grup işlemleri için ilgili eşitlikler oluşturulabilmektedir yani aşağıda verilen eşitlikler eğri karakteristiğinden bağımsızdır. $P_1 = (x_1, y_1), P_2 = (x_2, y_2)$ eğri üzerinde iki nokta olmak üzere, bu eğri üzerindeki bir noktanın tersi, $-P$ olup aşağıdaki gibi fade edilir,

$$-P_1 = (x_1, -y_1 - a_1 x_1 - a_3) \quad \textbf{(4.2)}$$

Eğri için genel formüller aşağıdaki gibidir;

$$\lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1}, & P \neq Q \\ \dfrac{3x_1^{\,2} + 2a_2 x_1 + a_4 - a_1 y_1}{2y_1 + a_1 x_1 + a_3}, & P = Q \end{cases} \quad \textbf{(4.3)}$$

$$\mu = \begin{cases} \dfrac{y_1 x_2 - y_2 x_1}{x_2 - x_1}, & P \neq Q \\ \dfrac{-x_1^{\,3} + a_4 x_1 + 2a_6 - a_3 y_1}{2y_1 + a_1 x_1 + a_3}, & P = Q \end{cases} \quad \textbf{(4.4)}$$

Bu eşitlikler kullanılarak $P + Q = (x_3, y_3) \neq$ Ó (sonsuz) için genel formüller aşağıdaki gibidir;

$$x_3 = \lambda^2 + a_1\lambda - a_2 - x_1 - x_2 \quad \textbf{(4.5)}$$

$$y_3 = -(\lambda + a_1)x_3 - \mu - a_3 \quad \textbf{(4.6)}$$

Yukarıdaki işlemlerde görüldüğü üzere nokta toplanma ve nokta ikileme işleminde çarpma, kare alma, toplama ve çarpmaya göre ters alma işlemlerine ihtiyaç duyulmaktadır. Çarpmaya göre ters alma işlemini donanım olarak gerçekleme yüksek alan gerektirir. Değişkenler üzerinde dönüşümler yaparak projektif koordinat sistemine geçilirse, bu koordinat sisteminde çarpmaya göre ters alma sayısı azaltılır. Projektif koordinat sistemi bir sonraki bölümde ele alınacak olup bu çalışmada kullanılacaktır.

Yukarıda verilen nokta toplama ve nokta ikileme eşitlikleri eliptik eğrinin üzerinde tanımlı olduğu sonlu alan için özelleştirilebilmektedir.

4.1. $GF(p)$ Alanı Üzerindeki Eliptik Eğriler

Özelleştirilmiş eliptik eğriler arasında en genel alanda tanımlı eğriler, $GF(p)$'de tanımlı olan ya da başka bir deyişle alan karakteristiği 2 ya da 3 olmayan ($char(K) \neq 2,3$) eliptik eğrilerdir. [27]

Eğer eliptik eğri alan karakteristiği 2 ya da 3 olmayan bir alanda tanımlanmışsa eşitlik 4.1 ile verilen Weierstrass denklemi sadeleştirilebilir.

Eğer $char(K) \neq 2$ ise değişkenlere aşağıdaki dönüşüm uygulanabilir;

$$(x, y) \rightarrow (x, y - \frac{a_1}{2}x - \frac{a_3}{2}) \quad \textbf{(4.7)}$$

Bu dönüşüm eğriye aşağıdaki gibi yansır;

$$E' : y^2 = x^3 + a_2x^2 + a_4x + a_6 \quad \textbf{(4.8)}$$

Bu noktada $GF(p)$'da $E \cong E'$ denkliği mevcuttur.[27]

Ayrıca eğer $char(K) \neq 2,3$ değilse;

$$(x, y) \rightarrow (\frac{x - 3b^2}{36}, \frac{y}{216}) \quad \textbf{(4.9)}$$

E' ne dönüşümü ile E'' eliptik eğri denklemi 4.10 eşitliğindeki gibi verilir;

$$E'' : y^2 = x^3 + a_4x + a_6 \quad \textbf{(4.10)}$$

Bu noktada da aynı benzerlik, $E' \cong E''$ mevcuttur. Dolayısıyla $GF(p)$ alanında $E \cong E''$ denkliği söylenebilir. [27]

Yukarıdaki dönüşümler sonucunca $char(K) \neq 2,3$ durumu için E eliptik eğrisi denklemi $a_1 = a_2 = a_3 = 0$ için

$$E : y^2 = x^3 + ax + b, a, b \in GF(p) \quad \textbf{(4.11)}$$

şeklindendir.

Bu denklemde verilen $a, b \in GF(p)$ değişkenleri, eliptik eğri oluşturulabilmesi için $4a^3 + 27b^2 \neq 0 \bmod(p)$ koşulunu sağlamalıdır. Bu koşul ise eğrinin tanımlanabilmesi için diskriminantının var olması gerekmesinden gelmektedir. Bu eğrinin diskriminantı ise

$$\Delta = -16(4a^3 + 27b^2) \quad \textbf{(4.12)}$$

olarak tanımlanır. Diskriminanttan yararlanarak da eğrinin j^{th} invariant değeri $j(E)$ hesaplanabilir. [29]

Bu iki değerin önemi iki teoremle açıklanmaktadır.

Teorem 4-1 : Herhangi bir alanda tanımlanmış E eliptik eğrisinin tekil olmayan olabilmesi için $\Delta = -16(4a^3 + 27b^2)$ değerinin sıfırdan farklı olması gerekmektedir. [28]

Teorem 4-2 :Aynı K alanı üzerinde tanımlanmış iki eliptik eğri E_1, E_2 izomorfik olabilmesi ancak ve ancak $j(E_1) = j(E_2)$ koşulu altında sağlanabilmektedir. [28]

Diskriminant ve j^{th} invariant hesabı için daha detaylı bilgi [28]' de verilmektedir.

$$j = 12^3 \frac{4a_4{}^3}{(4a^3 + 27b^2)} \tag{4.13}$$

Eğri üzerinde tanımlanan toplama işlemi daha önceden belirtilen grup özellikelerinin tamamına uymaktadır.

$P = (x_1, y_1), Q = (x_2, y_2) \in E$ eliptik eğrisi olmak üzere, $GF(p)$ için $P + Q = (x_3, y_3)$, $\mathrm{P} + \mathrm{P} = 2\mathrm{P} = (\mathrm{x}_3, \mathrm{y}_3)$ nokta ikileme işlemleri ve ilgili ara değerler aşağıdaki gibi ifade edilir;

$$\lambda = \begin{cases} \dfrac{y_2 - y_1}{x_2 - x_1}, & P \neq Q \\ \dfrac{3x^2{}_1 + a}{2y_1}, & P = Q \end{cases} \tag{4.14}$$

$$x_3 = \begin{cases} \lambda^2 - x_1 - x_2, & P \neq Q \\ \lambda^2 - 2x_1, & P = Q \end{cases} \tag{4.15}$$

$$y_3 = \begin{cases} \lambda(x_1 - x_3) - y_1, & P \neq Q \\ \lambda(x_1 - x_3) - y_1, & P = Q \end{cases} \tag{4.16}$$

4.2. $GF(2^m)$ Alanı Üzerindeki Eliptik Eğriler

$GF(2^m)$ alanında tanımlı ya da başka bir deyişle alan karakteristiği $char(K) = 2$ olan eliptik eğrilerdir.

$char(K) = 2$ olan eliptik eğriler için 4.1 eşitliği ile verilen Weierstrass denklemi sadeleştirilebilir.

Eğer $char(K) = 2$ ise eğrinin j^{th} invariant değeri $j(E)$ hesaplanır.

Eğer $j(E) \neq 0$ ise değişkenlere aşağıdaki dönüşüm uygulanabilir;

$$(x, y) \rightarrow (a_1^{\,2} x + \frac{a_3}{a_1}, a_1^{\,3} y + \frac{a_1^{\,2} a_4 + a_3^{\,2}}{a_1^{\,3}}) \qquad \textbf{(4.17)}$$

Bu dönüşüm eğriye aşağıdaki gibi yansır;

$$E' : y^2 + xy = x^3 + a_2 x^2 + a_6 \qquad \textbf{(4.18)}$$

Bu noktada $GF(2^m)$'da $E \cong E'$ denkliği sağlanır.[28] Bu eğri için $\Delta = a_6$ ve $j(E) = \frac{1}{a_6}$ olur

Ayrıca eğer $j(E) = 0$ ise ;

$$(x, y) \rightarrow (x + a_2, y) \qquad \textbf{(4.19)}$$

Dönüşümü uygulanarak 4.1 denklemindeki x^2'li terim elenir ve E' eliptik eğri denklemi aşağıdaki gibi verilir;

$$E' : y^2 + a_3 y = x^3 + a_4 x + a_6 \qquad \textbf{(4.20)}$$

Bu eğri için $\Delta = a_3^{\,4}$ ve $j(E) = 0$ dır.

Görüldüğü gibi eğrinin farklı $j(E)$ değerleri için farklı eğri denklemleri ve dolayısıyla farklı eşitlikler elde edilmektedir. Aşağıda kısaca formüle edilmiş hali ile;

$$E : \begin{cases} y^2 + xy = x^3 + a_2 x^2 + a_6, & j(E) \neq 0 \\ y^2 + a_3 y = x^3 + a_4 x + a_6, & j(E) = 0 \end{cases} \qquad \textbf{(4.21)}$$

şeklindendir.

$j(E) \neq 0$ için;

$GF(2^m)$ için $P = (x_1, y_1)$, $Q = (x_2, y_2) \in E$ eliptik eğrisi olmak üzere, $P + Q = (x_3, y_3) = P_3$ ve ilgili ara değerler aşağıdaki gibidir;

$$-P = (x_1, y_1 + x_1) \tag{4.22}$$

$$\lambda = \begin{cases} \dfrac{y_1 + y_2}{x_1 + x_2}, & P \neq Q \\ x_1 + \dfrac{x_1}{y_1}, & P = Q \end{cases} \tag{4.23}$$

$$x_3 = \begin{cases} \lambda^2 + \lambda + x_1 + x_2 + a_2, & P \neq Q \\ \lambda^2 + \lambda + a_2, & P = Q \end{cases} \tag{4.24}$$

$$y_3 = \left\{ \begin{matrix} \lambda(x_1 + x_3) + x_3 + y_1, P \neq Q \\ \lambda(x_1 + x_3) + x_3 + y_1, P = Q \end{matrix} \right\} \tag{4.25}$$

$j(E) = 0$ için;

$$-P = (x_1, y_1 + a_3) \tag{4.26}$$

$$\lambda = \begin{cases} \dfrac{y_1 + y_2}{x_1 + x_2}, P \neq Q \\ \dfrac{{x_1}^2 + a_4}{a_3}, P = Q \end{cases} \tag{4.27}$$

$$x_3 = \begin{cases} \lambda^2 + x_1 + x_2 = \lambda^2, P \neq Q \\ \lambda^2 \left(= \dfrac{{x_1}^4 + {a_4}^2}{{a_3}^2} \right), \quad P = Q \end{cases} \tag{4.28}$$

$$y_3 = \begin{cases} \lambda(x_1 + x_3) + y_1 + a_3, P \neq Q \\ \lambda(x_1 + x_3) + y_1 + a_3, P = Q \end{cases} \tag{4.29}$$

4.3. $GF(3^m)$ Alanı Üzerindeki Eliptik Eğriler

$GF(3^m)$ alanında tanımlı ya da başka bir deyişle alan karakteristiği $char(K) = 3$ olan eliptik eğrilerdir.

$char(K) = 3$ olan eliptik eğriler için 4.1 eşitliği ile verilen Weierstrass denklemi sadeleştirilebilir.

$char(K) = 3$ için eğrinin j^{th} invariant değeri $j(E)$ hesaplanır.

$j(E) \neq 0$ için değişken dönüşüm aşağıdaki gibi uygulanabilir;

$$(x, y) \rightarrow (x + \frac{a_4}{a_2}, y) \quad \textbf{(4.30)}$$

Bu dönüşüm eğriye şu şekilde yansır;

$$E' : y^2 = x^3 + a_2 x^2 + a_6 \quad \textbf{(4.31)}$$

Bu noktada $GF(3^m)$' da $E \cong E'$ olduğu açıktır. Bu eğri için

$$\Delta = -a_2{}^3 a_6 \quad \textbf{(4.32)}$$

$$j(E) = -\frac{a_2{}^3}{a_6} \quad \textbf{(4.33)}$$

bulunur.

Ayrıca eğer $j(E) = 0$ ise yani $a_2 = 0$ ise E' zaten istenen formdadır yani x^2'li terimin katsayısı sıfırdır eşitlikte görünmez, değişken dönüşümüne ihtiyaç yoktur.

$$E' : y^2 = x^3 + a_4 x + a_6 \quad \textbf{(4.34)}$$

Bu eğri için $\Delta = -a_3{}^4$ ve $j(E) = 0$ dır.

Görüldüğü gibi eğrinin farklı $j(E)$ değerleri için farklı eğri denklemleri ve dolayısıyla farklı eşitlikler elde edilmektedir.

$$E:\begin{cases} y^2 = x^3 + a_2x^2 + a_6, j(E) \neq 0 \\ y^2 = x^3 + a_4x + a_6, j(E) = 0 \end{cases} \quad \textbf{(4.35)}$$

$P = (x_1, y_1), Q = (x_2, y_2) \in E$, $Q \neq -P$ ve $\lambda \in \mathrm{GF}(3^m)$ olmak üzere, $P + Q = (x_3, y_3)$ aşağıdaki gibi ifade edilir;

$j(E) \neq 0$ $y^2 = x^3 + a_2x^2 + a_6$ eğrisi için;

$$-P = (x_1, -y_1) \quad \textbf{(4.36)}$$

$$\lambda = \begin{cases} \dfrac{y_1 - y_2}{x_1 - x_2}, & P \neq Q \\ \dfrac{3x_1^2 + 2a_2x_1}{2y_1} (char(E) = 3 \rightarrow \lambda = \dfrac{a_2x_1}{y_1}), & P = Q \end{cases} \quad \textbf{(4.37)}$$

$$x_3 = \begin{cases} \lambda^2 - x_1 - x_2 - a_2, P \neq Q \\ \lambda^2 - 2x_1 - a_2, \quad P = Q \end{cases} \quad \textbf{(4.38)}$$

$$y_3 = \begin{cases} (x_1 - x_3)\lambda - y_1, P \neq Q \\ (x_1 - x_3)\lambda - y_1, P = Q \end{cases} \quad \textbf{(4.39)}$$

$j(E) = 0$ yani $y^2 = x^3 + a_4x + a_6$ eğrisi için;

$$-P = (x_1, -y_1 - a_3) \quad \textbf{(4.40)}$$

$$\lambda = \begin{cases} \dfrac{y_1 - y_2}{x_1 - x_2}, & P \neq Q \\ \dfrac{3x_1^2 + a_4}{2y_1} (char(K) = 3 \rightarrow a_4 = -1, \lambda = \dfrac{1}{y_1}), & P = Q \end{cases} \quad \textbf{(4.41)}$$

$$x_3 = \begin{cases} \lambda^2 - x_1 - x_2, (char(E) = 3 \rightarrow x_3 = \lambda^2 + 2x_1 + 2x_2) P \neq Q \\ \lambda^2 - 2x_1 (char(E) = 3 \rightarrow x_3 = \lambda^2 + x_1), \quad P = Q \end{cases} \quad \textbf{(4.42)}$$

$$y_3 = \begin{cases} (x_1 - x_3)\lambda - y_1, P \neq Q \\ (x_1 - x_3)\lambda - y_1, P = Q \end{cases} \quad \textbf{(4.43)}$$

Yukarıda verilen nokta toplam ve nokta ikileme işlemleri için, projektif koordinat sistemindeki karşılıkları Bölüm 4.4 de verilecektir. Bu çalışma kapsamındaki $GF(3^m)$ sonlu alanı için eşitlikler Bölüm 4.4.1'de detaylı olarak incelenmiştir.

4.4. Projektif Koordinatlar

Tanım 4-2 Eşdeğerlilik Bağıntısı

$\sim, A$ kümesi üzerinde tanımlanmış bir bağıntı olmak üzere. $\sim$, $\forall x, y, z \in A$ için yansıma, simetri ve geçişlilik özelliklerini sağlıyorsa, $\sim$ bir eşdeğerlilik bağıntısıdır. [28]

Projektif koordinatlar ile ilgin koordinatlar birbiriyle yukarıdaki tanımda verilen ilişkiye sahiptir, yani birbiri ile eşdeğer tanım kümeleridir.

Projektif koordinatların kullanılma amacı; ilgin koordinat sisteminde işlemleri yaparken ortaya çıkan işlem sayısı fazla adımları farklı bir kümede çalışarak daha kolay bir şekilde gerçeklemektir.

İlgin koordinatlarda çalışma esnasında eliptik eğri kriptosistemlerinde yeralan en önemli işlemler olan toplama ve çarpma işlemlerinin yanında donanımsal olarak sistemde en önemli darboğazı oluşturan çarpmaya göre ters alma işlemi bulunmaktadır. Çarpmaya göre ters alma işleminin getirdiği işlem sayısı maliyeti yüzünden projektif koordinatlara geçilir. Bu geçişle çok daha fazla çarpma işlemi gerçekleşmesine rağmen çarpmaya göre ters alma işleminin olmadığı bir çalışma kümesi elde edilir. Bir çarpmaya göre ters alma işlemi, elemanları 100 bitle ifade edilen bir alan için yaklaşık olarak 30 çarpma işlemine karşılık gelecek maliyete sahiptir. [15] Koordinatlar arası dönüşümde, diğer işlemlere göre herhangi bir sistem maliyeti oluşmaz. Sadece projektif koordinatlardan ilgin koordinat sistemine geçişte bir adet çarpmaya göre ters alma işlemi yapılır ki bu da tüm işlemleri normal koordinatlarda gerçeklerken ortaya çıkan birçok çarpmaya göre ters alma işlemi maliyeti yanında çok azdır.

4.4.1. $GF(p^m)$ Alanlarında Projektif Koordinatlar

Bu bölümde, Bölüm 4.3'de ilgin koordinat sistemi için verilen nokta toplama ve nokta ikileme işlemleri için projektif koordinatlardaki karşılıkları verilecektir.

$Char(K) = 3$ eliptik eğrileri için projektif koordinat dönüşümü $(x, y) = (X / Z^2, Y / Z^3)$ şeklindedir. [30]

Bu dönüşümdeki eşitlikler aşağıdaki gibidir;

İlgin koordinatlarından projektif koordinatlara geçiş;

$$(x, y) \to (X, Y, Z) = (x.Z^2, y.Z^3, Z) \quad \textbf{(4.44)}$$

Projektif koordinatlardan ilgin koordinatlara geçiş;

$$(X, Y, Z) \to (x, y) = (X / Z^2, Y / Z^3) \quad \textbf{(4.45)}$$

İlgin koordinatlardan projektif koordinatlara dönüşüm sonucunda $Char(K) = 3$ için aşağıda verilen E eliptik eğrisi denklemleri;

$$E : \begin{cases} y^2 = x^3 + a_2 x^2 + a_6, j(E) \neq 0 \\ y^2 = x^3 + a_4 x + a_6, j(E) = 0 \end{cases} \quad \textbf{(4.46)}$$

aşağıdaki gibi değişir;

$$E : \begin{cases} Y^2 = X^3 + a_2 X^2 Z^2 + a_6 Z^6, j(E) \neq 0 \\ Y^2 = X^3 + a_4 X Z^4 + a_6 Z^6, j(E) = 0 \end{cases} \quad \textbf{(4.47)}$$

$j(E) = 0$ için nokta toplama, nokta ikileme ve nokta üçleme işlemleri aşağıda verilmektedir.

$P = (x_1, y_1, z_1)$ ve $Q = (x_2, y_2, z_2)$ E eliptik eğrisi üzerindeki noktaların projektif koordinatlardaki gösterimi olmak üzere, $P + Q = (x_3, y_3, z_3)$ ifade edilir;

Nokta Toplama:

$\lambda_1 = x_1 z_2^{\,2}$ 2Çarpma

$\lambda_2 = x_2 z_1^{\,2}$ 2Çarpma

$$\lambda_3 = \lambda_1 - \lambda_2$$

$$\lambda_4 = y_1 z_2^{\,3} \qquad \text{1Çarpma, 1KüpAlma}$$

$$\lambda_5 = y_2 z_1^{\,3} \qquad \text{1Çarpma, 1KüpAlma}$$

$$\lambda_6 = \lambda_4 - \lambda_5$$

$$\lambda_7 = \lambda_1 + \lambda_2$$

$$\lambda_8 = \lambda_4 + \lambda_5$$

$$z_3 = z_1 . z_2 . \lambda_3 \qquad \text{2Çarpma}$$

$$x_3 = \lambda_6^{\,2} - \lambda_7 . \lambda_3^{\,2} \qquad \text{3Çarpma}$$

$$y_3 = \lambda_8 . \lambda_3^{\,3} - \lambda_6^{\,3} \qquad \text{1Çarpma, 2KüpAlma}$$

Nokta İkileme:

$P = (x_1, y_1, z_1)$, E eliptik eğrisi üzerindeki noktanın projektif koordinatlardaki gösterilimi olmak üzere, $P + P = (x_3, y_3, z_3)$ aşağıdaki gibi ifade edilir;

$$\lambda_1 = -z_1^{\,4} \qquad \text{1 Çarpma, 1 KüpAlma}$$

$$z_3 = -y_1 . z_1 \qquad \text{1 Çarpma}$$

$$\lambda_2 = x_1 y_1^{\,2} \qquad \text{2 Çarpma}$$

$$x_3 = \lambda_1^{\,2} + \lambda_2 \qquad \text{1 Çarpma}$$

$$\lambda_3 = -y_1^{\,4} \qquad \text{1 Çarpma, 1 KüpAlma}$$

$$y_3 = \lambda_1 . (\lambda_2 - x_3) - \lambda_3 \qquad \text{1 Çarpma}$$

4.5. Eliptik Eğri Üzerinde Skaler Nokta Çarpma İşlemi

Eliptik eğri üzerindeki en temel işlemler bir noktanın k skaler sayısı ile çarpımıdır.

Q ve P tanımlanan eğri üzerinde noktalar, k bir tamsayı olmak üzere nokta çarpma işlemi aşağıdaki gibidir;

$$Q = k.P \qquad \textbf{(4.48)}$$

Eğri üzerinde tanımlanmış noktalar arasında olan ilişkiyi oluşturma maliyeti, skaler çarpma işleminin karmaşıklığına bağlıdır. Çok sık kullanılan bir işlem olmasından dolayı bu işlemin en hızlı şekilde gerçekleşmesi sistemin performansı için önemlidir bu sebeple birçok etkili algoritmalar geliştirilmiştir.

Skaler çarpma işlemi P noktasının kendisiyle $(k-1)$ defa toplanması anlamına geldiği için aşağıdaki gibi bir gösterim kullanılabilir;

$$Q = \underbrace{P + P + ... + P}_{k-a\ \det toplama} \qquad \textbf{(4.49)}$$

Çok büyük k değerleri için $Q = k.P$ işlemi çok uzun sürmekte ve fazla işlem gerektirmektedir. Bu durumda yukarıda bahsedildiği gibi ardarda toplayarak çarpmayı gerçekleme yavaş olacaktır.

$Q = k.P$ çarpması için ikile-ve-topla algoritması kullanarak işlem gerçeklenebilir. [9]

Bu teknikte k değerinin ikili tabanda gösterimi kullanılarak aşağıdaki gibi bir algoritma uygulanır.

Algoritma 4-1 : İkile-ve-Topla Algoritması

GİRİŞ: k_{n-1}=1 olmak üzere $K = k_{n-1}2^{n-1} + k_{n-2}2^{n-2} + ... + k_1 2 + k_0$ ve $P = (x, y)$

ÇIKIŞ: $Q = kP = (x', y')$

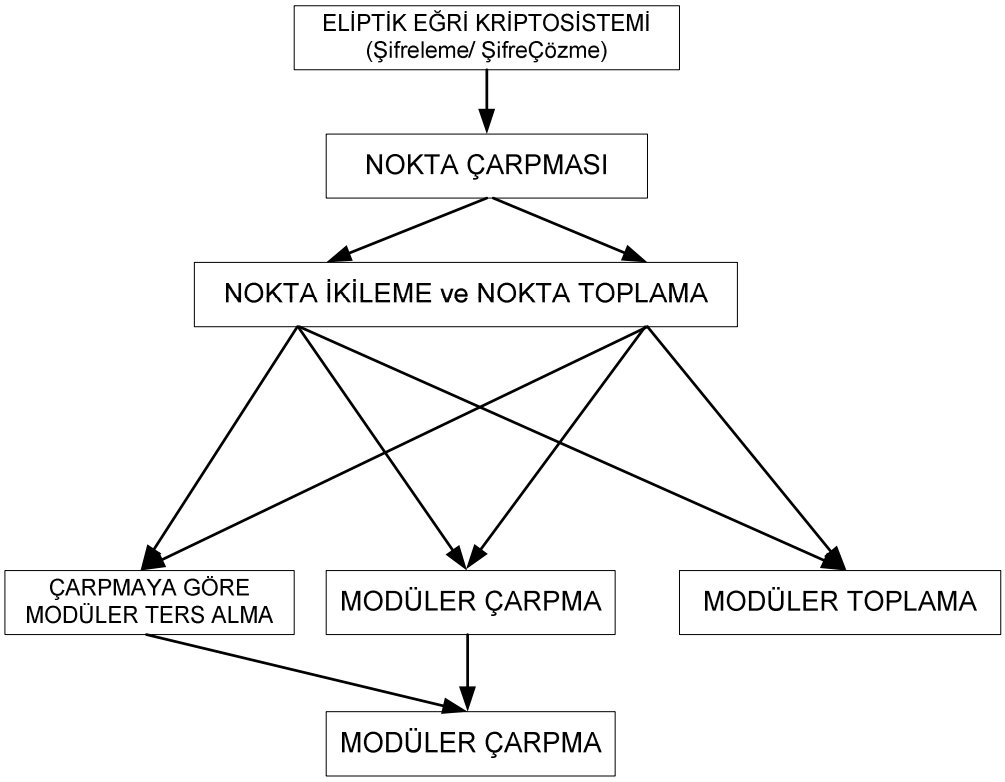
```
1.      Q <- P
2.      for i from n − 2 downto 0 do
3.        Q <- 2Q
4.          if k_i = 1 then
5.            Q = Q + P
6.          end if
7.      end for
```

Görüldüğü gibi sıradan ardarda toplama işleminde örneğin $25P$ değeri için 24 tane ardışıl toplama uygulanacaktı ancak bu algoritmayla dört çiftleme, iki toplama işlemi ile istenene sonuca ulaşılmıştır. Yukarıdaki sonuç genelleştirilirse bu algoritma

kullanılarak $l-bit$ uzunluklu k sayısı için $(l-1)$ çiftleme işlemi ve ortalama $(l-1)/2$ toplama işlemi ile çarpma tamamlanır. Toplam karmaşıklık $\frac{3}{2}(l-1)$ dir.

Yukarıdaki algoritmada görüldüğü gibi nokta çarpma işlemi, nokta toplama ve nokta çiftleme işlemleri kullanılarak yapılmaktadır. Daha önceden belirtildiği gibi eliptik eğri üzerinde toplama işlemi yaparken çarpmaya göre ters alma işleminden kurtulmak için projektif koordinat sistemine geçilebilir. Böylece sadece projektif koordinatlardan ilgin koordinatlara geçerken bir kere çarpmaya göre ters alma işlemi yapılır bunun dışında çarpma, kare alma ve toplama işlemi vardır.

Sonuç olarak, projektif koordinatların kullanıldığı bir nokta çarpma işleminde; nokta çiftleme ve nokta toplama işlemleri mevcuttur. Bu iki işlemin gerçeklenebilmesi için de modüler çarpmaya göre ters alma, modüler çarpma ve modüler toplama işlemlerine ihtiyaç vardır. İşlemlerin hiyerarşisi aşağıdaki şekilde açıkça belirtilmiştir.



Şekil 4.3 : Nokta Çarpma İşlemi ve Bileşenleri

4.6. Montgomery Modüler Çarpması

Eliptik eğri üzerindeki iki noktanın toplamının sonucu, (x_3, y_3) bulmak için çarpmaya göre ters alma işleminden neden sakınmak gerektiği daha önceden belirtilmişti. Bu amaçla projektif koordinatlara geçildikten sonra nokta toplama ve nookta çiftleme işlemleri için eliptik eğri üzerinde belirtilen noktaların projektif koordinatları ile çarpma ve toplama işlemi yapılacaktır. Bu işlemler sırasında doğal olarak çalışılan alandan dolayı modüler çarpma işlemi yapılır.

1985 yılında Montgomery, modüler çarpma için yeni bir yöntem önermiştir.[7] MMÇ algoritması sonucu modüler olarak indirgenmiş halde oluşur. Böylece her çarpma sonucunda bölme işleminden kurtulmuş olunur. Sistemi donanım olarak gerçeklerken modüler indirgeme işlemi için ayrı bir blok tasarımına da ihtiyaç duyulmaz.

$c = ab \bmod N$ şeklinde bir çarpma işlemi için sıradan çarpma yöntemi kullanıldığında çarpma için k kere k-bit toplama ve k kere k-bit çıkarma ve bölme için karşılaştırma yapmak gerekir.[7] Montgomery metodunda ise bölme işlemi yerine daha sonra açıklanacak olan özel bir değerle bölme yapılır. Buradan da görüleceği gibi sadece hız anlamında değil aynı zamanda gerçeklemede de işlem maliyeti vardır.

$GF(p)$ için Montgomery çarpması aşağıdaki şekilde ifade edilir;

$$Mont(x, y) = xyR^{-1} \bmod N \qquad \textbf{(4.50)}$$

Bu eşitlikte verilen sayıları eğer belli bir sayı tabanında ifade edersek ve R değeri de b tabanında n basamaklı bir sayı olmak üzere , $R = b^n > N$ olacak şekilde bir değerdir. Çarpıma giren her $x \in Z_N$ değerin bir de Montgomery gösterimi mevcuttur şöyle ki;

$$x' = Mont(x, R^2) = x.R \pmod N \qquad \textbf{(4.51)}$$

İki sayının modüler çarpmasını elde etmek için ise x' ile y değerlerinin Montgomery çarpması hesaplanır;

$$c = Mont(x', y) = xR.yR' \pmod N = x.y \pmod N \qquad \textbf{(4.52)}$$

Eğer iki sayıya Montgomery dönüşümü yapılırsa ve bu yeni değerlerle Montgomery çarpması yapılırsa ;

$$c'=Mont(x', y') = x.R.y.R.R' (\text{mod}\, N) = x.y.R \ (\text{mod}\, N) \quad \textbf{(4.53)}$$

değeri elde edilir. Bu değer asıl sonuç için geçiş değeri olup bu ara değer ile '1' değeri Montgomery çarpımı yapılırsa;

$$c =Mont(c',1) = x.y.R.1.R^{-1} (\text{mod}\, N) = x.y \ (\text{mod}\, N) \quad \textbf{(4.54)}$$

4.54 eşitliği ile istenen sonuç elde edilir.
MMÇ algoritması Algoritma 4.2 de verilmektedir.

Algoritma 4-2 : Çıkarma İşlemi olan Montgomery Modüler Çarpma Algoritması

Girişler: $N = (n_{l-1} n_{l-2...} n_1 n_0)_b$, $x = (x_{l-1} x_{l-2...} x_1 x_0)_b$, $y = (y_{l-1} y_{l-2...} y_1 y_0)_b$ x ve $y \in [0, N-1]$, $R = b^l$ ve $obeb(N,b) = 1$ and $N' = -N^{-1} \bmod b$.

Çıkış : $x.y.R^{-1} \bmod N$

1: $T \leftarrow 0$ (Gösterilim $T = (t_l t_{l-1} \cdots t_1 t_0)_b$)

2: **for** i **from** 0 **to** $l-1$ **do**

3: $u_i \leftarrow (t_0 + x_i.y_0) N' \bmod b$

4: $T \leftarrow (T + x_i.y + u_i.N)/b$

5: **end for**

6: **if** $T \geq N$ **then**

7: $T \leftarrow T - N$

8: **end if**

9: **Return** T

Girişler $x, y < N$ olacak şekilde sınırlandırılmıştır ve çıkış değeri $T < N$ olarak oluşur. $T > N$ yani modülo N tanım kümesi dışına çıkma durumunda bir çıkarma yapılmalıdır ve çıkarma işleminden sonra bir sonraki turda çarpmaya giriş olarak verilebilmektedir. Bu işlemden kurtulmak için R için tanımlanan değer $R = b^{l+2}$ şeklinde ve giriş değerleri de $x, y < 2N$ olarak değiştirilir. Sonuç değeri de $T < 2N$ olarak oluşur. [31]

Algoritma 4-3 : Sonda Çıkarma Yapılmayan Montgomery Modüler Çarpma Algoritması

Girişler : $N=(n_{l-1}n_{l-2...}n_1n_0)_b$, $x=(x_lx_{l-1}\cdots x_1x_0)_b$, $y=(y_ly_{l-1}\cdots y_1y_0)_b$, x ve $y\in[0,2N-1]$, $R=b^{l+2}$ ve $obeb(N,b)=1$ and $N'=-N^{-1} \bmod b$.

Çıkışlar : $x.y.R^{-1} \bmod 2.N$

1: $T \leftarrow 0$

2: **for** i **from** 0 **to** $l+1$ **do**

3: $u_i \leftarrow (t_0 + x_i.y_0)N' \bmod b$

4: $T \leftarrow (T + x_i.y + u_i.N)/b$

5: **end for**

6: **Return** (T)

Sonda çıkarma yapılmayan değiştirilmiş Montgomery çarpma algoritması ile döngü içinde fazladan iki tur arttırılarak ve giriş değerlerinden bazılarının birer bit fazlasını alarak karşılaştırma ve çıkarma devresi maliyetinden sakınılmış oluyor. Aynı zamanda bu değişiklikle zamanlama analizi saldırılarına karşı da koruma sağlanmaktadır.[32]

Bu algoritmada tüm değerler $\bmod 2N$ olacak şekilde hesaplanmaktadır. Sonuç Montgomery gösterilimi şeklinde elde edilir. Bu sonuçtan asıl istenen değeri elde etmek için ise girişleri T ve 1 olan bir tane daha $Mont(T,1) \le N$ şeklinde Montgomery çarpması yapılır.

4.7. $GF(3^m)$ Sonlu Alanı Üzerinde Montgomery Modüler Çarpması

Yukarıda detaylı şekilde bahsedilen MMÇ devresi, bu çalışmada kullanılan sonlu alan $GF(3^m)$ için özelleştirildiğinde matematiksel gösterilimler ve kullanılan bazı değerler değişmektedir.

$GF(p^m)$ sonlu alanında polinomsal bazda gösterilimi yapılan iki sayının Montgomery çarpması yapılırken polinomsal baz gösterilimi ve kullanılan sonlu alandan dolayı algoritma sonunda yapılan çıkartma işlemi uygulanmaz. [33]

Polinomsal gösterilimde iki sayının Montgomery çarpması, $r(x)=x^k$ olmak üzere $c(x)=a(x)b(x)r^{-1}(x) \bmod (n(x))$ çarpımı Algoritma 4-4 ile hesaplanır;

Algoritma 4-4 : $GF(3^m)$ için Bit Düzeyinde Montgomery Modüler Çarpma Algoritması

Giriş: $a(x) = \left(a_{k-1}x^{k-1} + a_{k-2}x^{k-2} + ... + a_1 x + a_0\right), a_i \in GF(3)$,
$b(x) = \left(b_{k-1}x^{k-1} + b_{k-2}x^{k-2} + ... + b_1 x + b_0\right), b_i \in GF(3)$,
$n(x) = \left(n_k x^k + n_{k-1}x^{k-1} + ... + n_1 x + n_0\right), n_i \in GF(3)$ ve
$(\alpha)_\beta^{-1}$ gösterilimi α nın Z_β ya göre tersi olmak üzere
$-n(x)'.n(x) \bmod 3 = 1$ yani $n(x)' \bmod 3 = (-n)_3^{-1} = (3 - n_0)_3^{-1}$
$derece(x) < derece(n)$, $derece(y) < derece(n)$, $r(x) = x^k$,
$obeb(n(x), r(x)) = 1$

Çıkış : $a(x)b(x)x^{-k} \bmod(n(x))$

1: $c(x) \leftarrow 0$

2: **for** $i = 0$ **to** $k - 1$ **do**

3: $u_i \leftarrow (c_0(x) + a_i b_0) n(x)' \bmod 3$

4: $c(x) \leftarrow c(x) + a_i . b(x) + u_i n(x)$

5: $c(x) \leftarrow c(x) / x$

6: **end for**

7: **Return** $c(x)$

Algoritmada verilen u_i hesabı aslında $a(x)b(x)$ çarpımının son basamağının değerini ara değerler oluştukça kontrol etmek için kullanılmaktadır. Montgomery algoritması aslında indirgeme işlemini sona bırakmadan ara değerler elde edildikçe indirgeme yapılmasına dayanmaktadır. Burada ara çarpmalar elde edildikçe ara değerin son basamağı, yani $a_i b_0 + c(x) \bmod 3$ değeri kontrol edilir. Eğer ara değerin son basamağı $c_0 = (00)_2$ ise x'e bölme işlemi yani sağa kaydırma işlemi direk gerçekleştirilebilir. Böylece ara değerler hesaplandıkça indirgeme işlemi yapılmış olur. Bu yaklaşım algoritmada u_i hesabı ile kontrol edilmektedir. Eğer $a_i b_0$ çarpımı ve döngünün bir önceki turundan gelen $c(x)$'in en düşük anlamlı basamağının toplamı sıfıra eşit ise, $a_i b_0 + c(x) \bmod 3 = 0$ ise, $u_i = (00)_2$ olur ve algoritmada 4. adımda $(00)_2$ olarak işleme girer. $n(x)$ indirgeme polinomu ile çarpımı $(00)_2$ olur. Dolayısıyla 4. adımdaki ara değer herhangi bir değerle toplanmadan kaydırma işlemi direk yapılır.

Eğer ara değer yani, $a_i b_0$ çarpımı ve döngünün bir önceki turundan gelen $c(x)$'in en düşük anlamlı basamağının toplamı $(01)_2$ ise, $a_i b_0 + c(x) \bmod 3 = (01)_2$ ise, $(10)_2$ değeri eklenmelidir ki son basamak modülo 3 e göre sıfır olsun ve indirgenebilsin. Ancak sonucu etkilemeyecek bir değer eklenmelidir. Bu da indirgeme polinomu $n(x)$ dir çünkü bütün işlemler zaten modülo $n(x)$ e göre yapılır. $n(x)$ indirgeme polinomundan elde edilen $n(x)'$ değeri $n(x)$'in kaç kere ekleneceğini belirler. Seçilen $n(x)$ polinomunun en düşük anlamlı basamağının $(01)_2$ değeri için $n(x)'$ değeri $n(x)' \bmod 3 = (-n)_3^{-1} = (3 - n_0)_3^{-1}$ eşitliğinden $(10)_2$ olarak hesaplanır. Ara değerin en düşük anlamlı basamağının $(01)_2$ değeri için algoritmanın 3. adımında $u_i = (10)_2$ hesaplanır. Algoritmada 4. adıma $(10)_2$ olarak girer. Bu adımda $n(x)$ ile çarpılır. Yani indirgeme polinomu iki kere eklenecektir. $n(x)$'in en düşük anlamlı basamağı $(01)_2$ olduğundan 4. adımda $u_i n(x)$ çarpımı $(10)_2$ olur ve istenen değer eklenmiştir. Artık ara değerin son basamağı $(00)_2$ dır ve sağakaydırma işlemi yani indirgeme gerçekleşebilir.

Aynı yaklaşımla eğer $c_0 = 10$ ise $c(x)$ ara toplamına 01 eklenmelidir ve bu da yine 3. adımda hesaplanır. $u_i = (01)_2$ bulunur ve bir alt adımda indirgeme polinomu sadece 1 kere eklenerek son basamak indirgenebilecek duruma gelir.

4.8. Modüler Çarpmaya Göre Ters Alma İşlemi

Modüler çarpmaya göre ters alma işlemi, Fermat Teoremine göre yapılabilir. [9][34] Bu teoreme göre, $obeb(a, p) = 1$ olmak üzere;

$$a^{-1} = a^{(p-2)} \bmod p \qquad \textbf{(4.55)}$$

Bu teoremi eliptik eğrilerin çalışıldığı sonlu alanlar için ifade edersek Tablo 4-1 elde edilir.

Tablo 4-1 : Sonlu Alanlar ve Modüler Çarpmaya Göre Tersleri

Sonlu Alan	F(x)	Fermat Teoremine göre Evriği
$GF(p)$	p	$a^{-1} = a^{(p-2)} \mod F(x)$
$GF(2^m)$	$p_m x^m + p_{m-1} x^{m-1} + ... + p_1 x + p_0 , p_i \in \{0,1\}$	$a^{-1} = a^{(2^m-2)} \mod F(x)$
$GF(p^m)$	$p_m x^m + p_{m-1} x^{m-1} + ... + p_1 x + p_0 , p_i \in \{0,1,..., p-1\}$	$a^{-1} = a^{(p^m-2)} \mod F(x)$

Çarpmaya göre ters alma işlemi sırasında ihtiyaç duyulan $a^{(p-2)}, a^{(2^m-2)}, a^{(p^m-2)}$ değerlerini hesaplamak için $(p-2), 2^m-2, p^m-2$ değerleri hesaplanıp ikili tabanda gösterilimi oluşturulur, daha sonra üs alma işlemi için kareal-ve-çarp algoritması [9] uygulanır.

Algoritma 4-5 : Kareal-ve-Çarp Algoritması

Girişler: $0 < c < n$ aralığında c tamsayı olup, $c_i \in \{0,1\}$ olmak üzere ikili gösterilimi $c = \sum_{i=0}^{l-1} c_i 2^i$ ve n modülo değeri

Çıkış: $z = x^c \mod n$

1: $z = 1$
2: **for** i : $(l-1)$ **downto** 0
3: $z = z^2 Mod n$
4: **if** $c_i = 1$
5: $z = (z * x) Mod n$
6: **end if**
7: **end for**
8: **return** z

4.9. Eliptik Eğriler ile Veri Şifreleme ve Şifre Çözme

Daha önceki bölümlerde açık anahtar kriptografisinde kullanılan matematiksel temellerden ve eliptik eğri üzerinde yapılan işlemlerden bahsedilmiştir. Bu bölümde eliptik eğrilerin kriptografide nasıl kullanıldığı açıklanacaktır.

İlk olarak bir noktanın ve şifrelenmek istenen verinin eğri üzerindeki noktalarla nasıl eşleştirileceği ve bu eşleştirilme sırasında kullanılan karekök çözümü [35], karesel rezidü [8] kavramları açıklanacaktır.

Eliptik eğrinin denklemi, $y^2 = x^3 + a_4 x + a_6$ olmak üzere, sağ taraf için $f(x)$ dersek kısaca,

$$y^2 = f(x) \tag{4.56}$$

şeklinde bir eşitlik oluşur. Bu eşitlik karesel bir eşitliktir. Sonlu bir alan üzerinde tanımlı eliptik eğri için bu eşitliğin çözümü Cohen tarafından [35]'de önerilmiştir. Bu denklemin çözümü için önce karesel rezidü kavramını açıklamak gerekmektedir.

Tanım 4-3 : Karesel Rezidü

p , tek ve asal sayı olmak üzere,

$$x^2 = a(\text{mod } p) \tag{4.57}$$

Eşitliği için üç durum söz konusudur.

Denklemin çözümü yoktur : a , modülo p 'de karesel rezidü değildir.

Denklemin sadece bir çözümü vardır : $a \equiv 0(\text{mod } p)$ dir.

Denklemin iki çözümü vardır : a modülo p 'de karesel rezidüdür.

Bir a tamsayısının modülo p 'de karesel rezidü olup olmadığını sınamak için a sayısının Legendre sembolü [34] değerine bakılır;

Tanım 4-4 : Legendre Sembolü

a , bir tamsayısı ve $p > 2$ olan bir asal sayı olmak üzere, Legendre sembolü (a / p) değeri aşağıdaki gibi ifade edilir;

$$\left(\frac{a}{p}\right) = \begin{cases} 0, eger \quad p|a \\ 1, eger\ a \bmod p\ de\ quadratik\ rezidü\ ise \\ -1, eger\ a \bmod p\ de\ quadratik\ nonrezidü\ ise \end{cases} \tag{4.58}$$

Yukarıdaki tanımdan elde edilen değer kullanılarak modülo p 'deki çözüm kümesi eleman sayısı $1+(a / p)$ olur.

Ayrıca Fermat teoremine göre [9], $GF(p)$ de $a^{(p-1)/2}$ gibi ifade edilebilen bir değerin karesi 1 dir. Dolayısıyla $a^{(p-1)/2} = \pm 1$ olur.

Bu tanımlarla beraber $GF(p^m)$ tanım kümesinde herhangi bir k tamsayı değeri için, $m = 2k+1$ olmak üzere,şu teorem herzaman geçerlidir; [34];

$$\left(\frac{a}{p}\right) = a^{(p-1)/2} \pmod{p} \qquad \textbf{(4.59)}$$

Tanım 4-5 : Karekök Eşitliği Çözümü

p'nin tek ve asal, a'nın karesel rezidü olması halinde $x^2 = a \pmod{p}$ eşitliğinin x gibi bir çözümü olduğu [35]' de belirtilmiştir ve aşağıdaki gibi verilmiştir;

$$x = a^{(p^m+1)/4} \pmod{p} \qquad \textbf{(4.60)}$$

İspatı için[35]'den yararlanılabilir.

Eliptik eğriler sonlu alan üzerinde tanımlandıklarından eliptik eğrinin denklemini sağlayamayan alan elemanları mutlaka olacaktır. Bu yüzden gerçek veri uzunluğu ya da sayısından çok daha fazla elemanlı bir alan üzerinde çalışma zorunluluğu olabilir. Aksi taktirde tüm veriler eğri üzerindeki bir noktayla eşleştirilemez.

Tablo 4-2 : $GF(11)$'de $y^2 = f(x)$ İçin Çözümler

	0	**1**	**2**	**3**	***4***	**5**	**6**	***7***	**8**	**9**	**10**
0	0	0	0	0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6	7	8	9	10
2	0	2	4	6	8	10	1	3	5	7	9
3	0	3	6	9	1	4	7	10	2	5	8
4	0	4	8	1	***5***	9	2	6	10	4	7
5	0	5	10	4	9	3	8	2	7	1	6
6	0	6	1	7	2	8	3	9	4	10	5
7	0	7	3	10	6	2	9	***5***	1	8	4
8	0	8	5	2	10	7	4	1	9	6	3
9	0	9	7	5	3	1	10	8	6	4	2
10	0	10	9	8	7	6	5	4	3	2	1

Yukarıda örnek olarak $GF(11)$ sonlu alanı için $y^2 = f(x)$ eşitliğinin çözümü verilmiştir.

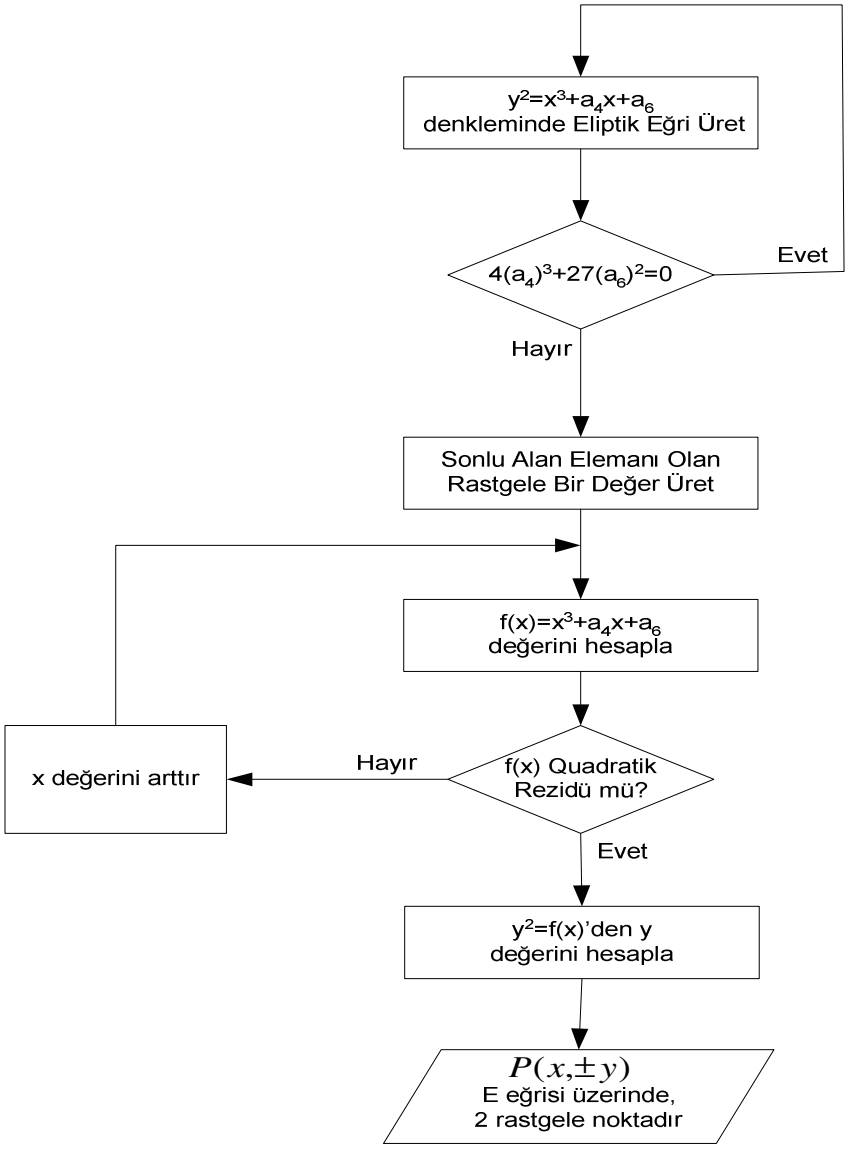
Şifrelenmesi gereken tüm veriyi eğri üzerinde bir noktaya eşleştirebilmek için, eliptik eğrinin $y^2 = f(x)$ denklemi ve çalışılan sonlu alanın karakteristiğinin önemi aşağıdaki tabloyla daha iyi anlaşılabilir.

Tablo 4-3 : $GF(11)$ 'de Tanımlı $y^2 = x^3 + x + 6 \mod 11$ Eğrisindeki Noktalar

X	y^2	$y_{1,2}$	P(x,y)	P'(x,y)
0	6	-		
1	8	-		
2	5	4,7	(2,4)	(2,7)
3	3	5,6	(3,5)	(3,6)
4	8	-		
5	4	2,9	(5,2)	(5,9)
6	8	-		
7	4	2,9	(7,2)	(7,9)
8	9	3,8	(8,3)	(8,8)
9	7	-		
10	6	2,9	(10,2)	(10,9)

Yukarıdaki tabloda görüldüğü gibi $y^2 = x^3 + x + 6 \mod 11$ eğrisi üzerinde toplam 12 adet nokta oluşturulabilir. Sonsuz noktasıyla beraber eliptik eğri üzerindeki 13 nokta bir grup oluşturmaktadır. Buradaki $n = 13$ nokta sayısı eğri grubunun derecesidir ve eğriyi oluştururken seçilen $a, b \in GF(p)$ değerlerine bağlıdır.

Eliptik eğriler üzerine verileri eşleştirmek için birçok yol olmakla birlikte en yaygın yöntem Koblitz tarafından [34]' de önerilen tekniktir. Bu teknikte veri, eğri üzerine yerleştirilmeden $y^2 = f(x)$ denklemini sağlayıp sağlamadığı kontrol edilir. Eğri üzerine veri yerleştirme işlemi ilgili akış diyagramı Şekil 4.4'de verilmiştir.



Şekil 4.4 : Rastgele Bir Noktanın Eliptik Eğri Üzerine Yerleştirilmesi

4.9.1. Açık Metin Bilgiyi Eliptik Eğri Üzerine Yerleştirme

Şifrelenmek istenen açık metin bilgisini eliptik eğri üzerindeki noktalarla eşleştirmek ile rastgele bir noktayı eliptik eğri üzerine yerleştirme arasında farklılıklar vardır. Eğri üzerine yerleştirilen açık metin başarıyla tekrar elde edilmelidir dolayısıyla her veri eğri üzerinde sadece bir noktayla eşleşmeli ve aynı şekilde çözüldüğünde her nokta sadece bir sonuç oluşturmalıdır.

Bir açık metni eliptik eğri üzerine yerleştirmek için;

T açık metni, l eşit uzunluklu t parçaya ayrılır.

Her parça, sonlu alanın bir elemanı olacak şekilde eşleştirilir. $(t \to x_i)$ $x_i = SonluAlanElemani$

Burada her blok uzunluğu l seçilebildiği kadar büyük seçilmelidir ki, $t \to x_i$ dönüşümü sonrasında aşağıdaki eşitlik oluşturulabilsin;

$$t_{m-2}x^{m-2} + t_{m-3}x^{m-3} + ... + t_1 x + t_0 \quad \textbf{(4.61)}$$

$j = t_{m-1}$ olmak üzere;

$$x_t = jx^{m-1} + \sum_{i=0}^{m-2} t_i x^i \quad \textbf{(4.62)}$$

Daha önceden verilen $f(x) = x^3 + a_4 x + a_6$ eşitliğindeki sağ taraf değeri, x_t için hesaplanıp karesel rezidü olup olmadığı Legendre Sembolü yöntemi ile kontrol edillir.

Eğer karesel rezidü ise $f(x)$'in karekökü olan y_t , $x = a^{(p^m+1)/4} (\text{mod}\, p)$ eşitliği ile hesaplanır.

$P_t(x_t, y_t)$ noktası, t ile gösterilen gömülü mesaj bloğunu ifade eder.

Eğer $f(x)$, karesel rezidü değil ise, j değeri bir artırılır ve yeni x_t değeri için tekrar denenir. j değerinin en fazla t_{m-1} değerine ulaşana kadar $f(x)$'in kare olduğu x_t değerinin varolduğu [34]' de Koblitz tarafından ispatlanmıştır.

Aynı şekilde, $P_t(x_t, y_t)$ noktasından t değerini yeniden elde etmek de mümkündür. Yapılması gereken sadece x_t'nin son terimini olan t_{m-1} değerini eleyip x_t'yi t olarak ifade edilen diziye çevirmektir.

4.9.2. Eğri Üzerinde Şifreleme / Şifre Çözme

Şifrelenmek istenen veri, eliptik eğrinin x koordinatlarıyla eşleştirilir ve buna karşı gelen y koordinatlarıyla beraber Q_m şeklinde mesaj noktaları oluşturur. Seçilen bu noktanın eğri üzerinde olması (x, y) koordinatlarının sağlaması yapılarak sağlanır.

Daha önceden paylaşılan anahtar k değeri ve eğri üzerindeki P taban noktası çarpılarak kP koordinatları oluşturulur. Bu çarpım değeri ve şifrelenmek istenen verinin eğri üzerine yerleştirilmiş hali olan Q_m değeri toplanarak $Q_c = Q_m + kP$ hesaplanır. Bu oluşan yeni noktanın sadece x koordinatı olan x_c karşı tarafa yollanır.

Alıcı veriyi şifreleyen taraftan sadece x_c bilgisini almıştır. Açık anahtar şifrelemesi için gerekli daha önceden belirlenen k ve P değerleri de kendisinde mevcuttur. İlk olarak kP değerini hesaplar. Kendisine ulaşan verinin açık halinin eğri üzerine düşen noktası $Q_m = Q_c - kP$ şeklinde hesaplanır ve gizli veri Q_m'nin bit dizilimlerinden tekrar veri haline dönüştürülmesiyle elde edilir.

5. ELİPTİK EĞRİ UYGULAMALARI

Bu bölümde eliptik eğrilerin kriptografide nasıl ve hangi amaçlarla kullanıldığı konusunda bilgi verilecektir.

Tüm açık anahtar kriptosistemlerinde genel prensip biri gizli biri açık olmak üzere iki anahtar olmasıdır. Uygulamaya bağlı olarak gönderici veriyi kimi zaman kendi özel anahtarı ile kimi zaman alıcının açık anahtarı ile işleme sokar. Bazı durumlarda her iki anahtarın da kullanılması gerekebilir. Genel olarak açık anahtarlı kriptosistemler şu üç ana başlık altında incelenir; [28]

Şifreleme /Şifre Çözme : Gönderici veriyi alıcının açık anahtarı ile şifreler.

Sayısal İmza Uygulaması : Gönderici özel anahtar ile veriye ait imza üretir. Bu imza verinin tamamı ya da önemli bir kısmı için yapılabilir.

Anahtar Değişimi : Gönderici ile alıcı haberleşme için kurulan oturumun anahtarlarını değiş tokuş ederler.

5.1. Neden Eliptik Eğri?

Bu konuda bir çok araştırma yapılmıştır ve hepsinin ortak noktası eliptik eğrinin aynı anahtar uzunluğu için RSA'e göre daha güçlü bir koruma sağladığıdır.[6] ve [36]'da EEK avantajlarını şöyle sıralamıştır;

- Aynı anahtar uzunluğu için daha fazla güvenlik sağlar.
- İstenen güvenlik seviyesi için daha kısa anahtar uzunluğu daha küçük ve hızlı kriptografik işlem yapabilen uygulama imkanı sağlar. Aynı zamanda devre daha küçük alanda gerçeklenebilir. Bu da daha az güç tüketimi ve daha az ısı üretimi demektir.
- Kırmık içinde gerçeklenen EEK uygulaması daha az alan kaplar.
- EEK kriptosistemlerinin yazılım uygulamalarında daha az bellek tüketimi oluşur. Akıllı kart gibi sınırlı belleğe sahip yapılar için bu özellik önemlidir.
- Sadece eliptik eğri parametreleri değiştirilerek güvenlik yeniden sağlanabilir.

5.2. Eliptik Eğri Parametreleri

Bir EEK için parametreler şunlardır; [2]

q: bir asal sayının üssüdür. Eğri için $GF(q)$, tanımlı olduğu alanı belirtir. $q = p^m$ şeklindedir.

Alan gösterilim metodu: Alan elemanlarının gösteriliminde uygulanan metodu belirtir.

a_4, a_6: $y^2 = x^3 + a_4 x + a_6$ eliptik eğri denklemini belirleyen katsayılardır.

G: (x_g, y_g) şeklinde ifade edilen $E(GF(p^m))$ eğrisi üzerndeki taban noktasıdır.

n: G noktasının derecesidir yani $n, nG = 0$ eşitiğini sağlayan en küçük tamsayıdır.

Diğer tüm kriptosistemlerde olduğu gibi bu sistemde de haberleşen uçlar vardır ve her ucun kendine ait bir adet gizli, bir adet açık anahtarı bulunmaktadır. Eliptik eğri kriptografisi kullanarak her uç kendi anahtarlarını şu şekilde üretir;

Uygun bir sonlu alan $GF(p^m)$ seçilir.

Seçilen sonlu alan üzerinde rastgele bir E eliptik eğri üretilir.

Seçilen sonlu alan içinde rastgele bir sonlu alan elemanı k seçilir yani $k \in GF(p^m)$

E eliptik eğrisi üzerinde bir G baz noktası üretilir.

$P = kG$ eşitliği ile $P \neq$ Ö olacak şekilde eğri üzerindeki P noktası hesaplanır.

k ,gizli anahtar, P ise açık anahtardır.

5.3. Eliptik Eğri Ayrık Logaritma Problemi

Ayrık logaritma problemi (ALP) kısaca şöyle açıklanabilir[8]; Verilen bir $a^i = b$ eşitliği için $i = \log_a b$ eşitliğini sağlayan i değerinin bulunmasına dayanır. ALP çözümünün sonlu alanlar grubunda çözümü şöyle tanımlanır;

Tanım 5-1 : Sonlu Alanda Ayrık LogaritmaProblemi[34]

G sonlu alan grubu, $b \in G$, $y \in G$ ve b'nin bir kuvveti olmak üzere, y'nin b tabanında ayrık logaritması x gibi bir tamsayıdır yani $b^x = y$ şeklindedir[34].

EEK grup logaritma problemini kullanır.[34] Eliptik eğriler üzerindeki noktaların oluşturduğu grup üzerinde ALP tanımı, eliptik eğri ayrık logaritma problemi aşağıda verilmiştir.

Tanım 5-2 : Eliptik Eğri Ayrık Logaritma Problemi [34]

E, $GF(q)$ üzerinde bir eliptik eğri; G, E üzerinde bir nokta olmak üzere, E üzerinde ayrık logaritma problemi; verilen bir $P \in E$ noktası için $P = kG$ eşitliğini sağlayan öyle bir $k \in Z$ bulma problemidir. [34]

k değerini bulmak için $G, 2G, 3G, ...$ şeklinde tekrarlayarak kG değeri deneme yoluyla bulunabilir ancak k değerinin yeterince büyük seçilmesi durumunda bu yöntemle sonucun bulunması günümüz işlemcileriyle yüzlerce yıl süreceği hesaplanmıştır. [36]

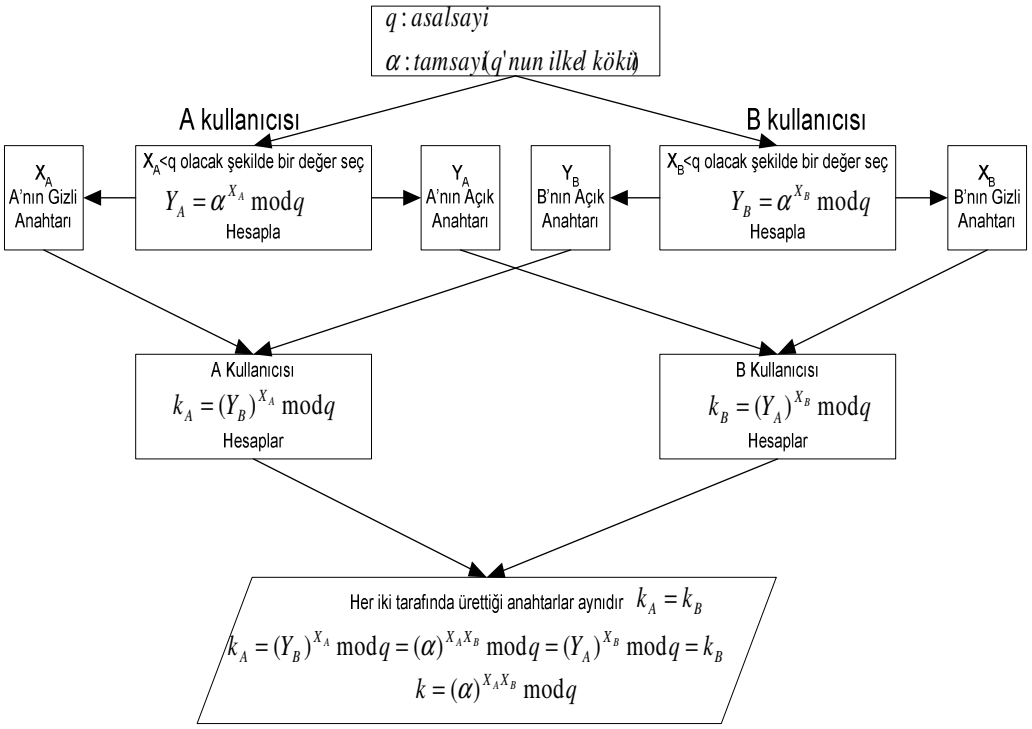
ALP' ye dayanan önemli açık anahtar kriptosistemlerinden birisi de ElGamal Kriptosistemidir.[17] ElGamal kriptosistemi kullanılarak anahtar üretimi Diffie-Hellman anahtar değişim protokü ile benzerlik göstermektedir. Bu anahtar üretiminden sonra ElGamal kriptosisteminde şifreleme ve şifre çözme işlemleri gerçekleştirilebilir.

5.4. Diffie-Hellman Anahtar Değişimi

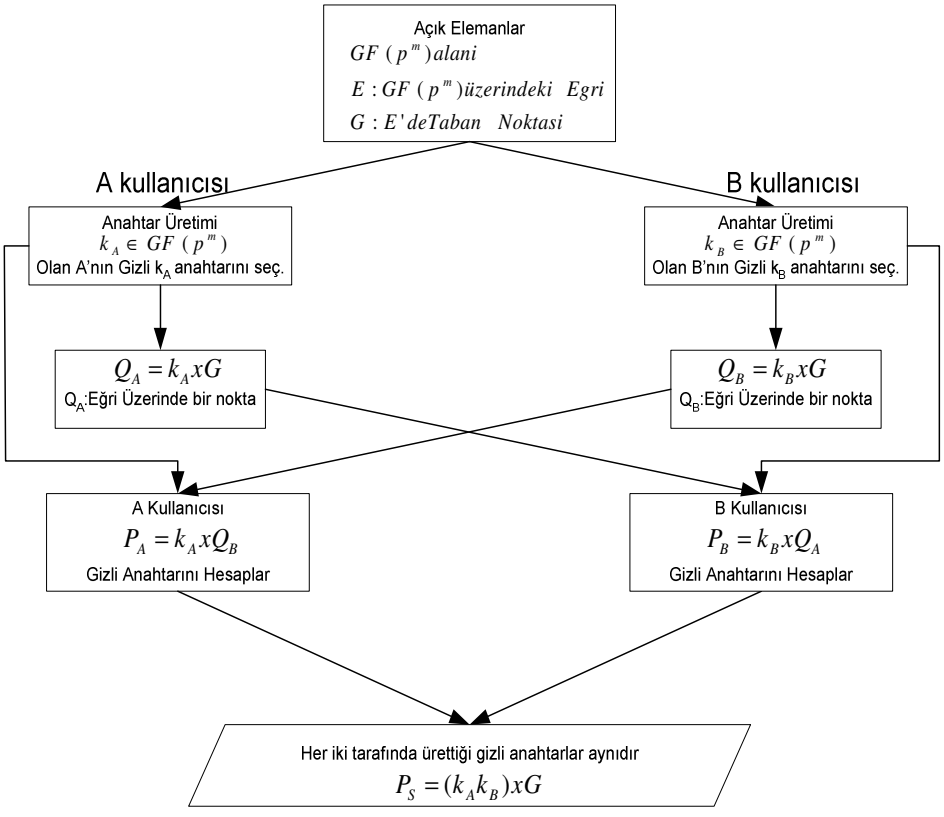
Diffie-Hellman protokolü[14], uçlar arasında gizli anahtar paylaşımı için tasarlanmış bir açık anahtar kriptosistemidir. İlk açık anahtar kriptosistemidir. Güvenliği sonlu alanda ALP'nin çözümünün zorluğundan gelir. Diffie-Hellman algoritmasının amacı güvensiz kanal üzerinde özel bilgiler oluşturup bunları haberleşen uçlar arasında güvenli şekilde paylaştırmaktır.

Diffie-Hellman anahtar değişimi protokolü Şekil 5.1' de verilen akış diyagramı ile özetlenmiştir.

Şekil 5.1' de genel akışı verilen Diffie-Hellman anahtar değişim protokolü için temel özellik ayrık logaritma hesabının zorluğudur.



Şekil 5.1 : Diffie-Hellman Anahtar Değişim Protokolü



Şekil 5.2 : Eliptik Eğri Diffie-Hellman Anahtar Üretim Protokolü

Verilen Diffie-Hellman anahtar değişim protokolü, eliptik eğriler üzerinde de uygulanabilmektedir. Bu uygulama sayesinde eliptik eğriler kullanılarak birbirinden bağımsız iki kullanıcı arasında anahtar paylaşımı gerçekleşir. Eliptik eğri Diffie-Hellman anahtar üretim protokolü Şekil 5.2' de verilen akışla gerçekleşir;

Bu akış sonunda oluşan $P_S = (x_S, y_S)$ noktası, E eğrisi üzerinde bir noktadır. y_S değerini kullanmaya gerek yoktur çünkü eğri denklemi ve x_S değerinden üretilebilmektedir. Ancak y_S'e ait sadece belirli bir bite ihtiyaç vardır bunun nedeni de denklemden elde edilen y_S ya da $-y_S$ hangi kökün kullanılacağıdır.

Eliptik eğri Diffie-Hellman protokolünün güvenliği, $P_S = (x_S, y_S)$ noktasını elde edilebilmesi için eliptik eğri Diffie-Hellman probleminin çözümünün zorluğuna dayanmaktadır.

Tanım 5-3 : Eliptik Eğri Diffie-Hellman Problemi
Verilen $G, Q \in E$ noktaları için, k sonlu elemanı olmak üzere, $Q = k.G$ eşitliği olsun. eliptik eğri Diffie-Hellman problemi bu eşitlikteki k değerini bulmaya dayanır.

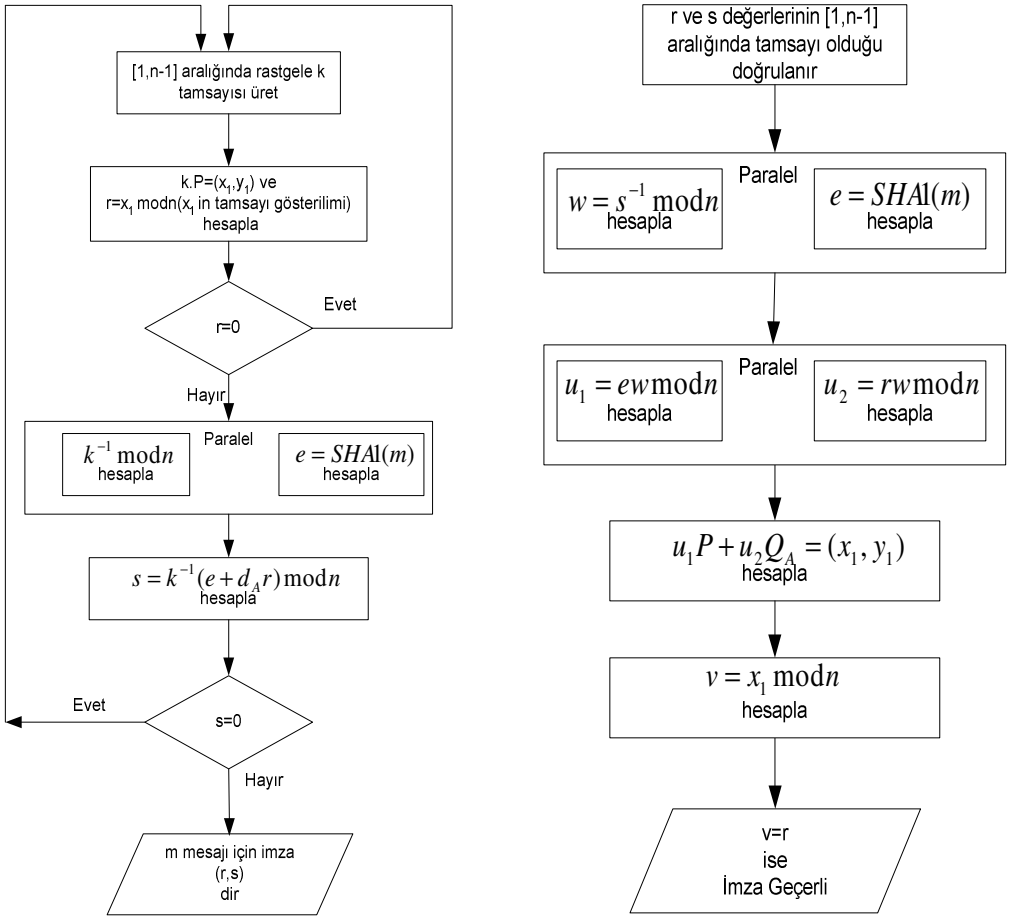
Sadece $G, k_A G, k_B G$ bilinerek P_S, k_A ve k_B hesaplanamaz. Eliptik eğri Diffie-Hellman fonksiyonları bu anlamda tek yönlü fonksiyonlardır.[14]

5.5. Eliptik Eğri Sayısal İmza Algoritması

Sayısal imzalar özellikle internet haberleşmesinde çok kullanılmaktadır.[10] Şifreli olarak gönderilen verinin sonuna gönderen kendi gizli anahtarı ile bir imza ekler. Alıcı gelen şifreli veriyi göndericinin açık anahtarı ile imza kontrolü yaparak alır. Böylece gönderici tanınmış olur. Temelde gelen veri kaynağının güvenliği ve inkar edememe servisleri için kullanılır.

Eliptik eğri sayısal imza uygulaması da aynı amaçla kullanılır. Temelde DSA' nın eliptik eğriler üzerine uygulaması olmakla birlikte çok daha fazla güvenlik sağlamaktadır.

Eliptik eğri imzalama ve imza onaylama işlemleri Şekil 5.3 deki gibi olmaktadır;



a) İmzalama **b)** İmza Onaylama

Şekil 5.3 : Eliptik Eğri a) Sayısal İmzalama ve b) İmza Onaylama Algoritması

6. ELİPTİK EĞRİ KRİPTOSİSTEMİNİN GERÇEKLENMESİ

Eliptik eğri kriptosisteminin gerçeklenmesi öncesinde tasarım gereksinimleri belirlenmelidir. Bu gereksinimler açık ve net bir şekilde belirlenerek üst seviye tasarımı gerçeklenir. Bir eliptik eğri kriptosistemi için belli başlı parametreler şu şekilde sıralanabilir;

- Eliptik eğrinin üzerinde tanımlanacağı sonlu alan
- Eliptik eğri seçimi
- Eğri üzerinde tanımlanan ve haberleşecek uçlar tarafından paylaşılan P taban noktası.
- Tanımlı olduğu sonlu alanda modülo işlemi için kullanılacak indirgenemez polinom
- Eliptik eğrinin üzerinde tanımlı olduğu sonlu alanda modüler çarpma ve modüler toplama aritmetik işlemleri için yöntemler. Örneğin modüler çarpma için MMÇ yöntemi ya da bunun yerine ikili bir çarpma devresiyle birlikte ayrı bir modüler indirgeme bloğu beraber kullanılabilir.
- Alan işlemleri için özel yöntemler varsa bu yöntemlerin seçimi

EEK için şifreleme ve şifre çözme işlemleri daha önce Bölüm 4.9.2 de bahsedildiği gibi sonlu bir alanda tanımlı bir eliptik eğri üzerinde nokta çarpma ve nokta toplama işlemleri ile gerçekleşmektedir. Şifreleme sırasında yapılan $Q_c = Q_m + kP$ ve şifre çözme sırasında yapılan $Q_m = Q_c - kP$ işlemlerindeki kP nokta çarpma işlemi için Bölüm 4.5 de verilen ikile-ve-topla algoritması kullanılır. İkile-ve-topla algoritmasında görüldüğü gibi k sayısı ikili tabanda ifade edilir. k ile çarpılacak nokta algoritmada döngüye her girişte iki ile çarpılır ve k'nın döngü sayısı indeksli bit değeri 1 ise çarpılacak nokta ile daha önceden iki ile çarpılarak elde edilen ara değer toplanır. Yani döngünün her turunda bir ikileme işlemi vardır ve yine her turda k' nın bit değerinin 1 olması durumunda bir nokta toplama işlemi yapılır.

Bahsedilen algoritmada görüleceği gibi eliptik eğri şifrelemesi nokta ikileme ve nokta toplama işlemlerine dayanır. Daha önce Bölüm 3 ve Bölüm 4' de bahsedildiği üzere, eliptik eğriler sonlu alanlar üzerinde tanımlanarak kriptografide kullanılmaktadır. Dolayısıyla bahsedilen bu iki gereksinim birleştirilirse, nokta ikileme ve nokta toplama işlemlerini sonlu alanlar üzerinde gerçekleyerek EEK gerçeklenebilir.

Bu çalışmada kullanılan $GF(3^m)$ sonlu alanı için ilgin koordinat sisteminde nokta ikileme ve nokta toplama işlemleri için eşitlikler Bölüm 4.3 de verilmiştir. Bu eşitlikler kullanılarak nokta ikileme ve nokta toplama işlemleri gerçeklenip eliptik eğri kirptosistemi oluşturulabilir. Ancak eşitliklerde görüldüğü gibi işlem sonucu nokta koordinatları (x_3, y_3) elde etmek için çarpmaya göre ters alma işlemi bulunmaktadır. Çarpmaya göre ters alma işlemi, gerçekleme esnasında fazla işlem içeren ve uzun süren bir uygulamadır. Fikir vermesi açısından bir karşılaştırma yapılırsa $GF(p)$'de 100-bitle ifade edilen bir sayı için bir çarpmaya göre ters alma işlemi 30 çarpma işlemine denk düşmektedir. Bu işlem sayısı fazlalığından kurtulmak için çarpmaya göre ters alma işlemi olmayan onun yerine daha fazla çarpma işlemi ile aynı sonuca ulaşılabilecek başka bir koordinat sistemine, projektif koordinatlara geçilmesi fikri bölüm 4.4' de açıklanmıştır.

Bu çalışmada kullanılan $GF(3^m)$ sonlu alanı için bölüm 4.3 de verilen ilgin koordinat sisteminde nokta çarpma ve nokta ikileme işlemlerinin projektif koordinat sistemindeki karşılıkları bölüm 4.4.1 de verilmiştir. Bu iki farklı koordinat sistemindeki işlemler karşılaştırıldığında ilgin koordinat sisteminde bir nokta toplama işlemi, 1 çarpmaya göre ters alma, 2 çarpma ve 6 toplama işlemi kadardır. Aynı işlemin projektif kooridnatlardaki karşılığı ise 16 çarpma işlemi ve 6 toplama işlem yüküdür. Bir çarpmaya göre ters alma işlem yükünün yaklaşık 30 çarpmaya karşılık geldiği [15] düşünülürse işlem maliyetinde ciddi bir azalma elde edilmektedir. Aynı şekilde bir karşılaştırma nokta ikileme işlemleri için yapılırsa ilgin koordinat sisteminde nokta ikileme işlemi için 1 çarpmaya göre ters alma, 6 çarpma ve 4 toplama işlemi kadardır. Aynı işlemin projektif koordinatlardaki karşılığı ise sadece 9 çarpma 3 toplamadır. Bu yüzden eliptik eğri için gerekli nokta çarpma ve nokta toplama işlemleri için projektif koordinatlarda çalışılmaya karar verilmiştir.

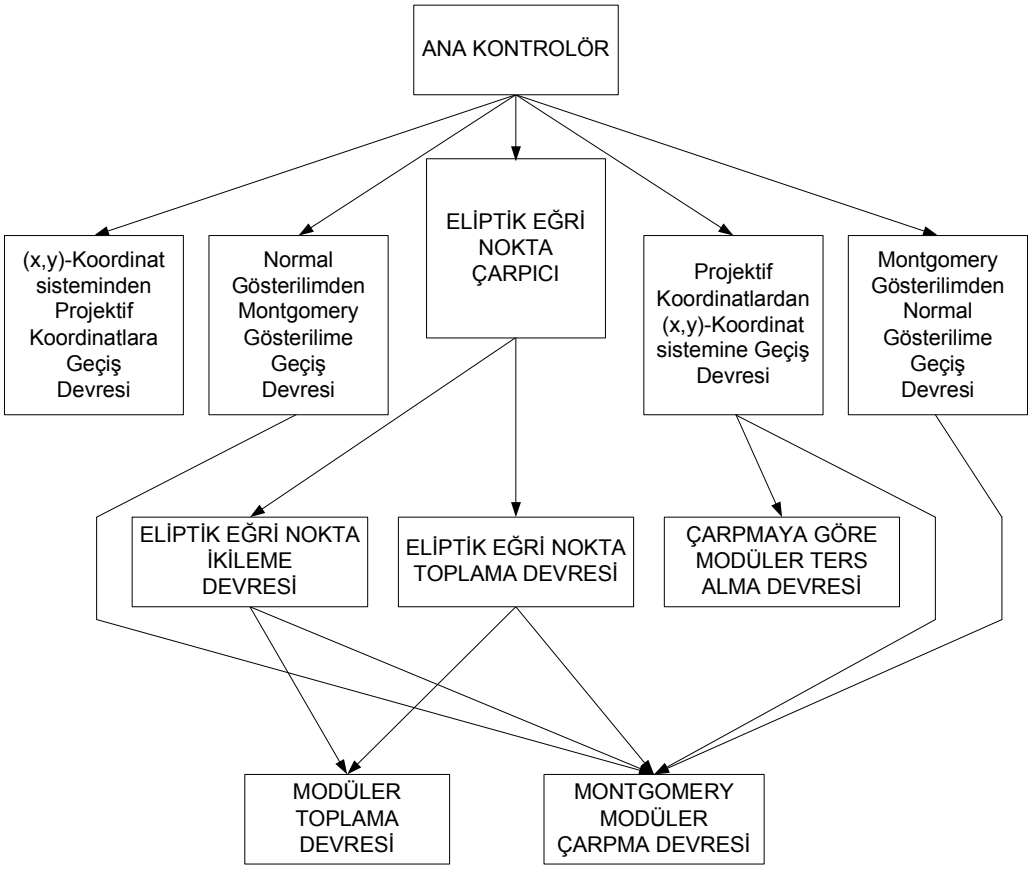
Projektif koordinatlarda çalışmak için ilgin koordinatlardan projektif koordinatlara geçip, işlemleri projektif koordinatlarda gerçekleştirip tekrar geriye ilgin koordinat sistemine dönülmelidir. Projektif koordinatlara geçiş Bölüm 4.4.1 de verilmiştir.

Bölüm 4.4.1 de $GF(3^m)$ için verilen işlemlere bakıldığında modüler çarpma ve modüler toplama işlemleri vardır. Modüler çarpma işlemlerinde çarpma işleminin yanısıra modüler indirgeme işlemi de gerekmektedir. Modüler indirgeme işlem sayısı fazla bir uygulamadır. Çarpma ve modüler indirgeme gibi iki ayrı blok yerine daha önce Bölüm 4.7 de bahsedilen MMÇ devresi sayesinde çarpma ve modüler indirgeme işlemi tek bir blokla daha az işlem yüküyle gerçeklenebilmektedir. Sonuç olarak projektif koordinatlarda gerekli çarpma ve toplama işlemleri için MMÇ devresi ve modüler toplama devresi gerçekleme ihtiyacı vardır.

$GF(3^m)$ sonlu alanı için MMÇ algoritması Bölüm 4.6 da verilmişti. Bu algoritma çıkışında elde edilen değerde algoritma içerisinde kullanılan $r(x)$ parametresinin istenen çarpımla birlikte sonuç değeri içerisinde görüldüğü açıktır. İstenen sonucu elde etmek için MMÇ algoritmasına giriş olarak verilecek parametrelerin Montgomery gösterilimleri kullanılmalıdır (Bölüm 4.6). Bu durumda montgomery çarpma algoritmasına verilen tüm giriş değerleri Montgomery gösterilimde olur ve çıkış da yine Montgomery gösterilim şeklindedir. Algoritma ile yapılacak tüm işlemler Montgomery gösterilimle yapılıp daha sonra tekrar mongomery gösterilimden normal gösterilime geçilerek istenen sonuç elde edilmiş olur.

Şu ana kadar anlatılanlar özetlersek; nokta ikileme ve toplama işlemleri için projektif koorditlara geçiş yapılır ve buradaki eşitlikler uygulanır. Bu eşitliklerde görülen çarpma işlemleri için Montgomery gösterilime geçilerek MMÇ algoritması uygulanır ve çarpma sonuçları bittikten sonra normal gösterilime geçilir. Son olarak da projektif koordinatlardan tekrar ilk başta verilen ilgin koordinatlara geri dönülerek işlem tamamlanır.

Şekil 6.1 eliptik eğri işlemcisinde kullanılan alt bloklar ve bloklar arası ilişki gösterilmektedir. Alt blokların ayrı ayrı tasarımları gerçeklenmelidir. Bir sonraki bölümde bu alt bloklar detaylı biçimde anlatılacaktır.



Şekil 6.1 : Eliptik Eğri İşlemcisinde Kullanılan Blokların Gösterilimi

6.1. Montgomery Modüler Çarpma Devresi Tasarımı

Şekil 6.1 de verildiği üzere eliptik eğri alt bloklarının büyük bir kısmı MMÇ devresini kullanarak kendi işlemlerini gerçeklemektedir. Bu yüzden tasarıma MMÇ devresi tasarımıyla başlanması gerekmektedir. Tasarıma daha sonra Şekil 6.1 de verilen gösterilim şekliyle alttan üste doğru blokları gerçekleyerek devam edilir.

Seçilen $GF(3^m)$ sonlu alanına özgü MMÇ algoritması Algoritma 4-4 de verilmiş ve bölüm 4.7 de detaylı olarak anlatılmıştır.

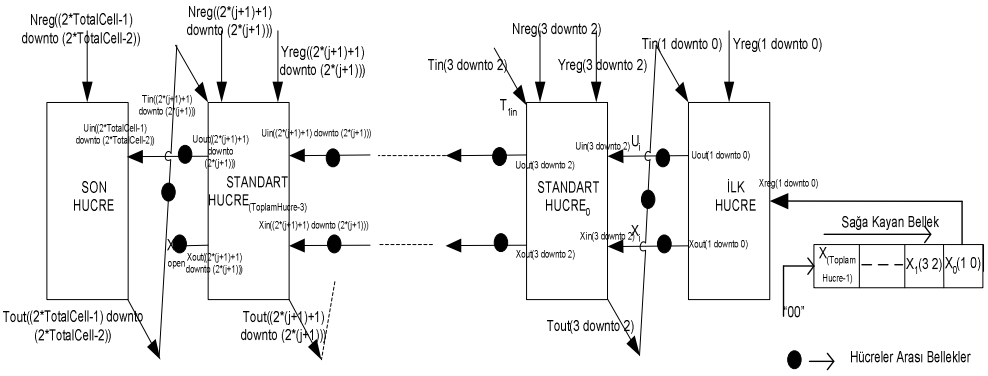
Algoritmanın donanımla gerçeklenmesi için ardışıl hücre dizileri şeklinde bir yapı tasarlanmıştır. Bu yapı sayesinde en uzun yol gecikmesi azaltılmıştır.

Algoritmada $c(x) \leftarrow c(x) + a_i.b(x) + u_i n(x)$ şeklinde ifade edilen ve her döngüde güncellenen değer döngü sayısı tamamlandıktan sonra sonuç değeri vermektedir. Bu istenen değerin her bir basamağı ardışıl hücre dizileri şeklinde ifade edilen yapıda bir

hücre tarafından hesaplanmaktadır. $GF(3^m)$ sonlu alanında çalışıldığı için $c(x)$ değerinin her basamağı $GF(3)$ alanından bir değerle ifade edilir ve daha önce Bölüm 3.3 de detaylı bahsedilen polinomsal gösterilimle ifade edilir. Algoritmada her hücrenin $c(x)$ değerinin bir katsayısını ifade etmesinin yanısıra, 3. adımda görülen u_i değeri hesabı gerekmektedir. Bu değer 4. adımdaki $c(x)$ hesabında giriş olarak kullanılmaktadır. Dolayısıyla $c(x)$ hesaplanmadan önce bir hücre yapısıyla u_i hesaplanmalıdır. Sonuç olarak gerekli hücre sayısı istenen sonuç değerinin basamak sayısının bir fazlası olmaktadır.

Kullanılan hücreler, iç yapıları ve Algoritma 4-4' de hangi işlemi gerçeklediği aşağıda verilmiştir;

Ardışıl hücre dizisinde 3 farklı hücre yapısı kullanılmaktadır. Bunlar İlkHücre, StandartHücre ve SonHücredir.



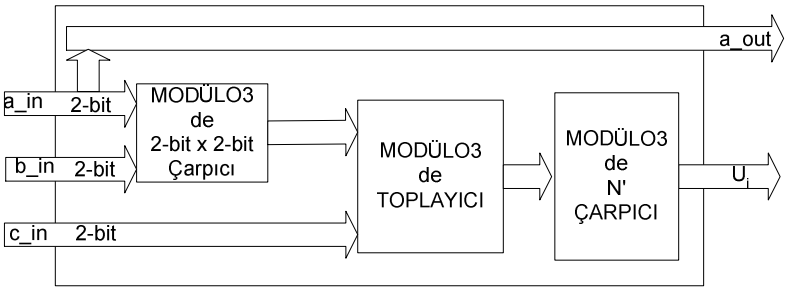
Şekil 6.2 : Ardışıl Hücre Dizileri Şeklinde Tasarlanmış MMÇ Devresi

İlkHücre algoritmada 4. adımda $c(x)$ değerinin hesabı için kullanılan ve 3. adımda oluşturulan u_i ara değerini hesaplamak için kullanılır. $u_i \leftarrow (c_0(x) + a_i b_0) n(x)' \bmod 3$ eşitliği ile verilen değerde $n(x)'$ kullanılmaktadır. Buradaki $n(x) = \left(n_k x^k + n_{k-1} x^{k-1} + ... + n_1 x + n_0\right), \quad n_i \in GF(p)$ algoritmada da belirtildiği gibi indirgenemez polinomdur ve çarpım sonucunu indirgeme için kullanılmaktadır.

İndirgenemez polinomlar seçilen sonlu alana göre belirlenir. Bir polinomun indirgenemez olması için herhangi başka iki polinomun çarpımı şeklinde ifade edilememesi gerekmektedir.[34] Seçilen alan içindeki her polinom indirgenemez polinom olarak kullanılamaz. Bu yüzden indirgenemez polinom daha önceden belirlenmelidir. Bu polinom devreye çalışma esnasında parametre olarak verilebileceği gibi daha önceden ilgili alanda var olan indirgenemez polinomlardan biri seçilerek devre içerisine sabit bir değer olarak da atanabilir. Bu çalışmada $GF(3^m)$ sonlu alanı için olası indirgenemez polinomlar incelenmiş ve $n(x) = x^{97} + 2x^{12} + 1$ seçilmiştir. İndirgenemez polinom seçimi için [22]'den yararlanılabilir.

Algoritma 4-3 ile verilen algoritmanın 3. adımının hesaplanması için gerekli olan $n(x)'$, $-n(x)'.n(x)Mod(3) = 1$ eşitliği ile elde edilir. $n(x)Mod(3) = 1$ olduğundan çarpımın 1 olabilmesi için indirgeme sonucunda $n(x)'$ değerinin -1 olması gerekmeltedir. Bu da $-1 \bmod(3) \equiv 2$ demektir. Yani $u_i \leftarrow (c_0(x) + a_i b_0) n(x)' Mod\, 3 = \quad (c_0(x) + a_i b_0) 2 \bmod 3$ eşitliği elde edilir. $u_i = 2.(c_0(x) + a_i b_0) \bmod 3$ eşitliğini gerçekleyen devre için blok şema Şekil 6.3 de verilmiştir. Bu devre ayrıca dizi hücre yapısında yer alan diğer hücreler için girişine verilen a çarpanını çıkışına aktararır.

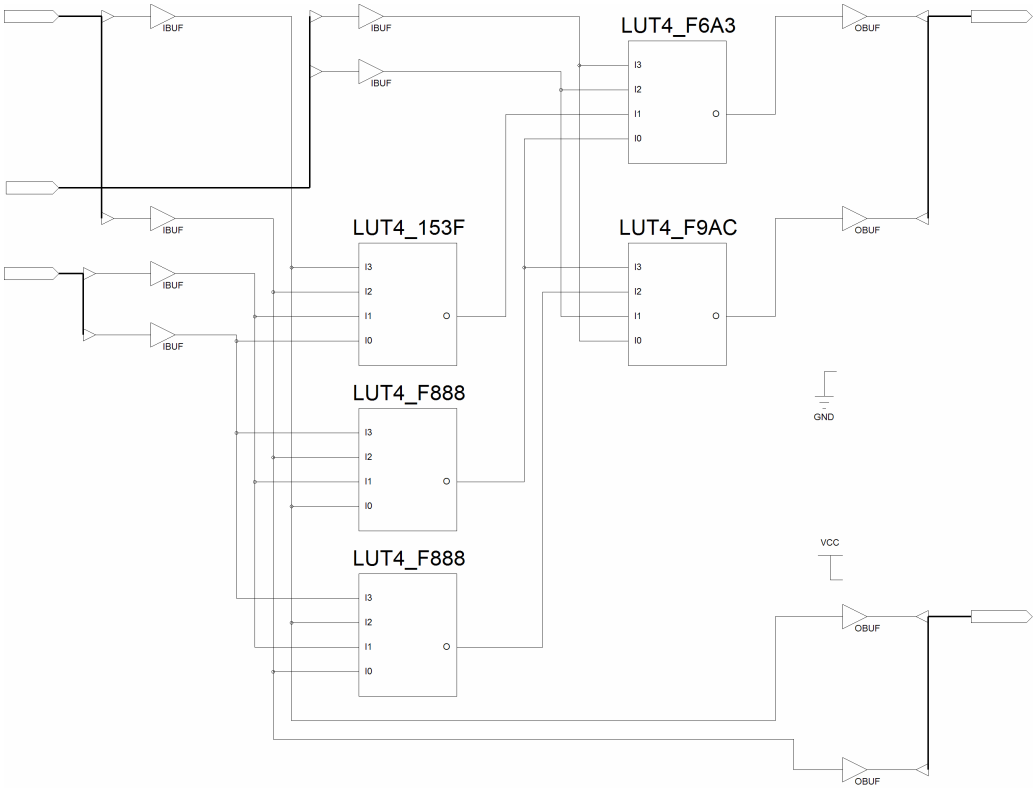
Xilinx firmasının VirtexE ailesi FPGA'leri üzerinde İlkHücre gerçeklemesinde toplam 3 adet dilim ve 5 adet 4 girişli doğruluk tablosu kullanılmıştır.



Şekil 6.3 : İlkHücre Blok Şeması

İlgili blok şemaya ait devre şeması gösterilimi

Şekil 6.4 de verilmektedir.

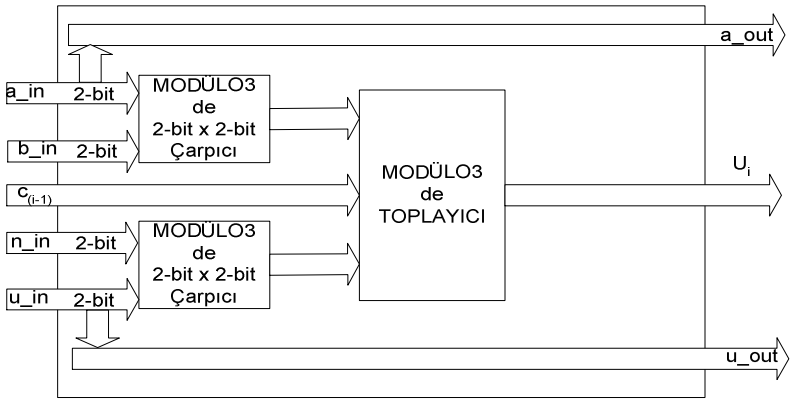


Şekil 6.4 : İlkHücre Devre Şeması

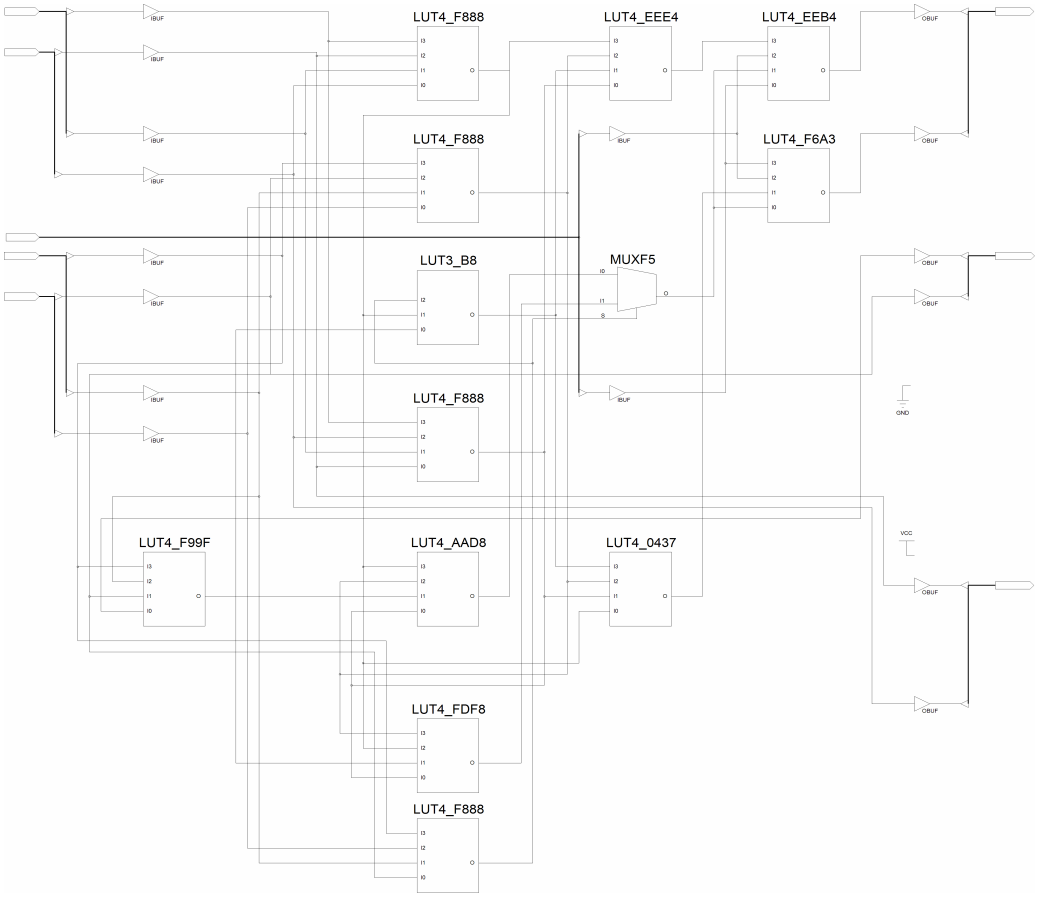
Bu eşitlikteki tüm değerler $GF(3)$'de ifade edilen değerler olduğu için donanım olarak gerçeklemede ikili çalışma için iki bitle gösterilmekte ve işlemlere iki bit olarak verilmektedir. Yani $GF(3)$ de elemanlar $\{0,1,2\}$ olduğundan bu değerler $\{00,01,10\}_2$ olarak ikili gösterilimde ifade edilir. Devrede de iki bitlik birer veri yolu şeklinde bloklar arasında taşınır. Başka bir deyişle $GF(3)$ de veriler iki bitlik veri yolları şeklinde taşınır ve işlenir.

Algoritma 4-3'de u_i olarak ifade edilen terimi İlkHücre ile hesapladıktan sonra, StandartHücrelerle algoritmada bahsedilen $c(x) \leftarrow c(x) + a_i.b(x) + u_i n(x)$ terimi her iterasyon için hesaplanacaktır. Sıralı hücre dizilerinden her biri $c(x)$'in bir basamağını hesaplamaktadır. Yani i numaralı döngü için j numaralı hücre $c_{i,j} = c_{i-1,j+1} + a_i.b_j + u_i n_j \mod(3)$ değerini hesaplar. Eşitliği gerçekleyen devre Şekil 6.5 de verilmiştir. Bu şekilde gösterildiği gibi hücre, a çarpanı ve u değerini kendi

işlemleri için kullanmakla birlikte çıkışında yer alan diğer hücrelere de aktararır. Böylece diğer hücreler de sırayla aynı değer için ilgili sonuç üretir.



Şekil 6.5 : StandartHücre Blok Şeması

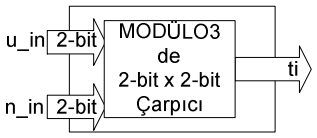


Şekil 6.6 : StandartHücre Devre Şeması

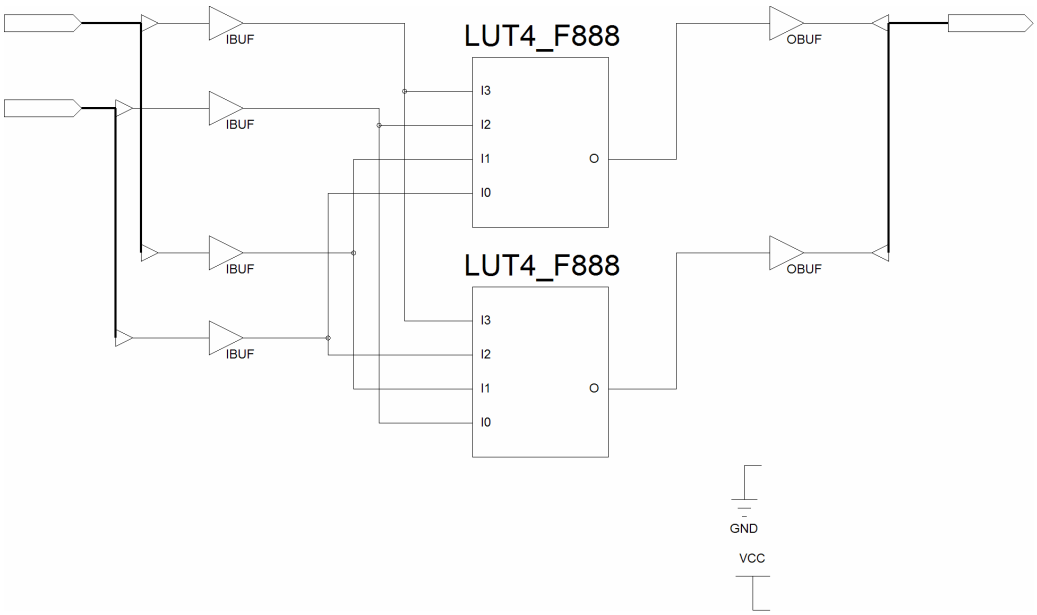
StandartHücre için blok şema ve devre şeması Şekil 6.5 ve Şekil 6.6 da verilmiştir.

Xilinx VirtexE ailesi FPGA'ler üzerinde StandartHücre gerçeklemesinde toplam 7 adet dilim ve 4 girişli 12 adet doğruluk tablosu kullanılmıştır.

Ardışıl hücre dizileri şeklinde tasarlanan devre için son olarak SonHücre adı verilen hücre yapısı tasarlanarak devre sonlandırılır. Bu hücrenin yaptığı iş tam olarak Standarthücre ile aynıdır ancak giriş parametrelerinden bir kısmı yoktur. Dolayısıyla Standarthücrenin basitleştirilmiş hali olarak tanımlanabilir. Bahsedilen hücreye ait blok şema ve devre şeması Şekil 6.7 ve Şekil 6.8 de verilmiştir.



Şekil 6.7 : SonHücre Blok Şeması



Şekil 6.8 : SonHücre Devre Şeması

Blok şemaları ve devre şemaları verilen 3 tip hücrenin ardışıl şekilde kullanılarak algoritmayı gerçekleyen devre için blok şeması Şekil 6.2 de verilmiştir.

Bu şemada görüldüğü gibi hücreler birbirine 2 bitlik veri yolları ile bağlanmıştır. Veri aktarımı doğrudan değil birer saklayıcı üzerinden yapılmaktadır. Saklayıcı üzerinden veri aktarımı bir saat işareti gecikmeye neden olmaktadır ve devrenin

algoritmayı doğru gerçekleyebilmesi için de bu gecikmeyle veri aktarımına ihtiyaç vardır. Bunun nedeni algoritmadan görülebilir. Şöyle ki, algoritmada 4. adımda görüldüğü gibi $c(x)$ hesaplanırken bir önceki döngüde oluşan $c(x)$ değeri kullanılmaktadır. Ardışıl hücre dizileriyle tasarımda $c(x)'$ in her basamağı bir hücreyle hesaplandığından, $c(x)$ ' in basamakları için döngünün bir önceki turunda kendinden bir sonraki hücrede oluşan değer kullanılmaktadır. Bahsedilen durum $c_{i,j} = c_{i-1,j+1} + a_i.b_j + u_i n_j$ eşitliği ile ifade edilebilir. Sonuç olarak, bir hücre işlem yaparken kendinden sonraki hücrenin çıkış değerini kullanmaktadır. j numaralı hücre i saat işaretinde başlangıç değerleriyle işlem yapıp çıkışını $(j+1)$ numaralı hücreye verir. $(i+1).$ saat işaretinde j numaralı hücre yeni girişler için işlem yapmaz, $(j+1)$ numaralı hücreyi bekler çünkü bir sonraki işlem için $(j+1)$ numaralı hücrenin çıkışına ihtiyacı vardır. Bu bekleme işlemi aradaki saklayıcılarla sağlanır. Saklayıcılar arasında veri transferi için bir saat işareti beklenerek bu işlem gerçeklenmektedir.

Ardışıl dizi hücreleri yapısı, en uzun yol gecikmesini en aza indirmektedir. Hücre dizisi yapısına çarpanlardan biri paralel olarak uygulanmakatadır, diğer çarpan ise İlkHücre yapısından SonHücre yapısına doğru öteleyerek gelmektedir. Çarpma işlemi her hücrede ayrı ayrı yapıldığı için en uzun yol gecikmesi sadece bir hücrenin girişinden çıkışına kadar olan yol kadardır.

Ardışıl hücre dizileri yapısında çarpanlardan birisi ve indirgenemez polinom değerleri her hücreye doğrudan uygulanır. İkinci çarpanın en düşük anlamlı basamağı İlkHücreye her iki saat işaretinde bir verilir ve hücreye uygulandıktan sonra çarpanın saklı olduğu saklayıcı sağa bir basamak (iki bit) kaydırılır. İki saat işaretinde bir çarpanın bir basamağı hücreye verildiğine göre basamak sayısının iki katı saat işareti kadar sürede çarpanın bütün basamakları hücre dizisinin ilk hücresine verilmiş olur. İlk hücreye verilen son basamağın hücre dizisinin son hücresine ulaşması için geçen zaman hücre dizisi sayısının iki katı süredir. Sonuç olarak çarpanları i basamaklı olan j hücreli bir devrede $2i$ saat işaretinde çarpanın tüm basamakları devre girişine gelir ve $2j$ saat işaretinde de son uygulanan basamak hücre dizisinin son hücresine gelir, yani toplam $2i + 2j$ saat işaretinde devre çalışmasını tamamlar.

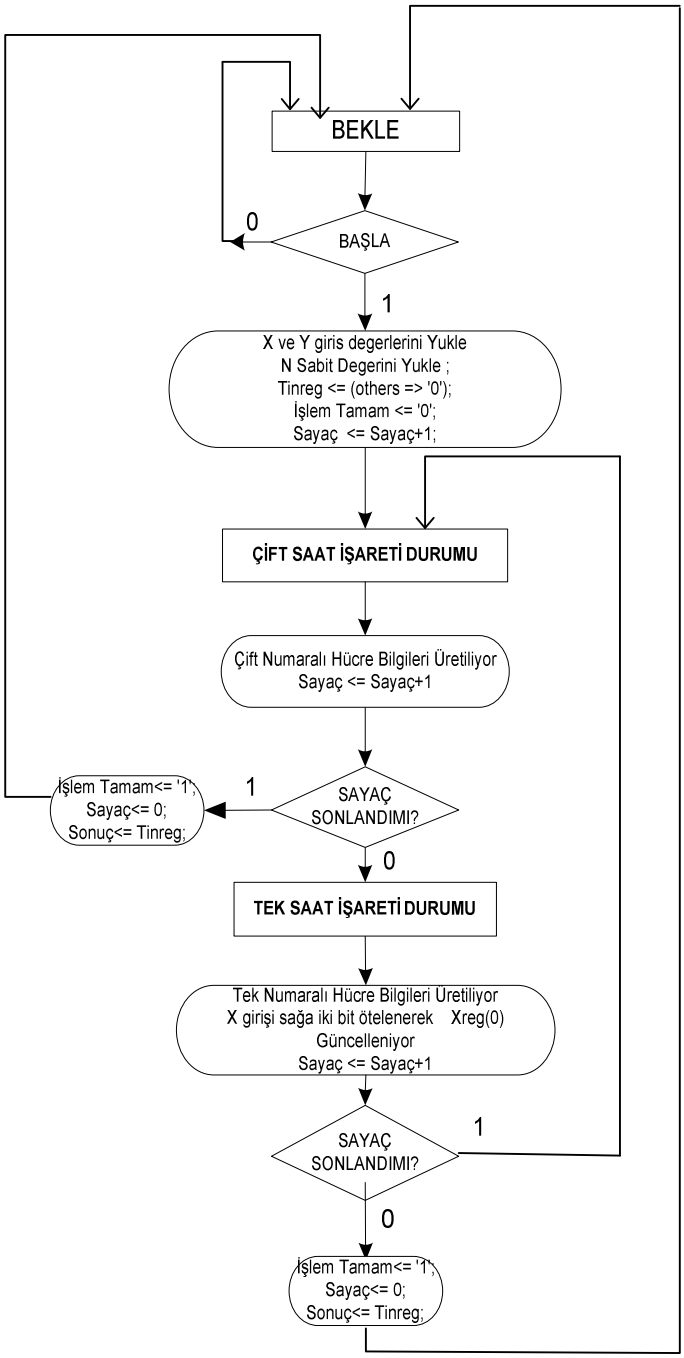
Şimdiye kadar alt bloklarından bahsedilen ve blok şemaları ile devre şemaları verilen ardışıl hücre dizileri şeklinde oluşturulan MMÇ devresi için blok şema Şekil 6.2 de verilmiştir.

Oluşturulan yapı Xilinx VirtexE ailesi FPGA'ler üzerinde toplam 951 dilimlik yer kaplamaktadır. Ayrıca toplam 1719 adet 4 girişli doğruluk tablosuna ihtiyaç vardır. Bu devrenin en fazla çalışma frekansı benzetim programıyla Xilinx VirtexE ailesi FPGA için 80,723 MHz yani en düşük periyodu 12.388ns olarak hesaplanmıştır.

Çarpma bloğu eliptik eğri gerçeklemesinde en önemli alt bloktur. Diğer işlemlerin birçoğu çarpma bloğunu kullanarak kendi işlemini gerçekleştirmektedir. Bu yüzden çarpma bloğunun performansı diğer birçok alt blokları etkilemektedir. Yukarıda verilen sonuçların yanısıra MMÇ devresinin kullanılması, eliptik eğri işlemcisinin gerçeklenmesi aşamasında da bir çok kolaylık sağlamaktadır şöyle ki; MMÇ devresi sayesinde modüler çarpma işlemi gerçeknirken ayrı bir modüler indirgeme bloğuna ihtiyaç yoktur. Modüler indirgeme için kullanılan devre yapıları gerçekleme sırasında alan maliyeti olan devrelerdir. Ayrıca çalışma zamanı anlamında da indirgeme blokları devre performansını etkilemektedir. Bazı özel seçilmiş indirgeme polinomları için tasarlanmış uygulamaya özel indirgeme blokları tasarlanıp alan-zaman performans artışına gidilebilmektedir. Bu çalışmada kullanılan MMÇ devresi tüm indirgeme polinomları için aynı karakteristiği göstermektedir. Tüm indirgeme polinomları için aynı zamanda çarpmayı gerçekleştirmektedir yani performans polinomların derecesinden ve katsayılarından bağımsızdır.

MMÇ devresinin bir diğer özelliği genişletilebilir bir yapıdadır. Sadece SıradanHücre olarak tanımlanan sonuç değerinin basamaklarını hesaplayan hücrelerin sayısı arttırılarak ve istenen dereceden indirgeme polinomu seçilerek daha yüksek dereceli modüler çarpmalar gerçekleştirilebilir.

Ayrıca sıralı hücre dizileri yapısı nedeniyle çarpılan polinomların derecesinin en fazla yol uzunluğuna bir etkisi yoktur. Başka bir deyişle en fazla yol uzunluğu devreden bağımsızdır. Bu özellik yüksek dereceli çarpma işlemleri için olumlu bir özelliktir.



Şekil 6.9 : MMÇ Algoritması için Algoritmik Durum Makinası

Çarpma devresi için alan ve zaman performans sonuçları Tablo 6-1 de verilmiştir.

Tablo 6-1 : MMÇ Devresi İçin Performans Değerleri

Tasarım	Parametre	Saat İşareti Sayısı (S)	Alan (Dilim)	Saat İşareti Hızı (MHz) (f)	Min. saat periyodu (nsn) (Tp)	Toplam Çarpma Süresi (µsn) (S/f)	Performans (Throughput Rate)(Mb/s) (k / (S/f))
Xilinx Virtex1000E	k = 97 (=194 bit)	388	951	80,723	12,388	4,8	40,41

Çarpma devresinin performansını daha önceki çalışmalarla tam olarak karşılaştırma şansı bulunmamaktadır çünkü $GF(p^m)$ sonlu alanlarında çeşitli çarpma yöntemleri önerilmesine rağmen Montgomery modüler çarpma devresi tasarımına rastlanamamıştır. Aynı şekilde $GF(p)$ ve $GF(2^m)$ sonlu alanları için Montgomery modüler çarpma devresi tasarımları olmasına rağmen $GF(p^m)$ şeklindeki sonlu alanlarda bir tasarım bulunamamıştır. Yine de fikir vermesi açısından yukarıda verilen sonuçlar [37]' nin sonuçları ile karşılaştırılabilir. [37] için sonuçlar Tablo 6-2'de verilmiştir.

Tablo 6-2 : Page D., Smart N.P. nin $GF(3^{97})$ 'de Donanım Uygulaması Sonuçları

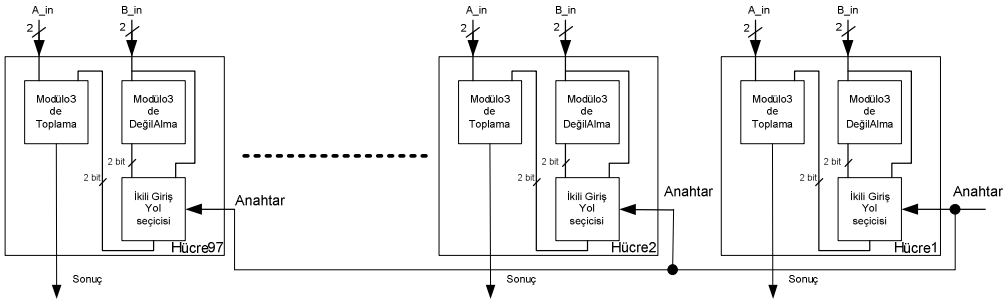
Alan	Toplama (µs)	Çarpma (µs)	Alan(Dilim)
3^{97}	1.15	50.68	8733
2^{241}	0.70	37.32	10139

Aynı parametrelerle karşılaştırma şansı olmamasına rağmen eldeki sonuçlar incelendiğinde bu çalışmada bir Montgomery çarpması yapmak için Tablo 6-4' den normal gösterilimden Montgomery gösterilime geçiş, çarpma işlemi ve Montgomery gösterilimden normal gösterilime geriye dönüş toplam süresi yaklaşık 24µs olmaktadır. Zaman performansı olarak daha iyi bir sonuç yakalanmış olmasına rağmen sonlu alanın aynı olmasının yanında birçok parametrenin farklı olduğu düşünülürse bu iki sonucun birebir karşılaştırması çok doğru bir yaklaşım olmayabilir.

6.2. Modüler Toplama-Çıkarma Devresi

$GF(3^m)$ sonlu alanı için modüler toplama-çıkarma devresi, girişine verilen polinomların derecelerini modülo 3 de toplama ya da çıkarma yapabilen devredir.

Devre girişinde bir anahtar kullanılmıştır ve anahtarın 0/1 konumuna göre devre toplama ya da çıkarma yapmaktadır. Devre modülo3 de toplama ya da çıkarma yapan hücrelerin ardışıl dizilmesiyle oluşturulmuştur. Devre boyutu, bir hücrenin işlem yapılacak polinomun derecesi kadar tekrarlanmasıyla genişletilebilir. Bu çalışmada $GF(3^{97})$ sonlu alanı kullanıldığı için, 97 basamaklı iki sayının modüler toplama-çıkarma işlemi yapılmaktadır. Dolayısıyla aynı hücre yapısı 97 kere ardarda tekrarlanarak paralel çalışan bir devre oluşturulmuştur. Bu yapının avantajı tek saat işaretinde 97 basamaklı (modülo 3 deki sayılar için ikili gösterimde her basamak 2 bitle ifade edildiği için 194 bitlik) iki sayıyı işleyebilmektedir. Diğer bir olasılık ise tek hücre kullanıp üzerine bir kontrol devresi tasarlamaktır. Giriş değerilerinin basamakları sırayla bu tek hücrenin girişine uygulanıp aynı hücreyi farklı girişlerle basamak sayısı kadar çağırılabilir. Bu yöntemde alandan büyük kazanç sağlanmakla birlikte aynı oranda zamandan kayıp vardır. Devre blok şeması Şekil 6.10 da verilmiştir.



Şekil 6.10 : Modüler Toplama-Çıkarma Devresi Blok Şeması

Devre, Xilinx VirtexE ailesi FPGA'ler üzerinde toplam 423 dilimlik alan kaplamaktadır. Toplam 789 adet 4 girişli doğruluk tablosuna ihtiyaç vardır. En fazla çalışma frekansı 112.082 MHz yani en düşük periyodu 8.922ns dir. Bu alan performansı değiştirilebilir. Tek bir modüler toplama bloğu ardarda kullanılması halinde alanda artışla birlikte aynı oranda çalışma frekansında bir düşüş olması beklenmektedir.

6.3. Projektif Koordinatlarda Nokta Toplama-Nokta İkileme Devreleri

Projektif koordinatların gerekliliğinden ve kullanım şeklinden Bölüm 4.4.1' de detaylı olarak bahsedilmiştir. Bu bölümde önceden bahsedilen projektif koordinatlarda nokta toplama işlemleri için oluşturulması gereken devre yapısından bahsedilecektir.

Bölüm 4.4.1 de eliptik eğri için nokta toplama ve nokta ikileme işlemleri şu şekilde verilmişti;

$P = (x_1, y_1, z_1)$ ve $Q = (x_2, y_2, z_2)$ E eliptik eğrisi üzerindeki noktaların projektif koordinatlardaki gösterimi olmak üzere, $P + Q = (x_3, y_3, z_3)$ şu şekilde ifade edilir;

Nokta Toplama:

$\lambda_1 = x_1 z_2^{\,2}$	2Çarpma
$\lambda_2 = x_2 z_1^{\,2}$	2Çarpma
$\lambda_3 = \lambda_1 - \lambda_2$	
$\lambda_4 = y_1 z_2^{\,3}$	1Çarma, 1Küp Alma
$\lambda_5 = y_2 z_1^{\,3}$	1 Çarpma, 1 Küp Alma
$\lambda_6 = \lambda_4 - \lambda_5$	
$\lambda_7 = \lambda_1 + \lambda_2$	
$\lambda_8 = \lambda_4 + \lambda_5$	
$z_3 = z_1 . z_2 . \lambda_3$	2 Çarpma
$x_3 = \lambda_6^{\,2} - \lambda_7 . \lambda_3^{\,2}$	3 Çarpma
$y_3 = \lambda_8 . \lambda_3^{\,3} - \lambda_6^{\,3}$	1 Çarpma, 2 Küp Alma

Nokta İkileme:

$P = (x_1, y_1, z_1), E$ eliptik eğrisi üzerindeki noktanın projektif koordinatlardaki gösterilimi olmak üzere, $P + P = (x_3, y_3, z_3)$ şu şekilde ifade edilir;

$\lambda_1 = -z_1^{\,4}$	1 Çarpma, 1 Küp Alma

$z_3 = -y_1.z_1$ 1 Çarpma

$\lambda_2 = x_1 y_1^{\,2}$ 2 Çarpma

$x_3 = \lambda_1^{\,2} + \lambda_2$ 1 Çarpma

$\lambda_3 = -y_1^{\,4}$ 1 Çarpma, 1 Küp Alma

$y_3 = \lambda_1.(\lambda_2 - x_3) - \lambda_3$ 1Çarpma

Yukarıda bahsedilen nokta ikileme ve nokta toplama eşitliklerinde birçok çarpma ve toplama işlemi vardır. Çarpma işlemi bölüm 6.1 de bahsedilen MMÇ devresiyle, toplama işlemi bölüm 6.2 de bahsedilen modüler toplama-çıkarma devresini kullanarak gerçeklenecektir. Burada yeni bir yapı oluşturma ihtiyacı yoktur. Daha önceden oluşturulmuş çarpma ve toplama devreleri uygun bir akışta kullanılarak işlem gerçeklenir.

Nokta toplama ve nokta çarpma için verilen eşitliklerde, saklayıcılarda oluşturulan ara değerler kullanıldıktan sonra ara değerin olduğu saklayıcı boşa çıkmaktadır. Daha sonraki değerler için yeni saklayıcı tanımlayıp daha fazla alan harcamak yerine önceden oluşturulup kullanılmış ve artık sakladığı veriye gerek duyulmayan saklayıcılar tekrar kullanılabilir. Saklayıcıların organizasyonu aşağıdaki yapılabilir.

$R_1 = z_2^{\,2}$, $R_2 = x_1 R_1$, $R_3 = z_1^{\,2}$, $R_4 = x_2 R_3$, $R_5 = R_2 - R_4$, $R_6 = R_2 + R_4$

---------- R2 ve R4 BOSTA

$R_2 = R_1 z_2$

---------- R1 ve R4 BOSTA

$R_1 = R_2 y_1$

---------- R2 ve R4 BOSTA

$R_2 = R_3 z_1$

---------- R3 ve R4 BOSTA

$R_3 = R_2 y_2$

---------- R2 ve R4 BOSTA

$R_2 = R_1 - R_3$

---------- R4 BOSTA

$R_4 = R_1 + R_3$

---------- R1 ve R3 BOSTA

$R_1 = z_1 z_2$

---------- R3 BOSTA

$z_3 = R_1 R_5$

---------- R1 ve R3 BOSTA

$R_1 = R_5 R_5$

---------- R3 BOSTA

$R_3 = R_1 R_6$

---------- R6 BOS

$R_6 = R_2 R_2$

---------- BOS YOK

$x_3 = R_6 - R_3$

---------- R3 BOSTA

$R_3 = R_6 R_2$

---------- R6 ve R2 BOSTA

$R_6 = R_1 R_5$

---------- R1,R2 ve R5 BOSTA

$R_1 = R_6 R_4$

---------- R2,R6 ve R4 ve R5 BOSTA

$y_3 = R_1 - R_3$

---------- R1, R2, R3, R4, R5, R6 BOSTA

Başta $R_1, R_2, R_3, R_4, R_5, R_6$ şeklinde 6 adet saklayıcı tanımlayıp yukarıda gösterildiği gibi tüm işlemler bu 6 saklayıcıyla gerçeklenebilir. İşlemler sırasında boşa çıkan saklayıcılar işlem sonrasında belirtilmiştir.

Nokta toplama devresi, Xilinx VirtexE ailesi FPGA'ler üzerinde toplam 5100 dilim ve 9532 adet 4 girişli doğruluk tablosu büyüklüğünde yer kaplamaktadır.

Nokta çarpma devresi ise aynı aileden FPGA'ler üzerinde toplam 4634 dilim ve 8693 adet 4 girişli doğruluk tablosu büyüklüğünde yer kaplamaktadır.

6.4. Modüler Çarpmaya Göre Ters Alma Devresi

Bölüm 4.8 de modüler çarpmaya göre ters alma işlemi ile ilgili matematiksel kavramlar verilmiştir. Bu işlem için Fermat teoreminden yararlanılmaktadır [9]. Eliptik eğrilerin oluşturulabileceği sonlu alanlarda çarpmaya göre ters alma işlemi ile ilgili eşitlikler Tablo 4-1 de verilmiştir. Bu çalışmada $GF(p^m)$ sonlu alanı kullanıldığı için tablodaki son satırda verilen eşitliklerden yararlanılır. Buna göre ; $GF(p^m)$' de indirgenemez polinom $F(x) = p_m x^m + p_{m-1} x^{m-1} + ... + p_1 x + p_0$, $p_i \in \{0,1,..., p-1\}$ olmak üzere $a^{-1} = a^{(p^m-2)} \bmod F(x)$ olur. Bu çalışmada $GF(3^{97})$ sonlu alanı ve $N(x) = x^{97} + 2x^{12} + 1$ değerleri için şu eşitlik oluşur;

$$a^{-1} = a^{(3^{97}-2)} \bmod N(x) \qquad \textbf{(6.1)}$$

Algoritma 4-5' i kullanabilmek için $(3^{97}-2)$ değerinin tam olarak hesaplanıp ikili tabanda ifade edilmesi gerekmektedir. $(3^{97}-2)$ değeri hesaplandığında;

$(3^{97}-2) = 190880563234078270754244862876156026926706489 61$

olarak bulunur. Daha sonra bu değer ikili tabanda ifade edildiğinde [11010011111111000000100011111100001101110000100000010011110000110011011100100001000110110010101001001000011111000110101101110110000100010001110011001010000001]$_2$ şeklinde 154 bitlik bir ifade oluşur. Tüm bu değerlerle algoritma aşağıdaki gibi oluşur;

Algoritma 6-1: $GF(3^{97})$ için kareal-ve-Çarp Algoritması

Girişler : $a(x), c(x)_2 = (110100111.....), N(x) = x^{97} + 2x^{12} + 1$

Çıkış : $z(x) = a(x)^{c(x)} \bmod N(x) = z(x) = a(x)^{(1101001111.....)} \bmod N(x)$

1: **for** i **from** 153 **downto** 0

2: $z(x) = z(x).z(x) \bmod N(x)$

3: if $c(i) = 1$

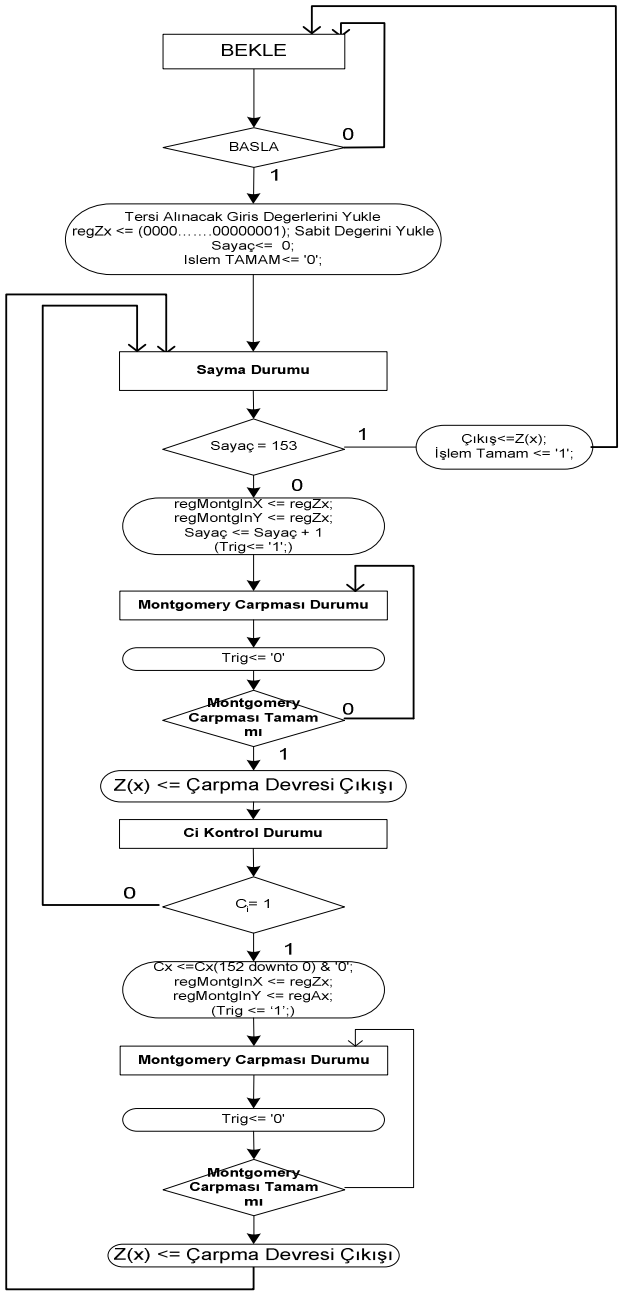
4: **then** $z(x) = z(x).a(x) \bmod N(x)$

5: **end if;**

6: **end for;**

7: **return** $z(x)$

Algoritmada görüldüğü gibi, döngünün her turunda bir modüler çarpma işlemi ve üs değerinin bit değerine göre tekrar bir modüler çarpma işlemi daha yapılmaktadır. Modüler çarpma işlemini gerçekleştirmek için daha önceden bölüm 6.1 de bahsedilen MMÇ devresi kullanılacaktır. Dolayısıyla çarpmaya göre modüler ters alma devresinin tasarımında MMÇ devresinin bir akış şeması ile kontrol edilmesi gerekmektedir. Modüler çarpmaya göre ters alma devresi için algoritmik durum makinası Şekil 6.11 de verilmiştir.



Şekil 6.11 : Modüler Çarpmada Evrik Alma İçin Algoritmik Durum Makinası

Bu devre, Xilinx VirtexE ailesi FPGA'ler üzerinde toplam 1876 dilim ve 3179 adet 4 girişli doğruluk tablosu büyüklüğünde yer kaplamaktadır.

6.5. İlgin Koordinatlar-Projektif Koordinatlar Arası Dönüşüm

Bölüm 4.4.1 de projektif koordinatlara geçiş ve projektif koordinatlarda işlemler ile ilgili detaylı bilgi verilmiştir. Bu bölümde iki koordinat sistemi arasında geçişin nasıl gerçekleneceğinden bahsedilecektir.

$Char(K) = 3$ için seçilen projektif koordinat dönüşümünün $(x, y) \rightarrow \quad (X, Y, Z) = (x.Z^2, y.Z^3, Z)$ olduğu daha önce Bölüm 4.4.1' de anlatılmıştı.

Bu dönüşüm için bir Z değeri seçilmelidir. $Z = 1$ seçilmesi ile dönüşüm şu şekli alır;

$$(x, y) \rightarrow (X, Y, Z) = (x, y, 1) \qquad \textbf{(6.2)}$$

İlgin koordinat sisteminden projektif koordinat sistemine geçiş için özel bir devre tasarımına ihtiyaç yoktur. Z = 1 seçimiyle giriş parametreleri çıkışa aynen aktarılmaktadır.

Ptojektif koordinatlarda işlemler gerçeklendikten sonra ters yönde bir dönüşüme ihtiyaç vardır. Bunun için ilgili formüllerden geriye dönüş şu şekildedir;

$$(X, Y, Z) \rightarrow (x, y) = (X / Z^2, Y / Z^3) \qquad \textbf{(6.3)}$$

Bu dönüşüm için yine modüler çarpma işlemleri ve 1 kere modüler çarpmaya göre ters alma işlemine ihtiyaç vardır. Çarpmalar yine MMÇ devresiyle yapılmaktadır. Bu amaçla ilk olarak bölüm 6.4 de tasarımı anlatılan modüler çarpmaya göre ters alma devresi girişine Z uygulanır ve $1/Z$ elde edilir. Daha sonra $1/Z$ değeri MMÇ devresine giriş parametreleri olarak verilir ve $1/Z^2$ değeri hesaplanır. Daha sonra bu değer ile $1/Z$ değeri MMÇ devresine giriş parametreleri olarak verilir ve $1/Z^3$ elde edilir. Bir sonraki adımda $1/Z^2$ değeri ile X girişi yine modüler çarpma devresine uygulanır. Elde edilen sonuç geri dönüşüm sonrası ilgin koordinat sistemindeki x değeridir. Benzer şekilde $1/Z^3$ değeri ile Y değeri modüler çarpma devresine verilerek ilgin koordinat sistemindeki y değeri elde edilir.

Yukarıda bahsedilen devre için Virtex E ailesi üzerinde gerçeklemede toplam 4115 dilime ve 7185 adet 4 girişli doğruluk tablosuna ihtiyaç vardır.

6.6. Normal Gösterilim-Montgomery Gösterilim Arası Dönüşüm

Eliptik eğri uygulamasında tüm çarpma işlemleri için MMÇ devresinin kullanılması bölüm 6 içerisinde bahsedilmiştir. Daha önce Bölüm 4.6 da bahsedildiği gibi Montgomery algoritması çarpmayı ve indirgeme işlemini aynı anda yapmakla beraber, sonuç değerinde istenmeyen bir $r(x)^{-1}$ değeri getirmektedir. Bu değerden kurtulmak için Montgomery gösterilimde işlem yapmak gerekmektedir. Şöyle ki, eğer MMÇ algoritmasına giriş değerlerini doğrudan değil de Montgomery gösterilimde verilirse sonuç da Montgomery gösterilim şeklinde elde edilir. MMÇ devresini kullanarak yapılan tüm işlemlerde Montgomery gösterilim kullanarak işlemler yapılır ve tüm işlemler tamamlandıktan sonra Montgomery gösterilimden normal gösterilime geriye dönüşüm yapılırsa sonuçlar istenen şekilde elde edilmiş olur.

Bir a sayısının Montgomery gösterilimi, $a' = Mont(a, r^2) = ar \bmod N$ şeklindedir. Montgomery gösterilimle ifade edilen iki sayı $a' = ar$ ve $b' = br$ olup bu sayıların Montgomery çarpması $c = Mont(ar, br) = arbrr^{-1} \bmod N = abr \bmod N$ olarak elde edilir. Bu da istenen $abModN$ sonucunun r ile çarpılmış hali yani Montgomery gösterilimidir.

Montgomery gösterilim için $N(x)$ indirgeme polinomu olmak üzere, $r(x)^2 ModN(x)$ değeri hesaplanmalıdır. Algoritma 4-4 den görüleceği gibi, k sonlu alanın derecesi olmak üzere $r(x) = x^k$ olur. Bu çalışma için $k = 97$ ve $N(x) = x^{97} + 2x^{12} + 1$ değerleri tasarım başında seçilmiş değerlerdir. $r(x) = x^{97}$ ve $r(x)^2 ModN(x) = (3^{97})^2 Mod(x^{97} + 2x^{12} + 1)$ dir. Bu değer hesaplandığında; $r(x)^2 ModN(x) = x^{24} + x^{12} + 1$ bulunur. Hesaplanan $r(x)^2 ModN(x)$ değeri dönüşüm devresi için sabit bir giriş parametresidir. Diğer giriş, Montgomery gösterilimi elde edilmek istenen değerdir ve çıkış da yine bu değerin Montgomery gösterilimidir. Devre MMÇ devresinin uygun girişlerle bir kere çağırılmasından oluşmaktadır.

Eliptik eğri nokta çarpmasıyla ilgili tüm işlemleri bittikten sonra çarpımın sonucu olan noktanın koordinatları normal gösterilimde ifade edilmelidir. Bunun için Montgomery gösterilimden normal gösterilime geçiş yapmak gerekir.

Montgomery gösteriliminden normal gösterilimine geçiş şu şekilde gerçekleşmektedir; $a' = ar$, Montgomery gösterilimde bir sayı olmak üzere, bu sayının normal gösterilimi, Montgomery çarpma devresinin girişlerine a' ve 1 değerleri uygulanarak bulunur.

$$Mont(a',1) = = a'.1.r^{-1} ModN = ar.1.r^{-1} ModN = aModN \quad \textbf{(6.4)}$$

Bu dönüşüm MMÇ devresini sadece 1 kere çalıştırarak elde edilir.

Normal gösterilimden Montgomery gösterilime geçiş için gerekli devrenin VirtexE ailesi FPGA ler üzerinde kapladığı alan; 1658 dilim ve 1718 adet 4 girişli doğruluk tablosudur.

Montgomery gösterilimden normal gösterilime geçiş için gerekli devrenin kapladığı alan ise; 1481 dilim ve 1664 adet 4 girişli doğruluk tablosudur.

6.7. Eliptik Eğri Nokta Çarpma Devresi

Eliptik eğri nokta çarpma devresi sadece bir kontrol devresidir. Devre eliptik eğri nokta çarpmasını gerçeklemek üzere daha önce Bölüm 4.5' de ikile-ve-topla algoritmasıyla ifade edilen işlemi kontrol eder. Devreye, eğri üzerinde bir noktanın koordinatları ve noktanın çarpılacağı k tamsayısının ikili tabanda gösterilimi giriş olarak verilir. Algoritmada, her turda nokta ikileme işlemi ve anahtarın 1 olan bitleri için bir nokta toplama işlemi gerçekleştirilmektedir. Bu devrenin kontrol edeceği nokta ikileme ve nokta toplama işlemleri bölüm 6.3 de detaylı şekilde anlatılmaktadır. Devreye ilişkin algoritmik durum makinası Şekil 6.12 de verilmiştir.

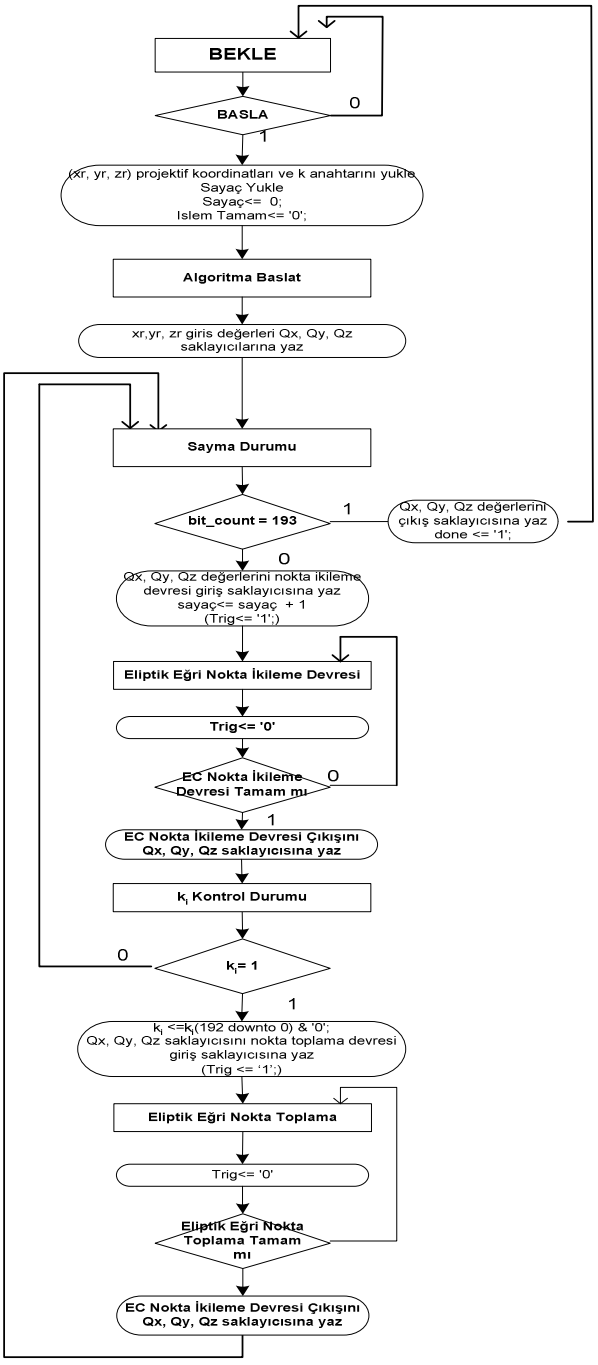
Eliptik eğri nokta çarpma devresi 11758 adet dilimlik yer tutmaktadır. Ayrıca 20482 adet 4 girişli dogruluk tablosuna ihtiyaç vardır.

6.8. Eliptik Eğri İşlemcisi

Tüm alt blokların kullanılmasıyla eliptik eğri nokta çarpma devresi tasarımı bölüm 6.7 de verilmiştir. Eliptik eğri işlemcisi Şekil 6.1 de verilen hiyerarşik yapıda en yukarıda yer alan ana kontrol devresi ve bu bloğun kontrol ettiği alt blokların tamamıdır.

İşlemci giriş parametresi olarak eliptik eğri üzerindeki bir noktanın ilgin koordinatlarını ve bu noktanın çarpılacağı tam sayı değerini alır. Çıkış olarak da yine eğri üzerinde bir noktanın koordinatlarını verir. Ayrıca saat işareti, reset, başla ve bitti işaretleri de işlemcininin dış dünyayla haberleşmesinde görev alan diğer sinyallerdir. Herhangi bir anda reset sinyali gelirse işlemci ilk saat işaretinin yükselen kenarında işlemini bırakıp tüm giriş ve çıkış saklayıcılarını sıfırlayarak başlangıç durumuna döner ve bekleme modunda sürekli başla sinyalini bekler. Başla sinyali ile birlikte giriş veri yolu üzerindeki veriyi okuyup dahili bir saklayıcıya alarak dış dünyayla bağlantısını koparır. Artık giriş veri yolu üzerindeki değişiklikler işlemciyi etkilemeyecektir. Başla işaretinin ardından ilk olarak normal gösterilimde verilen giriş parametresi koordinatları projektif koordinatlara çevirir. Bu iş için bölüm 6.5 de bahsedilen alt bloğu kullanır. Giriş değerlerini koordinat dönüşüm bloğunun girişindeki saklayıcılara yazarak bu bloğa başla sinyalini verir ve sonucu bekler. Bu blok işini bitirdikten sonra ana kontrol devresine bitti sinyali göndererek projektif gösterilimdeki koordinatları çıkışındaki saklayıcılara yazar. Ana kontrol devresi bu veriyi alıp Montgomery gösterilime çevirir. Bu amaçla bölüm 6.6 da bahsedilen alt bloğu yine başla ve bitti sinyalleri kontrolünde kullanır. Montgomery dönüşüm devresinden bitti sinyalini gören ana kontrol devresi çıkışındaki saklayıcıdan bu verileri alır. Artık giriş değerleri gerekli dönüşüm bloklarından geçmiştir ve çarpma işlemini yapmaya hazır şekildedir.

Çarpmaya göre ters alma işleminden kurtulmak için projektif koordinatlara geçilmiştir. Ayrıca modüler çarpma işlemleri için MMÇ devresinde kullanılmak üzere Montgomery gösterilime geçilmiştir. Ana kontrol devresi bir sonraki aşamada nokta çarpması yapmak üzere elde ettiği değerleri bölüm 6.7 de bahsedilen eliptik eğri nokta çarpması alt bloğu girişine verir. Ayrıca bu blok girişine noktanın çarpılacağı k değerini de yazar. Bu alt blok da kendi altında bölüm 6.1 de bahsedilen MMÇ devresini yine başla ve bitti sinyalleri kontrolünde kullanarak nokta toplama ve nokta ikileme işlemleri için gerekli algoritmaları ve ilgili komutları yürütüp eliptik eğri nokta çarpma işlemini gerçekler ve kontrol devresine bitti sinyali gönderir.



Şekil 6.12 : Eliptik Eğri Nokta Çarpması İçin Algoritmik Durum Makinası

Şu aşamada eliptik eğri şifrelemesi için nokta çarpma işlemi bitmiştir ancak sonuçlar projektif koordinat sisteminde ve Montgomery gösterilimdedir. İşlemciye verilen formda çıkış üretebilmek için bu değerler ilk önce projektif koordinat sisteminden ilgin koordinat sistemine dönüşüm devresine ve ardından Montgomery gösterilimden

normal gösterilime dönüşüm devresine gönderilir. Bu alt devrelerin çalışması da diğer alt bloklar gibi ana kontrol devresi tarafından gönderilen saat işareti ve başla komutuyla başlar, bitti komutunun dışarı verilmesiyle sona erer. Projektif koordinat sisteminden ilgin koordinat sistemine dönüşüm devresi işlemini gerçekleştirirken bölüm 6.4 de bahsedilen çarpmaya göre ters alma devresi sadece bir kere kullanılır. Böylece ilgin koordinat sisteminde çalışıp defalarca bu işlem yapılmamıştır sadece bir kere sonda yapılarak işlem yükü azaltılmıştır.

Tüm alt bloklar işlemlerini tamamladığında çıkış saklayıcısına son değeri yazmaktadır. Ana kontrol devresi son bloğun bitti sinyalini gördüğünde çıkış saklayıcısında sonuç değerinin hazır olduğunu bilmektedir. Son bloğun bitti sinyali ile işlemci çıkış saklayıcısını çıkış veri yoluna yazar ve işlemci kendi bitti sinyalini dışarıya verip tekrar bekleme durumuna geçer.

Eliptik eğri işlemcisi için gerekli tüm alt blokların alan değerleri Tablo 6-3' de ayrı ayrı verilmiştir.

Tablo 6-3 : Eliptik Eğri Alt Bloklarının Alan Performansları

İşlem Bloğu	**Alan Performansı** (Virtex 1000E)	
	Dilim (Slice) Sayısı	**Doğruluk Tablosu Sayısı**
Normal ->Mont	1658	1718
EC Nokta Çarpma	11758	20482
Projektif-> (x,y)	4115	7185
Mont->Normal	1481	1664
Nokta İkileme	4634	8693
Nokta Toplama	5100	9532
Modüler Ters Alma	1876	3179
Modüler Toplama	423	789
Montgomery Çarpması	951	1719

Eliptik eğri işlemcisi içinde yer alan tüm alt bloklar için zaman bilgisi Tablo 6-4 de dataylı olarak verilmiştir. En üstte yer alan ana kontrolör tüm blokları çağıracağı için tüm alt blokların çalışma frekansı olarak ana kontrolörün çalışma frekansı alınmıştır.

Tablo 6-4 : Eliptik Eğri Alt Bloklarının Zaman Performansları

İşlem Bloğu	Kullandığı Alt Blok	Saat İşareti Sayısı	İşlem Süresi (µsn)
Normal ->Mont	2 MMÇ	776	10.34
EC Nokta Çarpma	k Nokta İkile ,(k/2) Nokta Toplam	1134318	15124
Projektif-> (x,y)	4 MMÇ+ 1Mod.TersAlma	91180	1215.7
Mont->Normal	2 MMÇ	776	10.34
Nokta İkileme	8 MMÇ+ 6 Mod Toplama	3122	41.62
Nokta Toplama	14 MMÇ + 6 Mod.Toplam	5450	72.66
Mod. Ters Alma	3l /2 MMÇ	89628	1195
Mod. Toplama	-	3	0.04
MMÇ	-	4N = 388	5.17
Ana Kontrolör	Tüm Bloklar	1225498	16339

N = 97 ve k = 194.
l = 154 (3^{97}-2 değerinin ikili tabanda gösterilimindeki basamak sayısı)

[19]'da verilen çalışma ile bu tezde verilen sonuçların karşılaştırılması $GF(p^m)$ sonlu alanına geçildiğinde devrenin kapladığı alandaki artışı açıkça göstermektedir. İki çalışmanın alan yönünden karşılaştırılması Tablo 6-5'de verilmiştir.

Tablo 6-5 : $GF(p)$ ve $GF(p^m)$ 'de Eliptik Eğrilerin Alan Karşılaştırması

Tasarım	Parametreler	FlipFlop Sayısı	Doğruluk Tablosu Sayısı
Bu Çalışma VirtexE ailesi	$GF(p^m)$, p= 3, m = 97	27934	32308
Ref. [19], VirtexE ailesi	GF(p), p= 160	6959	11227

Tabloya göre, $GF(p^m)$ alanına geçişte alan kullanımında ciddi bir artış vardır. Bunun nedeni $GF(p)$ ve $GF(2^m)$ sonlu alanlarında işlemler ikili tabanda yapılmakta. Toplama mantıksal AND, çarpma mantıksal EXOR işlemi ile hızlı bir şekilde tek seferde gerçekleşmektedir. $GF(3^{97})$ sonlu alanında işlemler ve işlemleri gerçekleyen devrelerde farklılıklar vardır. $GF(3)$ tabanında tüm değerler iki bit ile ifade edilmekte ve bu alanda 3 tabanında çalışan özel toplama ve çarpma devrelerine ihtiyaç vardır. Ayrıca tüm veri yolları her basamağın iki ile ifade edilmesinden dolayı iki katına çıkmaktadır. Dolayısıyla devrenin kapladığı alan da artmaktadır.

Literatürde $GF(p^m)$ için fazla çalışma bulunmamaktadır. Olanlar da farklı yöntem ve teknolojiler kullanarak eliptik eğriyi gerçeklemektedirler. Bu yüzden diğer çalışmalarla aynı parametrelerle birebir karşılaştırma şansı bulunmamaktadır ancak olabildiğince çok ortak parametreli bir karşılaştırma yapılmaya çalışılmıştır.

$GF(3^{97})$ ile ilgili literatürde yer alan çalışmalarda eliptik eğri performansı yerine, modüler çarpma performansıdır[37] [38]. Aynı teknolojileri kullanarak elde edilen sentezleme sonuçları aşağıdaki tabloda verilmiştir.

Tablo 6-6 : Çarpma Devrelerinin Karşılaştırılması

	Yöntem	**Sentezleyici/ FPGA**	**Alan**	**Çarpma Zamanı** (µsn)	**Notlar**
[38]	Tate-Pairing Duursma-Lee	Belirtilmemiş/ virtex2 Pro	Çarpıcı : 4335Dilim, Evirici : 2210 Dilim	0.9µsn	İşlemler kelime düzeyinde olup kelime uzunluğu 12 dir.
Bu Çalışma	Montgomery	Xilinx ISE / Virtex2Pro	1215 Dilim	1.66µsn (223.7 MHz de)	Teknoloji: 0.13µm
[22]	LSDE Alg.	Synopsis 3.7.1/ XCV1000	7080 Doğruluk Tab,618 FF	0.097µsn (94.4 MHz de)	$GF(p^m)$ de p nin sadece tek değerleri için optimize edilmiş bir yöntemdir.
Bu Çalışma	Montgomery	Xilinx ISE / XCV1000	1868 Dilim	3.8µsn (101,86 MHz de)	Teknoloji: 0.18µm
[37]	Seri Bit Çarpma Yöntemi	Handel-C / Xilinx4000XL	Verilmemiş	50.68µsn (20 MHz de)	Teknoloji eski. Çalışma frekansı düşük.
Bu Çalışma	Montgomery	Xilinx ISE / Virtex5 Ailesi	1019 Doğruluk Tab,1178 FF	1.054µsn (368,05 MHz de)	En yeni teknoloji. Teknoloji: 0.65nm

Tablo incelendiğinde daha önceki çalışmaların hepsi farklı yöntemler kullandığı için performans karşılaştırılmasının çok da anlamlı olmadığı görülmektedir.Tablo incelendiğinde bu çalışmanın diğer tüm çalışmalardan alan yönünden daha iyi olduğu zaman yönünden ise üzerinde çalışıldığında [38] ile karşılaştırılabilir bir seviyede olduğu görülmektedir. [22]' de verilen sonucun sadece belli bir değer için optimize olup genel bir sonuç için değerlerin verilmemiş olması karşılaştırma sırasında iyi bir

değerlendirme yapılmasını engellemektedir. [37]' de verilen değerler ile bu çalışmayı karşılaştırmak mümkün değildir çünkü kullanılan teknoloji eski olup şu anda bu sentezleyici ve FPGA temin edilememektedir. Tablonun son satırında günümüzde kullanılabilecek en son teknoloji ile elde edilebilecek sonuç verilmiş olup elde edilebilecek en yüksek performans değeri olarak ele alınabilir. Bu teknolojinin kullanılması ile performans artışı karşılığında sistem maliyetinin de aynı oranda artacağını belirtmekte fayda vardır.

7. SONUÇLAR

Bu tezde bir açık anahtar kriptosistemi olan eliptik eğri kriptosistemi üzerine çalışılmıştır. EEK'yı oluşturan parametrelerin farklı seçimiyle farklı eliptik eğri kriptosistemleri oluşturulabilmektedir. Bu çalışmada $GF(3^{97})$ sonlu alanında seçilmiştir. Bu seçimin nedeni, $GF(p)$ ve $GF(2^m)$ seçeneklerine oranla üzerinde fazla çalışılmamış bir konu olmasıdır.

Bu alanda çalışmanın getirdiği avantaj aynı anahtar uzunluğuna göre daha yüksek seviyede güvenlik sağlaması önceki bölümlerde ve referans çalışmalarda tartışılmıştır. Getirdiği avantajın yanında işlem yükü diğer iki seçeneğe göre daha fazladır. Ayrıca kırmık üzerinde kaplayacağı alan da daha fazladır. Bu sonuç Tablo 6-5' de verilen karşılaştırma sonuçları ile açıkça görülmektedir. Bunun nedeni, $GF(p)$ ve $GF(2^m)$ sonlu alanlarında çalışmada tüm işlemler ikili tabanda yapılmakta ve toplama işlemi mantıksal AND, çarpma işlemi mantıksal EXOR işlemi ile tek saat işaretinde tek mantıksal kapı ile gerçekleşmektedir. $GF(3^{97})$ sonlu alanına geçildiğinde işlemler ve işlemleri gerçekleyen devreler bu kadar basit olmamaktadır. $GF(3)$ tabanında tüm değerler iki bit ile ifade edilmektedir ve dolayısıyla bu alanda 3 tabanında çalışan özel toplama ve çarpma devrelerine ihtiyaç vardır. Ayrıca işlemleri gerçekleyen bloklar arası tüm veri yolları her basamağın iki ile ifade edilmesinden dolayı iki katına çıkmaktadır dolayısıyla devrenin kapladığı alan da artmaktadır.

Eliptik eğri işlemlerinde çalışma hızını önemli ölçüde etkileyen modüler çarpma işlemleri için Montgomery modüler çarpması kullanılmıştır. Bu amaçla [7]' de verilen algoritma $GF(3^{97})$ için özelleştirilmiş ve uygun devre ardışıl hücre dizileri şeklinde tasarlanmıştır. Bu tasarımın en önemli katkısı çarpma işlemi sırasında modüler indirgeme işlemini de yapmasıdır böylece ayrı bir modüler indirgeme bloğu tasarımına gerek kalmamıştır. Bu hem devrenin kapladığı alan hem de iş yükü anlamında olumlu bir katkıdır. Ardışıl hücre dizisi şeklindeki tasarımda devre paralel ve tekrar eden hücrelerden oluşmaktadır. Tekrar eden hücre sayısı arttırılarak daha

yüksek dereceli sonlu alanlarda çalışılabilir. Devre bu yönüyle basit değişikliklerle genişletilebilir ve tekrar kullanılabilir bir yapıdadır. Ayrıca paralel yapı nedeniyle çarpma devresinin en uzun yol gecikmesi sadece bir hücrenin yol gecikmesi kadardır. Hücre sayısının artması başka bir deyişle çalışılan alanın genişlemesi devrenin en uzun yol gecikmesinin değişmesine neden olmayacaktır.

Elde edilen tüm bu sonuçlar geliştirmeye açıktır. Eliptik eğri işlemcisi içerisinde yer alan alt blokların alan performansları zaman performasları düşürülerek arttırılabilir. Bununla ilgili öneriler her alt bloğun tasarımı sırasında verilmiştir. Ayrıca farklı sentezleyiciler ve farklı FPGA aileleriyle daha farklı alan-zaman performans sonuçları elde etmek de mümkündür.

KAYNAKLAR

[1] **Koblitz N**., 1987. Elliptic Curve Cryptosystem. *Mathematics of Computation*, Vol. **48**: 203-209.

[2] **Miller V.,** 1985. Uses of Elliptic Curves in Cryptography. *In H. C. Willams, Editor, Advances in Cryptology: CRYPTO'85, volume 218 of Lectures Notes in Computer Science, Springer-Verlag*. 417-426.

[3] **Rivest R. L., Shamir A. ve Adleman L.,** 1978. A Method for Obtaining Digital Signatures and Public-Key Cryptosystems. *Communications of the ACM,* **21(2)**:120-126, February 1978.

[4] **NESSIE Consortium**, 2003. NESSIE Security Report, *Technical report NESSIE.* http://www.cosic.esat.kuleuven.ac.be/ nessie/ deliverables/ D20-v2.pdf

[5] **Jansma N., Arrendondo B.,** 2004. Performance Comparison of Elliptic Curve and RSA Digital Signatures, *Technical Report from Sun Microsystems Laboratories*. http://research.sun.com/projects/crypto/ CHES_2004.pdf.

[6] **Bailey, D.V. ve Paar**, C., 1998. Optimal Extension Fields for Fast Arithmetic in Public-Key Algorithms. *In H. Krawczyk, Editor, Advanced in Cryptology - CRYPTO'98, volume LNCS 1462*, *Springer-Verlag*. Berlin, Germany, 472-485.

[7] **Montgomery P.,** 1985. Modular multiplication without trial division. *Mathematics of Computation,* Vol. **44**:519-521.

[8] **Stinson D.R.,** Cryptography Theory and Practice, Second Edition, Chapman & Hall/CRC, CRC Press Company.

[9] **Menezes A. J., Oorschot P. C. V. ve Vanstone S. A.,** 1997. Handbook of Applied Cryptography. CRC Press, Boca Raton, Florida, USA.

[10] **Stallings W.,** 1999. Cryptography and Network Security. Prentice Hall, Upper Saddle River, New Jersey, USA, 2nd edition.

[11] **Schneier B.,** 1996. Applied Cryptography. John Wiley & Sons Inc., New York, New York, USA, 2nd edition

[12] **FIPS 46-3,** Data Encryption Standard, *National Institute of Standards and Technology (NIST).*

[13] **FIPS 197**, Advanced Encryption Standard, *National Institute of Standards and Technology (NIST).*

[14] **Diffie W. ve Hellman M. E.,** 1976. New directions in cryptography. *IEEE Transactions on Information Theory*, **IT-22**:644-654.

[15] **Ibrahim M.,** 2007. Elliptic Curve Cryptography, *Lecture in Istanbul Technical University, İstanbul.*

[16] **Murphy, T.**, Course 373: Finite Fields, *University of Dublin School of Mathematics.*

[17] **ElGamal T.,** 1985. A Public Key Cryptosystem and a signature scheme based on discrete logarithms. *In G. R. Blakley and D. Chaurn, editors, Advances in Cryptology: CRYPTO'84, Volume 196 of Lecture Notes in Computer Science,Springer-Verlag*, 1985. pages 10-18.

[18] **Mentens N., Ors S. B., Preneel B., ve Vandewalle J**., 2004 An FPGA implementation of an elliptic curve processor over GF (2^m). *In D. Garrett, C. Zukowski, and J. Lach, editors, Proceedings of the Great Lakes Symposium on VLSI (GLSVLSI), , Boston, MA, USA, April 26-28 2004. ACM Press.* Pages 454-457.

[19] **Ors S. B., Batina L., Preneel B., ve Vandewalle J.,** 2003 Hardware implementation of an elliptic curve processor over GF(p). *In IEEE 14th International Conference on Application-specific Systems, Architectures and Processors (ASAP), The Hague, The Netherlands, June 24-26 2003*. pages 433-443.

[20] **Lidl, R., Niederreiter, H.,** 1997. Finite Fields, *volume 20 of Encyclopedia of Mathematics and its Applications, Second edition. Cambridge University Press, Cambridge, UK.*

[21] **Bailey, D.V. ve Paar, C.,** 2001. Efficient arithmetic in finite field extensions with application in elliptic curve cryptography. *J. Cryptology* **14(3)**, 153–176.

[22] **Bertoni G., Guajardo J., Kumar S., Orlando G., Paar C. ve Wollinger T.,** 2003. Efficient GF(p^m) Arithmetic Architectures for Cryptographic Applications, *Topics in Cryptology – CT-RSA, LNCS 2612, , San Francisco, CA, USA, April Springer-Verlag Berlin.* pp. 158-175.

[23] **Ors S. B., Batina L., Preneel B., and Vandewalle J.**, 2003. Hardware implementation of a Montgomery modular multiplier in a systolic array. *In The The 10th Reconfigurable Architectures Workshop (RAW), Nice, France, April 22 2003.*

[24] **Guajardo J.**, **Paar C.,** 1997. Efficient Algorithms for Elliptic Curve Cryptosystems, *Advances in Cryptology - CRYPTO'97, LNCS 1294, 1997. Springer-Verlag Berlin Heidelberg 1997.* pp.342-356.

[25] **Karatsuba, A., Ofman, Y.,** 1963. Multiplication of Multidigit Numbers on Automata. *Soviet Physics –Doklady (English translation)* **7**, 595–596

[26] **Weimerskirch, A., Paar, C.,** 2003. Generalizations of the Karatsuba Algorithm for Polynomail Multiplication. Technical report, *Ruhr-University Bochum, Germany. http://www.crypto.rub.de/Publikationen/ texte/kaweb.pdf*

[27] **Washington L. C.,** Elliptic curves : number theory and cryptography, Boca Raton : Chapman & Hall/CRC, c2003

[28] **Menezes A. J.,** 1993. Elliptic Curve Public Key Cryptosystems. Kluwer Academic Publishers,

[29] **Mugino, S.,** 1997. Elliptic Curve Cryptosystems, Msc.Thesis, McGill University, Montreal

[30] **Harrison K., Page D. and Smart N.P.,** 2002. Software Implementation of Finite Fields of Characteristic Three, for use in Pairing-Based Cryptosystems. LMS J. Computational Math., pages 181-193

[31] **Walter, C.D.,** 1999. Montgomery exponentiation needs no final subtraction. *Electronic letters,* **35(21)**:1831–1832.

[32] **Dhem J.-F., Koeune F., Leroux P.-A., Mestre P., Quisquater J.-J., ve Willems J.-L.,** 1998. A practical implementation of the timing attack, Proc. *CARDIS 1998, Smart Card Research and Advanced Applications (J.-J. Quisquater and B. Schneier, eds.), LNCS, Springer-Verlag*

[33] **Koç Ç.K., Acar. T.,** 1998. Montgomery Multiplication in $GF(2^k)$, *Designs, Codes and Cryptography,* **14(1)**, 57-69,April 1998

[34] **Koblitz N.,** 1994. A Course in Number Theory and Cryptography, *volume 114 of Graduate text in mathematics. Springer-Verlag, Berlin, Germany, second edition*

[35] **Cohen H.,** 1995. A Course in Computational Algebraic Number Theory, *Springer-Verlag, New York, USA*

[36] **Certicom Report,** 2004. An Elliptic Curve Cryptography (ECC) Primer, Jun.

[37] **Page D., Smart N.P.,** 2003. Hardware Implementation of Finite Field of Characteristic Three. *B.S. Kaliski Jr. Et al.(Eds): CHES 2002, LNCS 2523, pp. 529-539, 2003. Springer-Verlag Berlin Heidelberg 2003.*

[38] **Kerins T., ve diğerleri,** 2005. Efficient Hardware for the Tate Pairing Calculation in Characteristic Three. *LNCS, Cryptographic Hardware and Embedded Systems, Springer Berlin Heidelberg 2005.* pp. 412-426.

ÖZGEÇMİŞ

İlker Yavuz 6 Ağustos 1981'de İzmir'de doğdu. 1999 yılında İzmir Atatürk Lisesinden mezun oldu. 2004 yılında İstanbul Teknik Üniversitesi, Elektronik ve Haberleşme Mühendisliği bölümünden mezun olup aynı yıl İstanbul Teknik Üniversitesi Fen Bilimleri Enstitüsü, Elektronik Mühendisliği programında Yüksek Lisans eğitimine başladı. 2004 yılından bu yana TÜBİTAK Ulusal Elektronik ve Kriptoloji Araştırma Enstitüsünde (UEKAE) araştırmacı olarak çalışmaktadır.