KOCAELİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

ELEKTRONİK VE HABERLEŞME MÜHENDİSLİĞİ
ANABİLİM DALI

YÜKSEK LİSANS TEZİ

BLUETOOTH DÜŞÜK ENERJİ İLETİŞİMİNDE GELİŞMİŞ ŞİFRELEME STANDARDI (AES) ŞİFRELEME, UYGULAMA VE KARMAŞIKLIK ANALİZİ

TUVSHİNJARGAL ULZİİUTGA

KOCAELİ 2017

KOCAELİ ÜNİVERSİTESİ

FEN BİLİMLERİ ENSTİTÜSÜ

ELEKTRONİK VE HABERLEŞME MÜHENDİSLİĞİ

ANABİLİM DALI

YÜKSEK LİSANS TEZİ

BLUETOOTH DÜŞÜK ENERJİ İLETİŞİMİNDE GELİŞMİŞ ŞİFRELEME STANDARDI (AES) ŞİFRELEME, UYGULAMA VE KARMAŞIKLIK ANALİZİ

TUVSHİNJARGAL ULZİİUTGA

Prof. Dr. Sarp ERTÜRK
Danışman, Kocaeli Üniversitesi .....

Yrd. Doç. Dr. Osman BÜYÜK
Jüri Üyesi, Kocaeli Üniversitesi .....

Yrd. Doç. Dr. Metin VARAL
Jüri Üyesi, Sakarya Üniversitesi .....

Tezin Savunulduğu Tarih: 13.03.2017

ÖNSÖZ VE TEŞEKKÜR

Tezde Bluetooth düşük güç veri haberleşmede gelişmiş şifreleme standardı uygulanmıştır. Tezde Bluetooth düşük gücün ne olduğunu, onun üzerinde gelişmiş şifreleme standardı nasıl uygulandığını ve ne gibi donanımlar kullanıldığını ve donanımların özellikleri anlatılır. Bununla birlikte kullanılan gelişmiş şifreleme standardı yöntem uygulamada nasıl etkilendiğini zaman ve güvenlik açısından açıklanmaktadır.

Uygulama kısmında çok yardımı dokunan değerli danışman hocam Prof.Dr. Sarp Ertürk ve yazı kısmında hep yanımda kalarak göz kulak olup yardım eden çok değerli arkadaşlarım Bolderdene Chinbat, Muhammet İkbal Deniz ve sınıf arkadaş Muhammet Erdemlere gönlümden teşekkür ederim. Aynı zamanda Moğolistan'dan her türlü desteğini esirgemeyen aileme teşekkürlerimi sunuyorum.

Mart – 2017 Tuvshinjargal ULZİİUTGA

İÇİNDEKİLER

ŞEKİLLER DİZİNİ

TABLOLAR DİZİNİ

SİMGELER VE KISALTMALAR DİZİNİ

InvMixColumns()	: Ters çipher'de kullanılan mixcolumns()'un ters işlemidir
InvShiftRows()	: Ters çipher'de kullanılan ShifRows()'un ters işlemidir
InvSubBytes()	: Ters çipher'de kullanılan SubBytes()'un ters işlemidir
MixColumns()	: Çipher'de kullanılan işlemidir. Durum matrisin her sütunları bir birinden bağımsız karışırılan işlemidir
Nb	: Durum matrisi içeren sütunların sayısı. Nb=4
Nk	: Çipher anahtarı içeren 32 bit kelime (32bit word). Nk = 4, 6 veya 8
Nr	: Nk ve Nb'den oluşan sabit tur sayısı. Nr = 10,12 veya 14
Rcon[]	: Sütunları karıştırmak için kullanılan sabit matris
ShiftRows()	: Çipher'de kullanılan fonksiyondur. Durum matrisin son üç satır satıra sıra numaralarına göre farklı aralıkla kaydıran Çipher'in bir dönüşümü
SubBytes()	: Durum matrisin elamanlar kendi değerleri lineer olmayan S-kutu denen dizindeki elemanlarla takas edilen Çipher'in bir dönüşümü
XOR	: Xor işlemi

Kısaltmalar

ADC	: Analog to Digital Converter (Analog Dijital Dönüştürücü)
ADT	: Android Development Tools (Android Geliştirme Araçları)
AES	: Advanced Encryption Standart (Gelişmiş Şifreleme Standardı)
AHB	: Advanced High Performance Bus (En Yüksek Hıza Sahip İletişim Yolu)
APB	: Advanced Peripheral Bus (Gelişmiş Çevresel İletişim Yolu)
API	: Application Programming Interface (Uygulama Yazılım Arayüz
APK	: Android Package Kit (Android Paket Kiti)
BLE	: Bluetooth Low Energy (Bluetooth Düşük Enerji)
BT	: Bluetooth
CMSIS	: Cortex Microcontroller Software Interface Standard (Korteks Mikro Denetleyici Yazılım Arayüz Standardı)
DAC	: Digital to Analog Converter (Dijital Analog Dönüştürücü)
DSP	: Digital Signal Processing (Dijital Sinyal İşleme)
FPU	: Floating point unit (Kayan Nokta Birimi)
GF	: Galois Field (Galois Alan)
GPIO	: General Purpose Input/Output (Genel Amaçlı Giriş/çıkış)
IDE	: Integrated Development Environment (Entegre Geliştirme Ortamı)

ISM	: Industrial, Science and Medical (Endüstriyel, Bilim ve Tıp)
PWM	: Pulse Width Modulation (Darbe Genişlik Modülasyonu)
RAM	: Random Access Memory (Rastgele Erişim Belleği)
RTC	: Real Time Clock (Gerçek Zamanlı Saat)
SRAM	: Static Random Access Memory (Statik Rastgele Erişim Belleği)
USART	: Universal Synchronous Asynchronous Receiver Transmitter (Evrensel Senkron/Asenkron Alıcı Verici)
XML	: Extensible Markup Language (Genişletilebilir İşaretleme Dili)

BLUETOOTH DÜŞÜK ENERJİ İLETİŞİMİNDE GELİŞMİŞ ŞİFRELEME STANDARDI (AES) ŞİFRELEME, UYGULAMA VE KARMAŞIKLIK ANALİZİ

ÖZET

Bu tezi STM32F407 Discovery geliştirme board ve akıllı mobil telefon arasında Bluetooth düşük enerji bağlantı kurarak aktarılan veriler üzerinde gelişmiş şifreleme standardı uygulanıp anlatmaktadır. Üstelik Bluetooth düşük enerji üzerinde aktarılan veriler üzerinde uygulanan gelişmiş şifreleme standardının ne kadar zaman harcandığını, yani zaman karmaşıklığından bahsetmektedir. Yani, teori olarak 16 bayt altında olan veriler aynı zaman içinde şifrelenip şifrelenmediğini kontrol etmekte olup verilerin miktarı arttıkça ne kadar zaman kaybettiğini belirlenip analiz yapılmaktadır. Verileri kısa mesafede göndermek için yapılan ve açık olan Bluetooth düşük enerji protokolü dünyanın her yerinde kullanılmakta ve piyasada yeni çıktığı için hızla gelişmekte olan bir teknolojidir. Örneğin, Bluetooth düşük enerjili hoparlör, insanların sağlığı için yardımı dokunan kalp atış ölçümü ve her türlü veri gönderilmesi için uygulanmıştır. Dolayısıyla Bluetooth düşük enerji, kullanılan donanımlar üzerinde şifreleme uygulanırsa veriler daha güvenli bir şekilde aktarılabilmekte ve herhangi biri tarafından kolayca bozulmasını önlemektedir. Aynı zamanda uygulanan gelişmiş şifreleme standardı simetrik olduğu için onu kullanarak şifrelenen verinin çözülmesi daha zordur ve bu şifreleme algoritması Birleşik Amerika devletinde şifreleme standardı tarafından kabul edilmiş bir algoritma olduğu için güvenirliği her açıdan sağlanmıştır. Veri şifreleme, veri haberleşmesinde kullanılarak daha gelişmiş hale gelmekte olup aynı anda veri şifrelemede harcanan zamanı da azaltmaktadır. Bu yüzden düşük güçlü BLE'de AES uygulanıp onun üzerinde analiz yapılmaktadır.

Anahtar Kelimeler: AES, Android, BLE, STM32F4, Veri Şifreleme.

IMPLEMENTATION AND COMPLEXITY ANALYSIS OF EMBEDDED ADVANCED ENCRYPTION STANDARD (AES) IN BLUETOOTH LOW ENERGY COMMUNICATION

ABSTRACT

This thesis describes and applies the advanced encryption standard on data transmitted by establishing Bluetooth low energy connection between the STM32F407 Discovery development board and the smart mobile phone. In addition, it mentions the time spent by the advanced encryption standard applied on the data transmitted through Bluetooth low energy, i.e. the time complexity. Thus, theoretically, it is checked whether the data under 16 Bytes is encrypted during the same time or not and also the duration of time wasted as the data amount increases is analysed. The Bluetooth low energy protocol, which is open and ready to send data over short distances, is being used all over the world and is a rapidly evolving technology since it was recently introduced to the market. For instance, the Bluetooth low-power speaker has been applied to send heartbeat measurements and any data to help people's health. Therefore, if encryption is applied on the used hardware in the Bluetooth low energy, the data can be transmitted more securely and its easily corruption by anyone is prevented. At the same time, since the advanced encryption standard applied is symmetric, it is more difficult to decrypt the encrypted data using it. Moreover, since this encryption algorithm is an algorithm adopted by the United States government's encryption standard, reliability is provided at every angle. Data encryption is becoming more sophisticated by using it in data communication and at the same time it is reducing the time spent encrypting the data. Therefore, analysis was done by applying AES on low-power BLE.

Keywords: AES, Android, BLE, STM32F4, Data Encryption.

GİRİŞ

Günden güne artmış olan verileri bir yandan diğer yana transfer etmekte olan her türlü iletişim protokolleri gözlenmektedir. Bunlardan kablosuz iletişim teknoloji hayatımıza daha fazla kullanılmaktadır. Günümüzde kullanılan kablosuz veri aktarma teknoloji cep telefonları veya diğer mobil terminaller aracılığıyla hiçbir arabirim bağlantısına ihtiyaç duymadan bir birine erişimlerini sağlayan iletişim standardı BLUETOOTH'dur. Her gün yeni çıkan telefon veya çıkmış olan telefonlarda BLUETOOTH sabit yerleştirilmiştir ve bunu istediği mikro denetleyici üzerinde uygulanarak kablosuz haberleşme sistem kurabilmektedir. BLUETOOTH ilk 1994 yılda geliştirilmiş ve 2,400 GHz – 2,483GHz arasında ISM bant kullanılmaktadır. Frekans atlatmalı yöntemi ile veri aktarma yapılır ve bu veriyi küçük parçalar içine böler, parçalar paket olarak adlandırılır. BLUETOOTH cihazlar arasında rastgele atlatmalı saniyede en az 1600 defa değişmekte ve bu Bluetooth ağlarda diğer kablosuz 2,4Ghz çalışan cihazlardan daha yüksek ayrıcalıkta girişim verir. 2010 yılından itibaren BLUETOOTH kendine daha farklı özellikleri taşınmaya başlamıştır ve bunun adı "BLUETOOTH düşük güç"(BLE) olarak tanıtılmıştır. BLE önceki nesile göre en az 20 kat enerji verimliliğine sahiptir. Bazı uygulamalarda enerji verimliliği önceki nesile göre 100 kata çıkabilir. BLE'nin taşıdığı özelliklerden en önemli olanları, akıllı ana denetleyici, ayarlanabilir ultra-düşük ve ayarlanabilir veri uzunluktur. Projede BLE üzerinden aktarılan verilerde gelişmiş şifreleme standardı (AES) uygulanarak daha güvenli bir şekilde veriler aktarılmaktadır. Bu projede bir tarafta STM32F4 geliştirme board ile TinySine BLE öteki tarafta android akıllı cihaz kullanılmıştır.

Yani, iki cihaz arasındaki BLE üzerinden gönderilen verileri AES ile şifrelenip haberleşmektedir. AES ve gelişmiş şifreleme standardı ise 2001 yılında Ulusal Standartlar ve Teknoloji Enstitüsü (NIST) tarafından elektronik verileri şifrelenmesi standardı olmuştur ve Belçika'nın kriptografçıları Joan Daemen ve Vincent Rijmen tarafından icat edilmiştir [1]. AES aynı zamanda Rijindael olarak da adlandırılır. AES önceden belirlenmiş anahtar metninin yardımıyla şifrelenme yapılır. Bununla birlikte

şifrelenme yapılırken başlangıçta verilen metin anahtar metin ile birkaç tane tur (anahtar metninin uzunlukla ilgili) farklı fonksiyonlarla işlenerek şifrelendiği için de-şifrelenme zamanı çok daha artırır. Yani, istenmediği taraflardan bilgiye ulaşmak zordur. Dolayısıyla, düşük güç ile çalışan BLE kablosuz haberleşmede AES uygulanıp, aktarılan verilerde şifrelenme yapıldığı zaman, ne kadar zaman harcandığını ve gerçek sistemde ne kadar gecikme oluşturulduğunu inceleyip zaman üzerinde analiz etmişti. AES şifrelenme yönteminde bir kerede 16 baytlık veri veya ondan küçük verilerin şifrelenmesi için harcanan zaman yaklaşık olarak aynıdır. Çünkü 1 ve 16 bayt arasındaki verilerin şifrelenme süresi aynıdır. CPU'nun aynı anda birkaç tane farklı işlem yapması biraz daha yavaşlanmasına neden olur. Eğer veri 16 bayttan büyük olursa bölünüp şifrelenmektedir. Yani, en büyük parça 16 baytlık bloklar halinde ayrı ayrı şifrelenmektedir. Projenin birinci bölümde AES şifrelenme ve de-şifrelenme kodları yazılmış olan STM32F407VG geliştirme tahtasının hakkında bilgi verilmektedir. STM32F407VG geliştirme tahtası ise TinySine BLE modülden gelen verileri kendi USART girişi ile bağlanıp haberleşmektedir. İkinci bölümde TinySine BLE hakkında bilgi verilmektedir. TinySine BLE modülü, USART seri haberleşmesinin yardımıyla kendine gelen ve kendinden giden verileri kontrol etmekte ve seri haberleşmesinin yardımıyla AT kumandası ile BLE ayarı yapılmaktadır. Üçüncü bölümde, BLE üzerinden aktarılan verilerin şifrelenmesi için kullanılan AES hakkında bilgi verilmektedir. Dördüncü bölümde BLE haberleşmesinin hakkında ayrıntı bilgi verilmektedir. Beşinci bölümde Android cihazlarda BLE'nin nasıl gerçekleştiğini ve onun özellikleri anlatılmaktadır. Altıncı bölümde projede, kullanılan aygıtlar arasındaki BLE haberleşme üzerinden AES şifreleme yöntemi uygulandıktan sonraki zaman analizi yapılmaktadır.

1. STM32F407, KORTEKS-M4 TABANLI GÖMÜLÜ SİSTEM

Bu bölümde STM32F407 mikro denetleyiciyle ilgili bazı genel bilgiler, “Discovery” tahtasının yazılım çerçevesi açıklanmaktadır. STM32F4 mikro denetleyicinin geliştirme tahtasıdır. Alt bölümlerde STM32F4’ün her bir parçasının genel içerikleri ve özelliklerine değinilmektedir.

1.1. Mikro Denetleyici

Mikro denetleyiciler genellikle tek çipli bilgisayar olarak bilinir. Mikro denetleyicinin içerisinde mikro işlemci, ”periferik” denilen paralel giriş/çıkışlar, seri giriş/çıkışlar, zamanlayıcılar, analogdan dijital dönüştürücü ve özel alt sistem fonksiyonlar vardır. Mikro denetleyicinin yardımıyla her türlü sistemi kontrol edip, onun üzerinde analiz yapılabilmektedir.

1.1.1. STM32F genel bakış

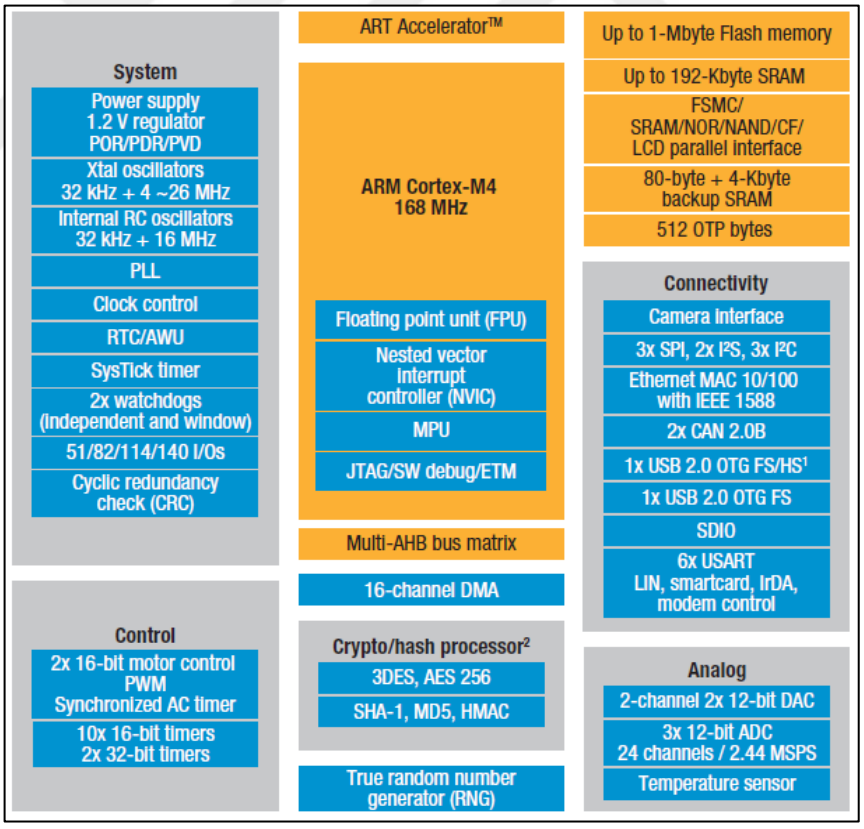
Korteks-M4 çekirdek tabanlı ST, STM32 ürünlerinin tanıtımını yapar. STM32F4’te 30’dan fazla yeni parça ve STM32F4 serisi ile uyumlu yazılımlar vardır. 168 MHz performansa sahip olan yeni DSP ve FPU talimatları Dijital Sinyal kontrol uygulamasını daha iyi bir seviye atlatmıştır. Şekil 1.1’de STM32F4 blok diyagramı [2] gösterilmektedir. Korteks-M4 çekirdek tabanlı mikro denetleyicilerden projede STM32F407xx serisi kullanılmaktadır. Korteks-M4 32-bit RISC çekirdek 168MHz’e kadar frekansa ulaşır. Korteks-M4 çekirdeği kendine kayan nokta birimi (FPU) sahip olduğundan tüm veri işleme talimatları ve veri türlerini 32-bit içinde daha fazla doğrulukla desteklenir. Aynı zamanda güvenlik arttırılması uğruna DSP birleşik talimatları ve bellek koruma birimi uygulanmıştır.

FPU ile Korteks-M4 çekirdek bundan sonra Korteks-M4F şeklinde ifade edilecektir. STM32F407xx ailesi, iki APB, üç AHB ve 32-bit çokluk-AHB matris yollar üzerinde giriş/çıkış ve 22 periferik ile bağlanıp bununla birlikte Flash bellek 1M bayttan, SRAM

192K bayttan, yedek SRAM 4K bayttan fazla olup çok hızlı gömülü bellekleri içermektedir.

Tüm cihazlar 3 tane 12-bit ADC, 2 tane DAC, düşük güçlü RTC, 12 tane genel amaçlı 16-bit zamanlayıcı içerisinde motor kontrol için 2 tane PWM zamanlayıcı ve rasgele sayı üretici (RNG) kapsamaktadır. Ek olarak, standardı ve ileri iletişim arabirimleri içeren özellikleri aşağıda gösterilmektedir.

- 2 tane I2C
- 3 tane SPI ve çift yönlü I2S
- 4 tane USART ve ek olarak 2 UART
- USB OTG tam hızla
- 2 CAN
- SDIO/MMC arabirim
- Ethernet ve kamera arabirim yalnızca STM32F407xx cihazlarda mevcut [2]



Şekil 1.1. STM32F4 blok diyagram [2]

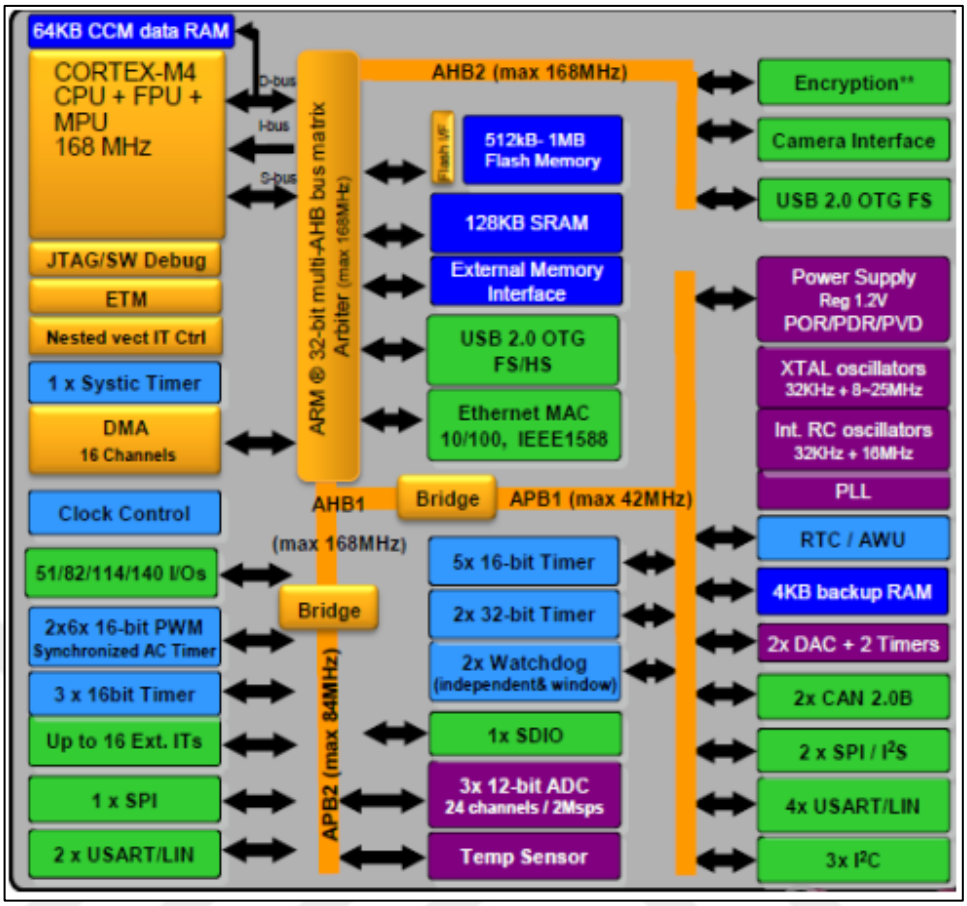
Yeni gelişmiş periferikler SDIO kapsamaktadır. Geliştirilmiş esnek statik hafıza kontrolü (FSMC) ve CMOS sensörler için kamera arabirimi vardır. STM32F407xx

ailesi -40℃ - 105℃ arasındaki sıcaklıkta çalışmaktadır. 1,8V – 3,6V aralığında çalışan güç kaynağa sahiptir. Besleme gerilim 1,7V'dan fazla olup aynı zamanda çalışma sıcaklığı 0'dan 70℃ aralığında ise ters sıfırlama sinyali PDR_ON uygulanmaktadır. STM32F407xx ailesi 64 - 176 bacak arasında değişen cihazları sunabilmektedir. STM32F407xx seçilen cihazda dahil olan periferik aracı uygun olarak tamamlamaktadır. Bu özellikler sayesinde STM32F407xx mikro denetleyici ailesi daha geniş uygulamalarda çalışabilmeyi sağlamaktadır. Aşağıda örnek olarak kullanılan uygulamalar gösterilmektedir.

- Motor sürücü ve kontrol uygulamaları
- Tıp ekipmanları
- Endüstriyel uygulamalar: PLC, invertörler, devre kesiciler
- Yazıcılar ve tarayıcılar
- Güvenlik alarm sistemleri vb. [3]

1.1.2. STM32F407 mikro denetleyicinin bileşenleri

Bu bölümde STM32F407'un giriş, çıkış ve onun içinde olan diğer parçaların bağlantı kısmı anlatılmaktadır.STM32F407'un parçalar AHB ve APB yolu üzerinde bağlanmıştır ve hangi yola bağladığına göre her bir parçanın hızı belirtir. Mikro denetleyicinin çekirdeği, hafızası, giriş/çıkışları Şekil 1.2'de STM32F4xx blok diyagramda gösterilmektedir. Blok diyagramda her bir giriş/çıkış bir birinden bağımsız olarak çalıştığı gösterilmektedir. Aynı zamanda giriş/çıkışların çalışma hızı, çalışma görevlerine göre farklı olduğu görülebilmektedir.



Şekil 1.2. STM32F4xx Blok diyagram [3]

1.1.3. ARM korteks-m4 çekirdekli gömülü flash ve sram

ARM Korteks-M4F işlemcisi gömülü sistemler için ARM işlemcisinin en son neslidir. Bu işlemcinin geliştirilmesinin ana amacı ucuz bir platform üzerine MCU uygulanarak azaltılmış bacak sayısı ve düşük güç tüketimi ile kesmeler için ileri tepki veren olağanüstü bir performans sağlanmaktadır. ARM Korteks-M4F 32-bit RISC işlemcisi ayrıcalıklı bir kod verimliliği sunuyor. Bu mimari, standart mikro denetleyici uygulamalarının yanı sıra DSP fonksiyonlarını da içinde barındıran bir mimaridir. İşlemci DSP birleşik talimatları desteklediği için karmaşık algoritmanın yürütülmesi ve verimli sinyal işlenmesi mümkündür. DSP, sinyallerin sayı dizileri şeklinde temsil edilmesi ve bu sayı dizilerinin, nümerik hesaplama yöntemleri ile dönüştürülmesi (transformasyon) veya işlenmesi ile ilgilidir. DSP sisteminde, analog sinyali artık zaman aralıklarında örneklenir. Örneklenmiş analog sinyal genlikleri bir A/D ile sayısal değerlere çevrilir ve sinyalin sayısal eş değeri, sayısal alanda sayı dizileri halinde, DSP'nin kendisi tarafından işlenir. Ayrıca M4 çekirdeğinde

birde FPU bulunuyor. FPU mikro denetleyici içerisinde yer alan ondalık sayıların hesaplanması için ayrılmış bir bölümdür. DSP kütüphanesi kullanılırsa mikro işlemci üzerinde gidilen işlemlerin daha hızlı çalışmasını sağlar.

1.1.4. Gömülü flash hafıza

Programları ve verileri depolamak için STM32F40x cihazlar tarafından mevcut olan 512K bayt veya 1M bayt FLASH hafıza gömülüdür [3].

1.1.5. Gömülü sram

64 KB çekirdekle birleştirilmiş bellek (CCM) ve RAM'a dahil olan 192 KB SRAM vardır. RAM bellek zamanın sıfır(0) bekleme süresinde CPU'ye (okuma veya yazma) erişebilmektedir. 4K bayt yedek SRAM vardır ve bu alana sadece CPU erişebilir.

1.1.6. Çoklu- (AHB) iletişim matrisi

32 bit çoklu-AHB iletişim matrisi bütün baş aletler (CPU, DMA, Ethernet, USBHS) ve yardımcı aletler (Flash bellek, RAM, FSMC, AHB ve APB) ile bağlantı yapılmıştır. Bu nedenle yüksek hızlı çevre birimler aynı anda daha verimli bir şekilde çalışabilmektedir [3].

1.1.7. Zaman ve başlangıç

Varsayılan 16 MHz iç RC osilatör CPU için seçilmektedir. 16 MHz iç RC osilatör çalışma sıcaklık aralığında %1 doğruluk sunulmaktadır. Uygulamaya göre dış 4-26MHz veya RC osilatörleri ayarlanabilmektedir. Bu çalışma frekansı üretici (clock) arızası tespit etmek için gereklidir. Diğer bir deyişle, bir arıza tespit edildiği anda sistem otomatik olarak iç RC osilatöre geri dönmektedir. Eğer arza mevcutsa yazılım kesmesi üretilmektedir. Böylece PLL giriş yardımıyla frekans 168MHz kadar arttırılabilmektedir. AHB, yüksek hızla APB (APB2), düşük hızlı APB (APB1) ve AHB1 olmak üzere üçe ayrılır. Üç AHB iletişim yolunun maksimum frekansı 168 MHz'dır. APB için maksimum frekans 84 MHz'dır. Düşük hızlı APB'nin ulaşılabileceği olan maksimum frekans 42 MHz'dır. Cihazlarda ses sınıf performansı çalıştırabilmesi için PLL (PLLI2S) mevcuttur. Bu nedenle, I2S merkez 8KHz'den 192KHz'e kadar örnekleme frekansları üretebilmektedir [3].

1.1.8. Genel amaçlı giriş/çıkışlar

Genel amaçlı her bir bacak yazılım yardımıyla giriş ya da çıkış olarak ayarlanabilmektedir. Giriş olarak tanımladığı durumda dijital ya da alternatif fonksiyonu olur. Çıkış olarak tanımladığı durumda dijital ya da alternatif fonksiyonu olur ve tanımlanırken açık kolektör ya da it-çek ayarı eklemekte mecburdur. Burada tanımlanan girişte (yukarı-çekme veya aşağı-çekme) ayarı eklenebilir ve bu ayar kullanışa göre değişebilmektedir. Bu tanım çıkışta da geçerlidir. Dijital veya analog alternatif fonksiyonlar çoğunlukla GPIO bacakları paylaşılabilmektedir. Tüm giriş/çıkışlar yüksek akım yeteneklidir. Güç tüketimi, elektromanyetik emisyon ve iç gürültüleri yönetmek için hız ayarı yapılmaktadır. Hızlı giriş/çıkış işleme 84MHz'e kadar geçiş kullanımına izin vermektedir [3].

1.1.9. Senkron seri alıcı, verici iletişim (USART)

STM32F405xx ve STM32F407xx dört adet senkron/asenkron alıcı verici iletişim (USART1, USART2, USART3 ve USART6) ve iki adet asenkron alıcı verici iletişim (UART4 ve UART5) vardır [3]. Bu altı asenkron iletişimin IrDA SIR ENDEC destek, çoklu işlemci iletişim yöntemi, tek telli tek yönlü iletişim yöntemi ve LIN merkez/yardımcı yeteneği vardır [3]. USART1 ve USART6 arabirimleri 10,5Mbit/s hıza kadar iletişim kurabilir ve diğer arabirimleri 5,25Mbit/s hıza kadar iletişim kurabilmektedir. USART1, USART2, USART3 ve USART6 iletişimler CTS ve RTS sinyallerde donanım yönetimi sağlamıştır. Tüm arabirimler DMA tarafından servis edilebilmektedir [3].

Tezde USART1 ve USART2 kullanılmıştır. TinySine BLE aygıttan gelen verileri kayıpsız alabilmesi için USART1'in kesmeyi etkinleştirip kullanması gerekiyor. Yani ana iş çalışırken TinyeSine BLE'ye gelen verileri kayıpsız ve gecikmesiz alınması için kullanılır. Sonra, hem TinySine BLE'den gelen veriler AES ile çözülüp "out" diye tanımlanan matrise atılmakta hem de USART1'e gelen veriler USART2'ye gönderilip aktırılan veriler kullanıcıya görünmektedir.

1.1.10. Dış kesme (EXTI)

Dış kesme kontroller 23 kenar algılayıcı hattından oluşmaktadır. Her hattın bağımsız şekilde (yükselen kenar, düşen kenar veya ikisi) ayarı yapılabilmektedir. Ancak APB2

hattın hızından daha kısa genişliği algılayabilir. 140 GPIO 16 dış kesme hattına bağlanabilir [3]. Bu sinyal dış veya genişletilmiş kesme/olay denetleyicisi (EXT) tarafından oluşur [4].

1.2. STM32F4 Discovery

STM32F4 yüksek performansı özellikleri taşınarak STM32F4 Discovery olarak bulunur. Bir STMF407VGT6 üzerinde çalışabilmektedir ve bunun içerisinde gömülü ST-Link arabirim, ST MEMS dijital ivmeölçer, ST MEMS dijital mikrofon, ses DAC ile D sınıf hoparlör sürücü, Ledler, düğme ve USB OTG mikro-AB bağlantısı vardır [5].

Şekil 1.3. STM32F4 Discovery tahta [5]

1.2.1. STM32F4 özellikleri

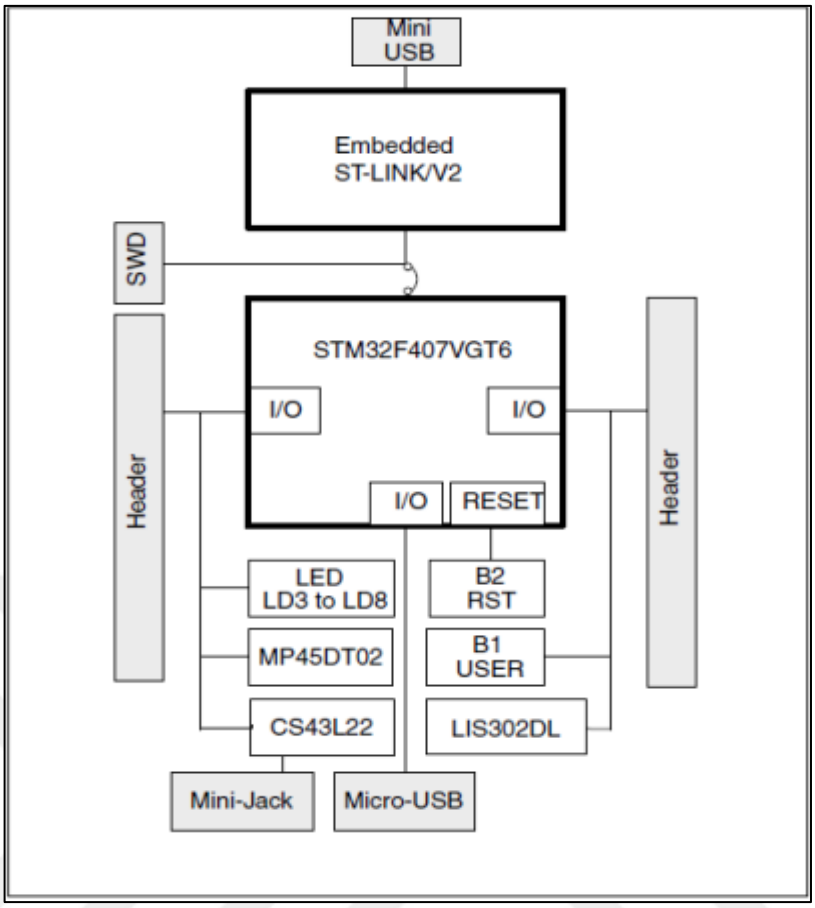
Bu bölümde Discovery tahtanın genel özellikleri ve yeteneklerini değinilmektedir. Aşağıda STM32F407 tahtanın özellikleri gösterilmektedir.

- STM32F407VGT6 mikro denetleyici LQFP100 paketinde bulunup bünyesinde 1MB FLASH bellek ve 192KB RAM'a sahiptir
- Tahta üzerinde ST-Link/V2 anahtar ile ayarlanabilen bağımsız programlama ve hata onarma için SWD ayak vardır

- Tahta güç kaynağı: USB üzerinde veya dış hatta 5V besleme vardır
- Dış uygulama için 3V ve 5V güç kaynağı vardır
- LIS302DL, ST MEMS hareket algılayıcı ve 3 eksenli dijital ivmeölçer çıkışı vardır
- MP45DT02, ST MEMS ses algılayıcı ve çok yönlü dijital mikrofonu vardır
- CS43L22 ve D sınıf hoparlör sürücüsü ile ses DAC vardır
- Sekiz tane Led vardır
  - LD1 (kırmızı/yeşil) USB iletişimi içindir
  - LD2 (kırmızı) 3,3V güç kaynağı içindir
  - Kullanıcı için 4 Led vardır: LD3 (turuncu), LD4 (yeşil), LD5 (kırmızı), LD6 (mavi)
  - 2 USB OTG Led vardır: V hat için LD7 (yeşil) ve LD8 (kırmızı) aşırı akım
- 2 düğme (kullanıcı ve sıfırlama) vardır
- Mikro-AB bağlantı ile USB OTG vardır
- Tahtanın hızlı tanıtımı için LQFP100 giriş/çıkışta uzatma başlıklar vardır

1.2.2. STM32F4 donanımı ve arka planı

STM32F4 Discovery STM32F407VGT6 mikro denetleyici üzerinde bir bitişik olarak tasarlanmıştır. STM32F407VGT6 ve çevre birimler arasındaki bağlantıları şekil 1.4'te gösterilmiştir [5]. Burada donanımsal olarak STLINK/V2, düğme, led, ses DAC, USB, ST MEMS ivmeölçer, ST MEMS mikrofon ve konektörler görülmektedir.



Şekil 1.4. Donanım blok diyagram [5]

1.2.3. STM32F407VGT6 mikro denetleyici

Bu ARM Korteks-M4 32-bit mikro denetleyicinin içerisindeki FPU'de 210 DMIPS 1MB FLASH + 4KB RAM, USB OTG HS/FS, Ethernet, 17 TIMS, 3 ADC, 15 tane iletişim arabirimi vardır. Mikro denetleyicinin özellik ve çevre birimleri bölüm 1.1'de ifade edilmiştir.

1.2.4. STM32F4 Discovery güç kaynağı

Güç kaynağı USB kablosu aracılığıyla bilgisayar tarafından ya da harıcı güç kaynağı (5V) tarafından beslenebilmektedir. Harici güç kaynağı 3V ve 5V'tan korumak amacıyla yapılmış olan D1 ve D2 diyotların ana görevi [5]:

- 3V ve 5V dış güç kaynakları olarak kullanıldığı durumunda diğer uygulama tahta bacak P1 ve P2'ye bağlanır. Bu koşulda 5V ve 3V bacakların teslim ettikleri 5V veya 3V güç tüketimi 100mA'dan daha düşük olması gerekmektedir.

- Giriş güçte 5V uyguladığında ancak USB konektör bağlı olmadığı halde STM32F4Discovery tahta EN-60950-1: 2006+A11/2009 standart ile uyumlu güvenli artık düşük gerilim (SELV) beslenmesi gerekmektedir [6].

1.3. MDK – ARM Mikro Denetleyici Geliştirme Kiti

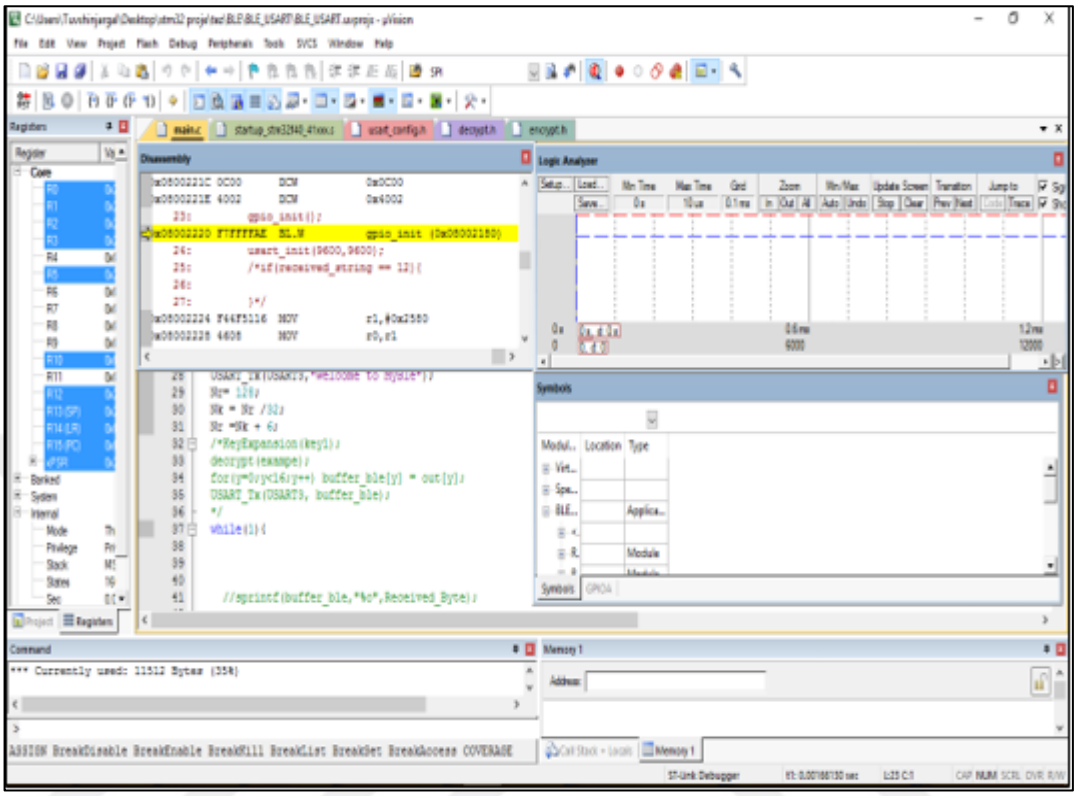
Bu bölümde Keil mikro denetleyici geliştirme kiti anlatılmaktadır. Şekil 1.5'te MDK'nın 5.sürümü geliştirme çerçevesi gösterilmektedir. MDK-ARM mikro denetleyici geliştirme kiti, Korteks-M, Korteks-R4, ARM7 ve ARM9 tabanlı işlemci aygıtlar için yazılım geliştirme ortamıdır [7]. MDK-ARM'de mevcut olan 4 baskı vardır. MDK-Lite, MDK-Korteks-M, MDK-Standart ve MDK-Pro'dur. 4 MB Flash belleğine ücretsiz olarak yazılım yüklenebilir. Özellikleri [8];

- Korteks-M, Korteks-R4, ARM7 ve ARM9 cihazları için tamamen destekleyebilen ortama sahiptir
- ARM C/C++ derleme alet vardır
- uVision4 IDE, hata ayıklayıcı ve simülasyon ortamı vardır
- Keil RTX, gerçek zamanlı işletim sistemi vardır
- TCP/IP ağ takımı, birden fazla protokol ve çeşitli uygulamalar sunar
- USB aygıt ve USB ana bilgisayar host'ları standart sürücü ile sağlanmıştır
- Grafiksel kullanıcı arabirimi ile gömülü sistemler için komple grafiksel kullanıcı arabirim kütüphanesi vardır
- ULINKpro, her yürütülen Korteks-M talimatını kaydeder
- Yazılım yürütülmesi için kapsam bilgileri vardır
- Yürütme profilleri ve performans çözümleyicisi, yazılımın optimizasyonunu etkinleştirir.
- Birçok örnek proje, hızlı bir şekilde MDK-ARM'ı tanımanıza yardımcı olur
- CMSIS Korteks mikro denetleyici yazılım arabirim standart uyumludur

1.3.1. Keil mdk

Keil mdk, ARM tabanlı mikro işlemcilerde bulunan yazılımlardan biridir. Aynı zamanda ARM tabanlı mikro işlemcilerde bulanan Altium, Atollic, IAR yazılımları vardır. Tezde Keil $\mathrm{MDK-ARM}^{\mathrm{TM}}$ yazılım geliştirme ortamı kullanılmıştır. MDK ARM Korteks-M tabanlı mikro işlemciler de gömülü uygulama oluşturması için

yardım eden yazılımdır. MDK 5.sürümü MDK Çekirdek ve Yazılım paketleri diye ayrılmaktadır.



Şekil 1.5. MDK geliştirme çerçevesi

1.3.1.1. Keil mdk çekirdek

MDK çekirdek, Korteks-M tabanlı işlemci mikro denetleyici cihazlarda gömülü uygulama oluşturmak için gereken bileşenleri inşa etme ve uygulamada hata ayıklamayı içerir. Pack yükleyici, MDK çekirdeği istediği zaman yükleyerek yazılım paketleri yönetebilmektedir. Bu toolchain'dan bağımsız halde yeni cihaz desteği ve katman güncellemeleri yapmaktadır.

1.3.1.2. Keil yazılım paketi

Yazılım paketleri mikro denetleyici ailesi için destek ve sistem başlatma kodu ve periferik sürücüleri içermektedir. CMSIS paket Korteks-M çekirdek erişimi, DSP kütüphane ve standart gerçek zamanlı işletim sistemi (RTOS) sağlamaktadır. MDK profesyonel sürümde bulunan katman paket içinde TCP/IP ağ, USB Host, USB cihaz, dosya depolama ve grafiksel kullanıcı arabirim kütüphaneleri bulunmaktadır.

1.3.2. CMSIS

Basit bir yazılımı işlemciye bağlamasını CMSIS sağlar ve bunda periferik arabirim, gerçek zamanlı işletim sistemi ve katmanları içerir. Firma ürettiği mikro denetleyiciler için CMSIS sürücü dosyaları geliştirir. Bunları web sitesinden indirip kendi projenize ekleyebilirsiniz. CMSIS uygulama yazılımı bileşenlerinden aşağıda bahsedilmektedir. CMSIS-çekirdek: Korteks-M işlemci çekirdeği ve periferikler için API tanımlar ve uygun bir sistem başlatma kodu içermektedir.

CMSIS-RTOS: Standard gerçek zamanlı işletim sistemi sağlar. Bu nedenle yazılım şablonları, katman, kütüphaneler ve diğer bileşenler RTOS desteklenen sistemler arasında çalışabilmektedir.

CMSIS-DSP: Dijital sinyal işleme (DSP) için bir kütüphane koleksiyonudur. Bunda 60 fonksiyon üzerinde çeşitli veri türleri, düzeltme noktası ve hassasiyet ayar noktası işlenmektedir [8].

1.3.3. µVision IDE

µVision verimli yazılım geliştirme için sağlam bir editör ve yapı tesisi ile bütünleştirilmiş proje yöneticisidir. µVision IDE tek bir geliştirme ortamı içinde proje yönetimi ve kaynak kod düzenleme yeteneğine sahiptir.

- Entegre edilmiş cihazın veri tabanı**:** ARM tabanlı mikro denetleyici cihazlarda geniş bir dizi için "start-up" kodu ve özel çevre görünümleri sağlar.
- Editör**:** Optimize edilmiş bir iş akışı ile araç çubukları sağlar.
- Kaynak tarayıcı**:** Tüm uygulama sembollerin erişimini sağlar.
- Yapılandırma çubuğu**:** Bit düzeyinde görüntüleme, değiştirme ve belgelendirmeleri hızlı ve kolay bir biçimde yapılmasını sağlar.

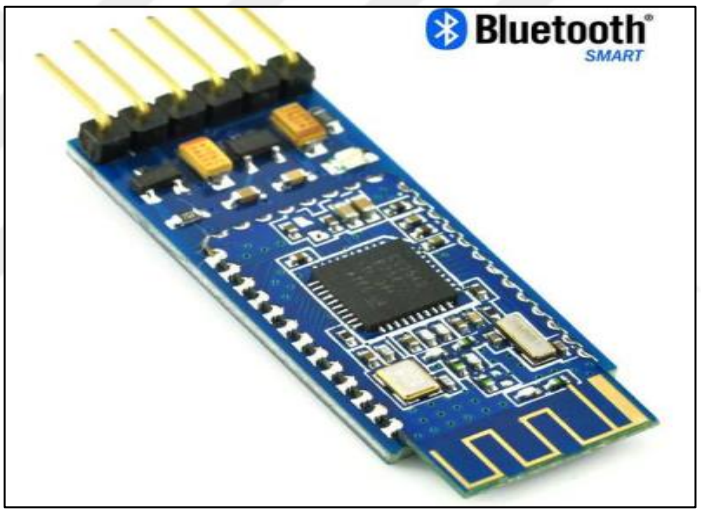
1.3.4. µVision derleyici

µVision Debugger, deneme, doğrulama ve uygulama optimize etmek için tek bir ortamı sağlar ve aşağıda ortam içerikleri açıklanmıştır.

- Sistem Görüntüleyici: Pencere periferik kayıtlarının ayrıntılı bilgilerini görüntülemektedir. İçerik değerleri sürekli hedef donanımı tarafından güncellenir.
- Mantık Analizör: Sinyaller ve değişkenlerin grafik görüntüsünü verir.
- Kod Kapsamı: Uygulamada deneme ve doğrulama için gereken istatistiktir.
- Performans Analizör: Uygulamadaki fonksiyonların yürütme zamanını görüntülemektedir.
- Yürütme Profiler: Her CPU (Ana işlem birimi)'in talimat istatistiklerini kayıt etmektedir. İçerisinde işlem sayımı ve işlem zamanı vardır.
- Call Stack (Çağrı yığını): Pencerede şimdiki değişim ve yerel değişkenleri göstermektedir.

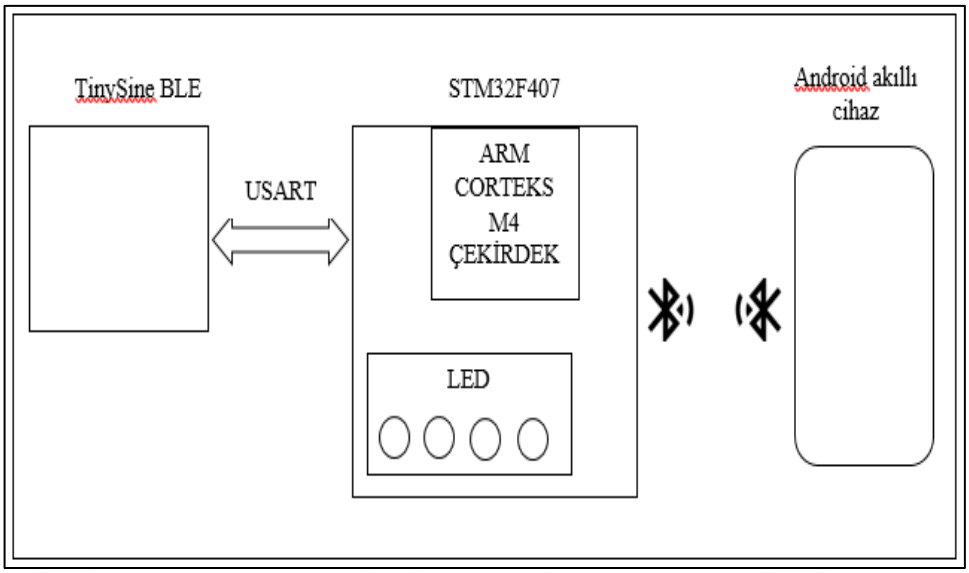
2. TINYSINE BLUETOOTH 4.0 BLE MODÜL

Bu bölümde projede kullanılan Bluetooth düşük gücü destekleyen TinySine Bluetooth 4.0 BLE aygıtından bahsedilecektir. TinySine'nin bağlantısı ve onun ayarları anlatılmaktadır. TinySine Bluetooth 4.0 modül USART veri haberleşmesi üzerinden AT kumandası ile ayarlanabilen bir modüldür. Bu mödül ARM Nuvoton mimarisi ile tek çip üzerinde yapılmıştır. Kullanıcı AT yardımıyla merkez veya yardımcı olarak rol değiştirebilir. BLE'yi destekleyen Android, IOS, Windows ve başka akıllı cihazlarda veri transferi yapılabilmektedir.



Şekil 2.1. TinySine Bluetooth 4.0 [9]

Bu modül bütün dünyadaki akıllı cihazlarla birlikte bir otomatik sistem olarak çalışabilmektedir. Şekil 2.2'de tasarlanan sistemin genel görünümü gösterilmiştir.



Şekil 2.2. Tasarlanan sistemin genel görünümü

Aşağıda TinySine modülünün özellikleri verilmiştir;

- BT sürüm: Bluetooth V4.0 BLE
- Gelen ve giden baytlar sınırsızdır
- Çalışma frekansı: 2,4GHz ISM
- Modülasyon yöntemi: GFSK (Gaussian frekans kaydırmalı anahtarlama)
- RF güç: 0,01mw-5mw, AT komuta (AT+POWE) ile değişebilmektedir.
- Hız: Asenkron – 6K bayt, senkron – 6K bayt
- Güvenlik: Kimlik doğrulama ve şifreleme
- Hizmet: Merkezi ve Çevresel UUID FFE0. FFE1
- Kaynak: +3,3VDC 50mA
- Güç: uyuma modunda 400uA ~ 1,5mA, etkin modda 8,5mA
- Çalışma sıcaklığı: -5℃ ~ 65℃

2.1. Sistemin İşlevi

Sistem uzun zaman aktif olmadığı sürece uyku moduna geçer ve 80'dan fazla karakter USART üzerinden gönderildiği anda "OK+WAKE" karakteri geri bildirim yaptıktan sonra etkin olacaktır. Bu karakterler her hangi bir AT kumandası içermez [9]. Keşfedilebilir modda, USART üzerinden "OK+SLEEP" gönderildiği zaman aygıt uyku moda geçer ve "OK+SLEEP" karakteri geri bildirim yapmaktadır [9].

2.2. TinySine BLE Terminal Komutlar

Varsayılan ayarlar;
Adı: HMSoft; Baud hızı:9600, N, 8,1; Şifre: 000000; Rol: Çevresel rol [10];

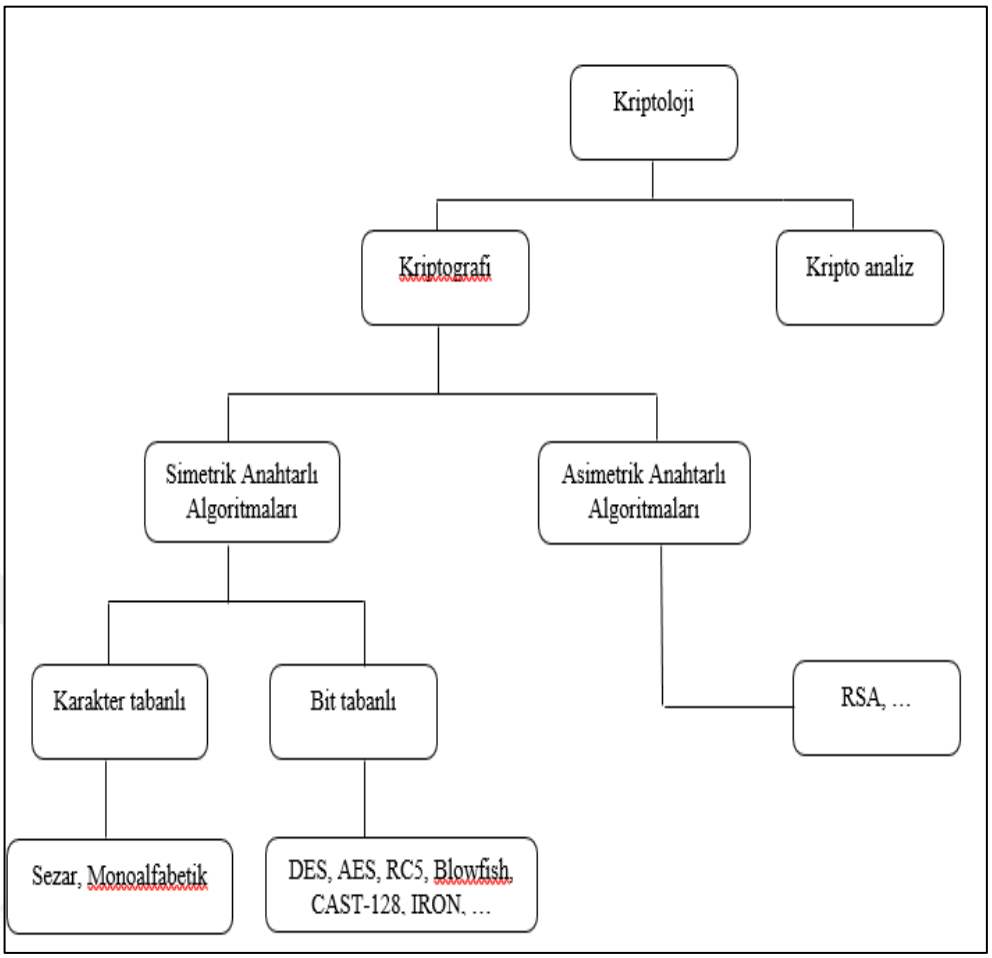
2'inci bölümün başında bahsedildiği gibi TinySine BLE modül USART üzerinde AT kumandası ile ayar yapılır. AT komutlar büyük harf ile yazılmaktadır. Komutlar herhangi bir Bluetooth cihazlıyla bağlantı kurmadan önce işlenir. Modül herhangi bir bluetooth cihazla bağlanıldığı anda "OK+CONN" dönüş yapar ve bağlantı kesildiği anda "OK+LOST" dönüş USART üzerinden gelir. USART üzerinde TinySine BLE modül de yapılmış ayarlar ek kısmında Tablo 2.1'de gösterilmiştir. Ekler kısmında Tablo A.1. [11] AT komutlar'da , proje için yapılmış genel ayarları gösterilmiştir. Reset yapılmadığı zaman USART üzerinden yapılmış olan ayarı modül takip etmektedir.

3. VERİ ŞİFRELEME TEKNİKLERİ KRİPTOGRAFİ

Bu bölümde şifrelemenin ne olduğunu anlatılmaktadır. Aynı zamanda BLE üzerinden aktarılan verilerde kullanılan gelişmiş şifreleme standardın matematiksel teorisi ve gelişmiş şifreleme standardının uygulamada nasıl gerçekleştiği anlatılmaktadır. Kriptografi ve şifreleme binlerce yıldır iletişimin güvenliği için kullanılmaktadır. Kriptoloji gizli mesajlaşma, onaylama, dijital imzalar, elektronik para ve diğer uygulamaların tümüyle ilgili bilim dalıdır. Tarih boyunca en fazla askeri haberleşmede kullanılarak geliştirilmiştir. 1980 yılında ticari güvenlik ve özel iletişim önderliğinde oluşmuştur [12]. Internet icat edildikten sonraki yılların sonundan 1989 yılına kadar halka açık olmamıştır [12].

3.1. Kriptoloji (Cryptology)

Kriptografik, metotların matematiksel temelleriyle ilgilenen bir matematik dalıdır. Geçmişte ilgilenilen kriptografi algoritmaları, algoritmanın gizliliğine dayanmaktaydı. Günümüzde kullanılmakta olan modern ve güçlü şifreleme algoritmaları ise artık gizli değildir. Bu algoritmalar güvenliklerini kullandıkları farklı uzunluk ve yapılardaki anahtarlarla sağlarlar. Bütün modern algoritmalar şifrelemeyi ve şifre çözmeyi kontrol etmek için anahtarları kullanır. Modern kriptoloji genel olarak ikiye ayrılmaktadır, simetrik ve asimetrik kriptoloji. Şekil 3.1 algoritmaların genel tasnifi gösterilmektedir. Asimetrik şifreleme açık anahtar ilkesine dayanmaktadır. Gizlenmek istenen metin herkesin bildiği bir anahtar ile şifrelenir ve ancak gizli anahtarla çözülebilir. Simetrik algoritmalarda ise tek bir gizli anahtar bulunur; şifreleme ve şifre çözme için bu anahtara ihtiyaç duyulur. Simetrik şifreleme, blok şifreleme ve dizi şifreleme olmak üzere iki ana başlığa ayrılır. Bu çalışma kapsamında bir blok şifreleme türü olan Gelişmiş Şifreleme Standardı kullanılmıştır. Sonraki bölümde AES hakkında daha ayrıntılı bilgi verilecektir.



Şekil 3.1. Algoritmaların genel tasnifi

3.2. Gelişmiş Şifreleme Standardı (AES)

Ocak 1997'de NIST, yeni bir şifreleme standardının geliştirilmesi için bir çalışma başlatmıştır. Geliştirilecek yeni şifreleme standardının mevcut şifreleme standardı olan DES'in yerini alması düşünülmüştür. Çünkü DES'in 64 bitlik anahtar uzunluğu, gelişen teknoloji ve artan işlemci hızları karşısında güvenirliğini yitirmeye başlamıştır. NIST tarafından, yeni şifreleme standardını belirlemek amacıyla bir yarışma düzenlenmiş ve Eylül 1997'de algoritmalar için resmi çağrıda bulunmuştur. Ağustos 1998'de 15 adayın algoritmalarının 1.turda değerlendirilmeye alındığı duyurulmuş ve Nisan 1999'da gerçekleşen 2. tur seçimlerinde algoritma sayısı 5'e indirilmiştir. Dört yıl boyunca süren değerlendirme ve eleme süreci sonrasında, Ekim 2000'de sonuç açıklanmış ve NIST, Joan Daemen ve Vincent Rijmen tarafından tasarlanan, Rijndael algoritmasının Gelişmiş Şifreleme Standardı (AES) olarak kullanılacağını ilan edilmiştir [14].

3.2.1. AES matematiksel kavramlar

Rijndeal'daki çeşitli işlemler bayt düzeyinde tanımlanır. Baytlar sonlu alan GF(2^8) elemanlarını temsil etmektedir. Diğer işlemler 4-bayt ile tanımlanır. Alt bölümde AES'de kullanılan temel matematiksel kavramlar anlatılmaktadır.

3.2.1.1. Sonlu alan GF(2^8)

Sonlu alanın elemanları çeşitli şekillerde temsil edilebilir [14]. Her bir asal sayı için sonlu bir alan vardır. Dolayısıyla tüm GF(2^8) temsilleri izomorfik olur. Diğer bir deyişle her bir denklem üzerinde karmaşık bir işlemler dizisi uygulanmış demektir. Bir bayt B, b7b6b5b4b3b2b1b0 bitten oluşup katsayısı {0,1} olan bir polinom ifadesi Denklem (3.1)'deki gibi;

$$b_7x^7+b_6x^6+b_5x^5+b_4x^4+b_3x^3+b_2x^2+b_1x+b_0 \quad (3.1)$$

şeklinde olacaktır.

Örneğin: bayt onaltılık tabanında '57' (01010111 ikili) değer polinom Denklem (3.2)'deki gibi;

$$x^6+x^4+x^2+x+1 \quad (3.2)$$

şeklinde olacaktır.

3.2.1.2. Toplama işlemi

Sonlu alanda iki elemanın toplanması polinomuna karşılık gelen "ekleme" katsayılarının elde edilmesidir. Toplama işlemi XOR ile gerçekleştirilir (ifade: $\oplus$). Böylece iki elemanda gerçekleşen işlemin sonucu: $1 \oplus 1 = 0$, $1 \oplus 0 = 1$, $1 \oplus 0 = 1$, $0 \oplus 0 = 0$. Sonuçta polinomlarda çıkarma işlemi, polinomların toplaması ile aynıdır;
Örneğin: '57' + '83' = 'D4', yani '57' XOR '83' = 'D4'

polinomal gösterim: $(x^6+ x^4+x^2+x+1) + (x^7+x+1) = x^7+x^6+x^4+x^2$.

İkili gösterim: "01010111" + "10000011" = "11010100". Açıkçası, toplama bayt seviyesinde işlenen basit bir bitsel XOR işlemidir.

3.2.1.3. Çarpma işlemi

Polinom temsilinde, iki elamanın çarpılmasından (ifade: ·) GF(2^8) polinoma karşılık gelen irreducible (indirgenemez) ikili polinomin 8'inci dereceden mutlak değer alarak elde edilir. Bir ve kendisinden başka böleni yoksa polinom indirgenemez. Rijndeal'de, bu polinom m(x) olarak adlandırılır ve polinomi Denklem (3.3)'deki gibi;

$$m(x)= x^8+ x^4+ x^3+x+1 \tag{3.3}$$

şeklinde olacaktır.

Örneğin: '57' · '83' ='C1'.

Modüler m(x) tarafından azaltıldığı için sonuç 8 dereceden az, ikili polinom olacaktır ve bu nedenle, bir bayt ile temsil edilebilir olmasını sağlar. Toplama gibi, bayt seviyesinde çarpmaya karşılık gelen basit bir işlem yoktur.

Yukarıda tanımlanan işlem çarpma birleştiricisidir ve {01}elemanı çarpma özdeşliğidir. 8 dereceden az olan herhangi bir sıfır olmayan ikili polinom b(x) için, b(x) çarpmanın tersi $b^{-1}(x)$ aşağıdaki gibi bulunabilir ve Euclid'in genişletilmiş algoritma a(x) ve c(x) polinomlarını hesaplamak için kullanılır [15]. Denklem (3.4), (3.5) ve (3.6)'daki gibi;

$$b(x)a(x)+m(x)c(x)=1 \tag{3.4}$$

Buradan, $a(x) \cdot b(x) \bmod m(x)=1$

$$b^{-1}(x)=a(x) \bmod m(x) \tag{3.5}$$

Üstelik her bir a(x),b(x) ve c(x) için aşağıdaki işlem sağlar;

$$a(x) \cdot\big(b(x)+c(x)\big)=a(x) \cdot b(x)+a(x) \cdot c(x) \tag{3.6}$$

şeklinde olacaktır. Bununla birlike 256 bayt değerler kümesi, yukarıda tanımlanan toplama işlemi gibi kullanılmış XOR ve çarpma işlemi sonlu alan GF(2^8)'in yapısıdır.

3.2.1.4. X ile çarpılması

b(x) polinomu x ile çarpılırsa aşağıdaki Denklem (3.7)'deki gibi;

$$b_7x^8+b_6x^7+b_5x^6+b_4x^5+b_3x^4+b_2x^3+b_1x^2b_0x \quad (3.7)$$

şeklinde olacaktir. Yukarıdaki bölen $m(x)$ modülo işlemi yapıldıktan sonra $x \cdot b(x)$ sonucu elde edilmişti (tam Denklem (3.3)'deki gibi). Eğer $b_7 = 0$ ise, sonuç tamamen gerçekleşmiş (indirgenmiş) demektir. Eğer $b_7 = 1$ ise, m(x) polinomu çıkarılarak sonuç elde edilir. Bunula birlikte x ile çarpılması (yani,{00000010} ya da {02}) bayt seviyesinde bir sola kaydırma ve {1b} ile bitlik (XOR) sırayla uygulanabilmektedir. Bu işlem bayt üzerinden xtime() olarak gösterilir. x'in büyük dereceler ile çarpımı xtime()'in tekrarlanan uygulama kullanarak sağlanabilir. Ortak sonuçları toplayarak, her sabit sayı ile çarpım uygulanabilir.

Örneğin: (Onaltılık sistemde)

{57} · {13} = {fe} ve şekil 3.2'de çarpımı görülmektedir.

$$\{57\} \cdot \{02\} = \text{xtime}(\{57\}) = \{ae\}$$
$$\{57\} \cdot \{04\} = \text{xtime}(\{ae\}) = \{47\}$$
$$\{57\} \cdot \{08\} = \text{xtime}(\{47\}) = \{8e\}$$
$$\{57\} \cdot \{10\} = \text{xtime}(\{8e\}) = \{07\}$$
$$\{57\} \cdot \{13\} = \{57\} \cdot (\{01\} \oplus \{02\} \oplus \{10\}) = \{57\} \oplus \{ae\} \oplus \{07\} = \{fe\}$$

Şekil 3.2. Polinom çarpımı

3.2.1.5. GF(2^8) katsayı ile polinomlar

Polinomlar GF(2^8) katsayı ile tanımlanır. Bu durumda, 4 bayt vektörün karşılığında 4 derecenin altındaki polinom gelir. Polinomlar karşılıklı gelen katsayıları toplayarak toplanabilir. Ayrıca GF(2^8)'deki bitlik XOR, iki vektörün toplaması basit bitlik XOR'dur.

Çarpma işlemi daha karmaşıktır. GF(2^8) alanında iki polinom Denklem (3.7)'deki gibi;

$a(x)=a_3x^3+a_2x^2+a_1x+a_0$ ve $b(x)=b_3x^3+b_2x^2+b_1x+b_0$. (3.7)

şeklinde olacaktır. Bunlar $c(x)=a(x)\cdot b(x)$ olarak verilirse elemanların çarpma işlemi şekil 3.3'de gibi gösterilmektedir.

$$c(x) = c_6x^6 + c_5x^5 + c_4x^4 + c_3x^3 + c_2x^2 + c_1x + c_0$$
$$c_0 = a_0b_0 \quad c_4 = a_3b_1 \oplus a_2b_2 \oplus a_1b_3$$
$$c_1 = a_1b_0 \oplus a_0, \qquad c_5 = a_3b_2 \oplus a_2b_3$$
$$c_2 = a_2b_0 \oplus a_1b_1 \oplus a_0b_2 \quad c_6 = a_3b_3$$
$$c_3 = a_3b_0 \oplus a_2b_1 \oplus a_1b_2 \oplus a_0b_3.$$

Şekil 3.3. Elemanların çarpma işlemi

Açıkçası, artık c(x) 4 bayt vektörü ile ifade edilmez. c(x) 4 derece polinom azaltılarak sonuç polinom 4 dereceden az olur. AES algoritması için bu polinom Denklem (3.8)'deki gibi;

$x^i \bmod x^4+1=x^{i \bmod 4}$ (3.8)

şeklinde olacaktır.

a(x) ve b(x) modüleri $d(x)=a(x)\otimes b(x)$ ile gösterilirse Denklem (3.9)'daki gibi;

$d(x)=d_3x^3+d_2x^2+d_1x+d_0$ (3.9)

şeklinde olacaktır. Bundan eşitlik çarpımı şekil (3.4)'de gösterilmektedir.

$$d_0 = a_0b_0 \otimes a_3b_1 \otimes a_2b_2 \otimes a_1b_3$$
$$d_1 = a_1b_0 \otimes a_0b_1 \otimes a_3b_2 \otimes a_2b_3$$
$$d_2 = a_2b_0 \otimes a_1b_1 \otimes a_0b_2 \otimes a_3$$
$$b_3d_3 = a_3b_0 \otimes a_2b_1 \otimes a_0b_2 \otimes a_0b_3$$

Şekil 3.4. Denklemin eşitlik çarpımı

$a(x)$ sabit bir polinom olduğunda, Denkelm (3.10)'deki gibi;

$$\begin{bmatrix} d_0 \\ d_1 \\ d_2 \\ d_3 \end{bmatrix} = \begin{bmatrix} a_0 & a_3 & a_2 & a_1 \\ a_1 & a_0 & a_3 & a_2 \\ a_2 & a_1 & a_0 & a_3 \\ a_3 & a_2 & a_1 & a_0 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} \tag{3.10}$$

matris şeklinde olacaktır. Çünkü x^4+1 $GF(2^8)$ üzerindeki indirgenemez polinom değildir. Dolayısıyla sabit polinomun çarpımında ters işlem gerekmez. Ancak, AES algoritmasında seçilen sabit polinomda ters işlem vardır. Bu bölüm (3.3.4 ve 3.3.10)'te gösterilmiştir.

3.3. Algoritmanın Genel Yapısı

AES algoritmasında giriş bloğunun uzunluğu, çıkış bloğunun uzunluğu ve durum bloğunun uzunluğu 128 bittir. Nb = 4 ile temsil edilir, bu durum bloktaki 32-bit kelimelerin sayısını yansıtır (sütun sayısı). Bununla birlikte şifreleme anahtarının uzunluğu 128, 192 ya da 256 bittir. Nk = 4, 6 ya da 8 olarak anahtarın uzunluğunu ifade eder ve bu şifreleme anahtardaki 32-bit kelimelerin sayısını yansıtır (sütun sayısı). Algoritma yürütülmesi sırasında yapılacak turun sayısı anahtarın uzunluğuna bağlıdır. Turun sayısı Nr ile temsil edilir. Yani, Nk = 4 ise Nr = 10, Nk = 6 ise Nr =12 ve Nk = 8 ise Nr =14. Yukarıda yazılan standart tablo 3.1'de anahtar tur ile nasıl eşleştirildiği gösterilmektedir.

Tablo 3.1. Anahtar blok tur [15]

	Anahtar uzunluğu – Nk	Bloğun boyut – Nb	Turun sayısı
AES – 128	4	4	10
AES – 192	6	4	12
AES – 256	8	4	14

Şifreleme ve de-şifreleme yapılırken AES algoritması 4 farklı dönüşüm fonksiyonu kullanılır. Aşağıda 4 farklı dönüşüm fonksiyonu belirtilmiştir [15];

1. S-kutusundaki değer ile bayt değişimi
2. Farklı uzaklıklarla Durum matrisin satır kaydırması
3. Her Durum matrisin sütündeki verilerin karıştırılması
4. Durum içine tur anahtarlarının eklenmesi

Bu dönüşümler 3.3.2, 3.3.3, 3.3.4 ve 3.3.5 bölümlerinde daha ayrıntılı anlatılacaktır. Aynı zamanda gelişmiş şifreleme standardı her turda eski anahtarları kullanarak yeni anahtar üretmektedir. Üretilen anahtar ile durum matrisi toplanır. Böylece gelişmiş şifreleme standardı gerçekleşir.

3.3.1. Şifreli mesaj

Şifreleme başında, girişten gelen metin 128 bitlik parçalara bölünerek durum matrisine yerleştirilir. Yani, şifreleme işlemi yapılmadan önce şifrelenecek olan veriler durum matrisine 4x4 (satır ve sütun) yerleştirilir. Durum matrisin her bir eleman bir bayttır ve bir satır toplamda 4 bayttır. Durum matrisi oluşturulduktan sonra, üzerinde tüm işlemler yapılabilir duruma gelmiş demektir. Aynı şekilde önceden alınan 128 bitlik anahtarda bu durum matrisi halini alır. Giriş metninin yazıldığı durum matrisi ilk olarak anahtar ile toplanır. Ardından algoritmanın temeli sayılabilecek işlemler yapılır ve toplam tur sayısı 10, 12 veya 14 olacaktır. Yani, bu anahtarın uzunluğuna bağlı olarak, en son tur diğer turlardan farklı işlenecektir. Bu işlemler yapıldıktan sonra son durum matrisi işlemin sonucunu aktarır. Her turda daha önce saydığımız işlemlerle birlikte bir de tur anahtarı oluşturma işlemi yapılmaktadır. Her turun sonucunda oluşan durum ile o tur için hazırlanmış olan yeni anahtar toplama işlemine tabi tutulur. Şifreli mesaj şekil 3.5'deki pseudo kod ile anlatılmıştır. Bireysel dönüşümler SubBytes(), ShiftRows(), MixColumns() ve AddRoundKey() süreçler alt bölümlerde açıklanacaktır.

```
Cipher(byte in[4*Nb],byte out[4*Nb]){
begin
    byte state[4,Nb];
    for(i = 1 step to 1->Nb)
        for(j = 1 step to 1->Nb)
            state[j][i] =  in[i*4 + j];


    AddRoundKey(0);
    for(round=1 step to Nr-1)
        {
            SubBytes();
            ShiftRows();
            MixColumns();
            AddRoundKey(round);
        }
    SubBytes();
    ShiftRows();
    AddRoundKey(Nr);
    for(i = 1 step to 1->Nb)
        for(j = 1 step to 1->Nb)
            out[i*4+j]= state[j][i];

end
```

Şekil 3.5. Cipher için pseudo kod

Şekil 3.5'de gösterildiği pseudo kodun içindeki her alt fonksiyonda durum matrisi kullanılmıştır. Toplam Nr turda bütün fonksiyonlar işlenmiştir, ancak son turda MixColumn() işlemi bu dönüşümü kapsamaz.

3.3.2. Bayt değiştirme SubBytes()

SubBytes() dönüşümü lineer olmayan baytlık işlemdir. Durum matrisinin her baytı bir birine bağımsız olan s-kutudaki elemanlar kullanılarak işlenmektedir [54]. Tablo3.1'de S-kutu gösterilmiştir. S-kutuda ters işlem yapılabilir ve bu işlem iki dönüşümden oluşmaktadır. Şekil 3.5'te gösterildiği gibi hazır olan s-kutuya erişim yapılır. O anda gelen sayıya karşılık gelen s-kutudaki indeks üzerinde olan sayı geri dönüş yapar. Sonlu alan $GF(2^8)$ çarpımın tersi alınır (3.2.1.3)'de açıklanmıştır. {00} kendisini eşleştirir. Aşağıda gösterildiği gibi $GF(2^8)$ üzerinde afın dönüşümu Denklem (3.11)'deki gibi;

$$b_i^{'}=b_i \oplus b_{(i+4)mod8} \oplus b_{(i+5)mod8} \oplus b_{(i+6)mod8} \oplus b_{(i+7)mod8} \oplus c_i \quad (3.11)$$

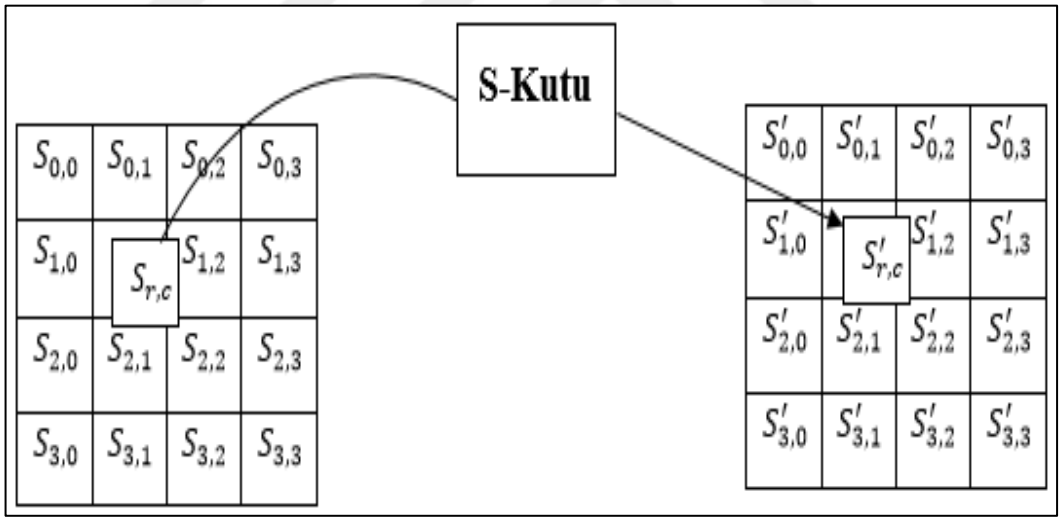
şeklinde olacaktır. $0 \leq i < 8$'se, b_i baytın i'deki bit, c_i c baytın i'deki bit ve değeri ise {63} ve ya {01100011}.

Matris halinde, S-kutunun elemanların afın dönüşü Denklem (3.12)'deki gibi;

$$\begin{bmatrix} b'_0 \\ b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \\ b'_5 \\ b'_6 \\ b'_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} \quad (3.12)$$

şeklinde olacaktır.

Şekil 3.6'da Durum matrisin SubBytes() dönüşümü gösterilmektedir.



Şekil 3.6. SubBytes() dönüşümü

Tablo 3.2'de S-kutunun onaltılık hali gösterilmiş ve SubBytes() dönüşümde kullanılır.

Tablo 3.2. S-kutu alt değerleri xy bayt için (onaltılık hali)

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
	1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
	2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
	3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75

Tablo 3.1. (Devamı) S-kutu alt değerleri xy bayt için (onaltılık hali)

	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
	6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
	7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
	8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
	9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
	a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	fd	ea	65	7a	ae	08
	c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
	d	70	3e	b5	66	48	03	F6	0e	61	35	57	b9	86	c1	1d	9e
	e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
	f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Örneğin: Eğer $S_{1,1}$ ={53}, alt değeri satırın indeks ‘5’ ve sütun indeks ‘3’ olarak seçilir. Sonuç $S'_{1,1}$={ed} olur.

3.3.3. Satır kaydırma ShiftRows()

ShiftRows() dönüşüm ise, durum matrisinin son üç satırındaki baytlar farklı uzaklıklarla kaydırılarak ilk durum matrisindeki baytların yerleri değiştirilecektir. Ancak, ilk satırdaki baytlar kaydırılamaz (r=0).
Özellikle, ShiftRows() dönüşümü Denklem (3.13)’deki gibi;

$$S'_{r,c}=S_{r,(c+\text{kaydır}(r,Nb))\bmod Nb},\ 0<r<4 \text{ ve } 0\leq c<Nb \tag{3.13}$$

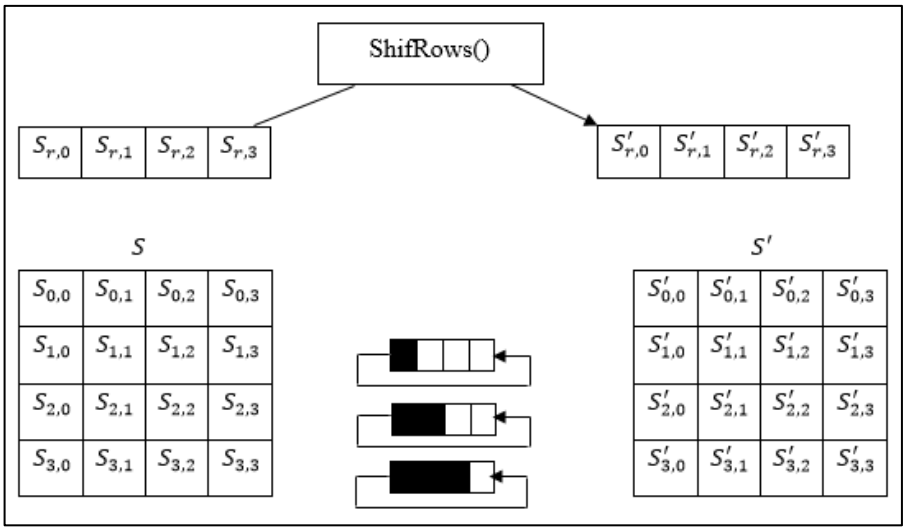
şeklinde olacaktır.

Kaydırma değeri shift(r,Nb) satır sayısına bağlıdır, Nb = 4 olduğuna göre Denklem (3.14)’deki gibi;

$$\text{shift}(1,4) = 1;\ \text{shift}(2,4) = 2;\ \text{shift}(3,4) = 3. \tag{3.14}$$

şeklinde olacaktır.

Bu adımın önemi sütundaki elemanlar doğrusal durumundan kaçınmaktadır. Şekil 3.6’de ShiftRows() dönüşüm gösterilmiştir.



Şekil 3.7. ShiftRows() durum matrisin son üç kaydırma

Şekil 3.7'de satır kaydırma işleminde satırlar sırasıyla çevrimsel şekilde kaydırılırlar. Yani ilk satır değiştirilmez, ikinci satır da sola 1 ötelenir, üçüncü satır sola 2 ötelenir ve son satır sola 3 ötelenir. Taşınan bölmeler kaydırmanın başına eklenir.

3.3.4. Sütun karıştırma MixColumn()

MixColumn() dönüşüm durum matrisinin sütundan sütununa işlem yapar. Bölüm 3.2.1.5'de anlatıldığı gibi her sütun için polinom işlenecektir. Sütunları GF(2^8) polinom gibi düşünüp x^4+1 sabit polinom a(x) ile çarpılacaktır. Sabit polinom a(x) Denklem (3.15)'deki gibi;

$$a(x)=\{03\}x^3+\{01\}x^2+\{01\}x+\{02\} \tag{3.15}$$

şeklinde olacaktır.

Bölüm 3.2.1.5'de tanımladığı gibi matris çarpımında yazılabilir ve Denklem (3.16)'daki gibi;

$$\begin{bmatrix} S'_{0,c} \\ S'_{1,c} \\ S'_{2,c} \\ S'_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} S_{0,c} \\ S_{1,c} \\ S_{2,c} \\ S_{3,c} \end{bmatrix} \text{ ve } 0 \leq c < Nb. \tag{3.16}$$

şeklinde olacaktır.

Çarpımın sonunda, sütündeki 4 baytlar aşağıdaki Denklem (3.17), (3.18), (3.19) ve (3.20)'deki gibi;

$$S'_{0,c}=(\{02\}\cdot S_{0,c})\oplus(\{03\}\cdot S_{1,c})\oplus S_{2,c}\oplus S_{3,c} \quad (3.17)$$

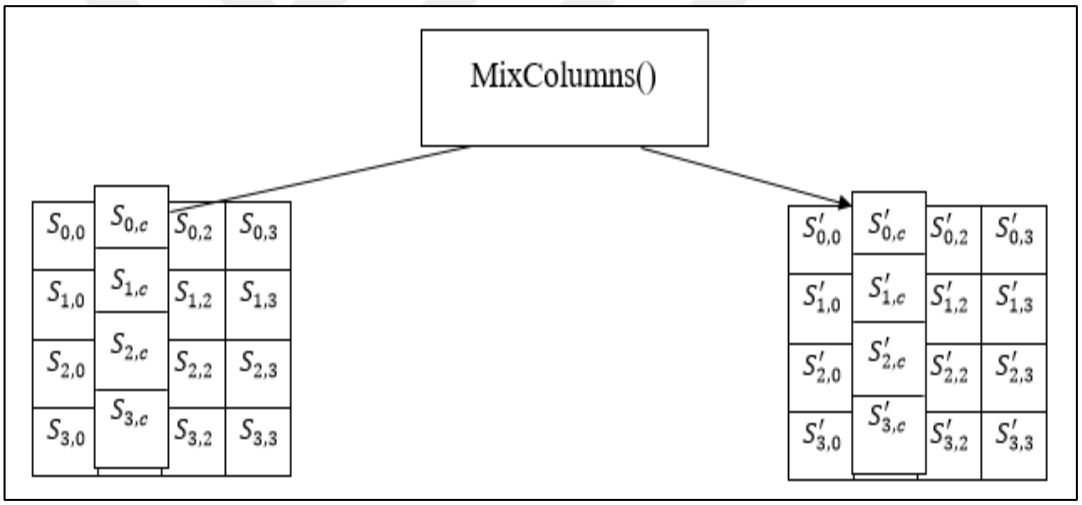
$$S'_{1,c}=S_{0,c}\oplus(\{02\}\cdot S_{1,c})\oplus(\{03\}\cdot S_{2,c})\oplus S_{3,c} \quad (3.18)$$

$$S'_{2,c}=S_{0,c}\oplus S_{1,c}\oplus(\{02\}\cdot S_{2,c})\oplus(\{03\}\cdot S_{3,c}) \quad (3.19)$$

$$S'_{3,c}=(\{03\}\cdot S_{0,c})\oplus S_{1,c}\oplus S_{2,c}\oplus(\{02\}\cdot S_{3,c}) \quad (3.20)$$

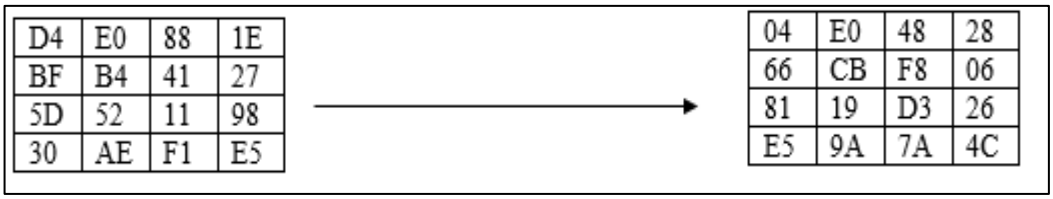
şeklinde olacaktır.

Şekil 3.8'de MixColumns() dönüşümü görüntülenmektedir.



Şekil 3.8. MixColumns() işleme (Durum matris üzerinde sütundan sütuna işleme)

Bu işlemde eski sütunun elemanları kullanılarak yeni sütun elde edilmektedir. Bu yapılırken yeni sütunun elemanları eski sütunun her elemanının hesaba katılarak tek tek hesaplanması gerekir. Yapılan hesap çarpma ve toplama işleminden oluşur. Çarpma işleminde belirli bir sabit sayı (a(x)) (3.16) kullanılır. İşlem örneği şekil 3.9'da gösterilmektedir.



Şekil 3.9. Sütun Karıştırma Örneği

S_0'={04} 'ü bulalım:İlk önce formülde yer alan 4 XOR işlemini parçalara ayırarak, işlemleri ayıralım.{02}·S_0 = {02}·{D4} ve buradan {02} = 0000 0010 = x ve {D4} = 1101 0100.

Burada x^8 olduğu için indirgeme işlemi gerçekleştirilecektir. Bu işlemde ise, x^8 ve üzeri derecede bulunan yerler indirgenerek yeni denklem elde edilir. 8 dereceden indirgenemez polinomun böleni tekdir ve sadece kendisidir. AES için bu polinom gösterimi ise;

Bu 4 işlem sonucunu XOR işlemine sokarız.

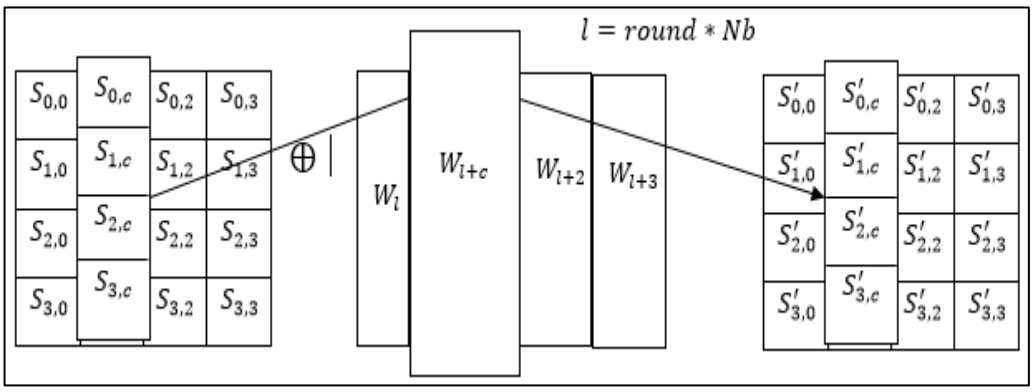
Tablo 3.2. XOR işlem

{02}*{D4}	1	0	1	1	0	0	1	1
{03}*{BF}	1	1	0	1	1	0	1	0
{5D}	0	1	0	1	1	1	0	1
{30}	0	0	1	1	0	0	0	0
S_0(ikili)	0	0	0	0	0	1	0	0
S_0(onaltılık)	0				4			

3.3.5. Tur anahtarıyla toplama addRoundKey()

AddRoundKey() dönüşümde, Tur Anahtar ve Durum matrisi arasında basit bir bitlik XOR işlem yapılır. Bütün tur anahtar zamanlaması Nb'den oluşmaktadır (bölüm 3.3.6'da açıklamaktadır). Bu Nb ve sütunun sayısı durum matrisinin içine eklenmektedir. Denklem (3.21)'deki gibi:

$$[S'_{0,c},S'_{1,c},S'_{2,c},S'_{3,c}]=[S_{0,c},S_{1,c},S_{2,c},S_{3,c}]\oplus[W_{round+Nb+c}]\ 0\leq c<Nb \quad (3.21)$$



Şekil 3.10. AddRoundKey() Zamanlama anahtar kelime ile durum matrisin her sütunla XOR işlemi

şeklinde olacaktır. [W_i] bölüm 3.3.6'da açıklanan anahtar zamanlamasıdır. Turun değeri ise 0≤round≤Nr. Şifreleme aşamasında tur anahtarının başlangıç tur=0 ve şekil 3.3'de uygulamadaki round(tur)=0 olarak belirlendiğini gösterilmiştir. Uygulamadaki AddRoundKey() dönüşümü round : 1≤round≤Nr ise Şifrelemesinin toplam turu Nr olacaktır.

Şekil 3.10'da dönüşümün işlemi verilmiştir. Şekilde görüldüğü gibi işlem yapılmaktadır. Her turda daha önce saydığımız işlemlerle birlikte bir de tur anahtarı oluşturma işlemi yapılmaktadır ve her turun sonucunda oluşan durum ile o tur için hazırlanmış olan yeni anahtar toplama işlemine tabi tutulur. Bu işlem sonlu alanlarda yapılan toplama işlemidir ve bit mertebesinde özel işlemine karşılık düşer. 128 bitlik durum matrisi, 128 bitlik ara anahtar değeri ile toplanır.

3.3.6. Anahtar üretme (Key Expansion)

AES algoritması anahtarı alır ve bir dizi işlemden geçirerek işlem sayısı kadar anahtar oluşturur. Bu sayı 128 bitlik uzunluk için 10'dur. 10 farklı anahtar oluşturulur ve oluşan son anahtar şifreyi çözmede kullanılan ilk anahtar haline gelir. Çözerken de aynı işlemler benzer olarak tersten yürütülerek kullanılır. Anahtar üretmede her yeni oluşturulan yeni anahtar kendinden önceki anahtarlar kullanılarak elde edilir. Giriş anahtardan, anahtar üretme pseudo kodu Şekil 3.9'de verilmiştir. W[i] ise o zamanda üretilen anahtar ve bir önceki aşamada üretilen W[i-1] ile XOR bitlik işlem yapılarak tur sabit anahtarı elde edilmektedir.

Şekil 3.11'de açıklandığı gibi SubWord() fonksiyonu kodu fonksiyon olarak değil direkt kod olarak yazılmıştı. Yani, dört baytlık giriş değeri S-kutusu ile eşleştirilip ona karşılık gelen dört baytlık değer S-kutusudan alınmalıdır. Bununla aynı durumda RotWord() fonksiyonu direkt kodda yansıtmıştır. RotWord() fonksiyon basit bir işlemi ve girişte gelen kelimeyi bir sola kaydıran fonksiyondur. Turun sabit matrisi Rcon ise, alan GF(2^8) ve bölüm 3.2.1.3'de açıklanan polinom değerler olan matristir. Şekilde de görüldüğü üzere, yeni anahtarın oluşumundaki temel işlem bir önceki sütun ile önceki sütunun toplanmasıdır. Ancak bir istisna nokta var ki o da her 4'ün katı olan sütunda toplamadan önce bir dizi işlemden (T İşlemi) daha geçirilir. Bu işlemler öteleme, S kutusundan geçirme ve Rc(x) vektörü ile toplama işlemidir.

```
keyexpansion(int key[], int roundkey[])
begin
    word temp;
    int i,j,k;
    for(i=0;i<nk;i++)
    {
        roundkey[i*4]=key[i*4];
        roundkey[i*4+1]=key[i*4+1];
        roundkey[i*4+2]=key[i*4+2];
        roundkey[i*4+3]=key[i*4+3];
    }
    // bütün tur anahtarlar önceki tur anahtarlardan bulunur.
    while (i < (nb * (nr+1)))
    {
                    for(j=0;j<4;j++)
                        temp[j]= roundkey[(i-1) * 4 + j];
                    if (i % nk == 0)
                    {
                        // Bu fonksiyonda 4 baytlık kelime sola 1 kere kaydırılır.
                        // [a0,a1,a2,a3] -> [a1,a2,a3,a0]
                        // function rotword()
                        {
                            k = temp[0];
                            temp[0] = temp[1];
                            temp[1] = temp[2];
                            temp[2] = temp[3];
                            temp[3] = k;
                        }

                        // subword() fonksiyon 4 baytlık girişte gelen kelime S-kutu ile eşleştirilip
                        // s-kutudan karşılık gelen 4 baytlık kelime alınır.
                        // function subword()
                        {
                            temp[0] = getsboxvalue(temp[0]);
                            temp[1] = getsboxvalue(temp[1]);
                            temp[2] = getsboxvalue(temp[2]);
                            temp[3] = getsboxvalue(temp[3]);
                        }
                        temp[0] = (temp[0] ^ rcon[i/nk]);
                    }
                    else if (nk > 6 && i % nk == 4)
                    {
                        // function subword()
                        {
                            temp[0]=getsboxvalue(temp[0]);
                            temp[1]=getsboxvalue(temp[1]);
                            temp[2]=getsboxvalue(temp[2]);
                            temp[3]=getsboxvalue(temp[3]);
                        }
                    }
                    roundkey[i*4+0] =  (roundkey[(i-nk)*4+0] ^ temp[0]);
                    roundkey[i*4+1] =  (roundkey[(i-nk)*4+1] ^ temp[1]);
                    roundkey[i*4+2] =  (roundkey[(i-nk)*4+2] ^ temp[2]);
                    roundkey[i*4+3] =  (roundkey[(i-nk)*4+3] ^ temp[3]);
                    i++;
    }
```

Şekil 3.11. Key Expansion için pseudo kod

Şekil 3.12'de bir anahtar üretim örneği gösterilmektedir.

2B	28	AB	09
7E	AE	F7	CF
15	D2	15	4F
16	A6	88	3C

→

A0	88	23	2A
FA	54	A3	6C
FE	2C	39	76
17	81	39	05

Şekil 3.12. Anahtar üretim örneği

3.3.7. Ters şifreli mesaj

AES algoritmasında oluşturulan şifreli metinde kolaylıkla, ters işlemlerle tekrar çözülerek giriş metni elde edilebilmektedir. Bu çözme işi için yapılan işlemler ters satır kaydırma (InvShiftRows()), ters S kutusundan geçirme (InvSubBytes()), ters sütun karıştırma (InvMixColumns) ve tur anahtarı (AddRoundKey) ile toplama işlemidir. Şekil 3.13'te ters şifreleme ve çözme işleminin pseudo kodu verilmiştir.

```
InvCipher(byte in[4*Nb],byte out[4*Nb])
begin
    byte state[4,Nb];
    for(i = 1 step to 1->Nb)
        for(j = 1 step to 1->Nb)
            state[j][i] =  in[i*4 + j]; // girişte olan veri 4x4 matrisa koyulur
    AddRoundKey(Nr); //Tur başlamadan önce durum matrisina ilk tur anahtarı eklenmektedir
        for(round=1 step to Nr-1)
        {
            InvShiftRows();
            InvSubBytes();
            AddRoundKey(round);
            InvMixColumns();
        }


        InvShiftRows();
        InvSubBytes();
        AddRoundKey(0);
    for(i = 1 step to 1->Nb)
        for(j = 1 step to 1->Nb)
            out[i*4+j]= state[j][i];
end
```

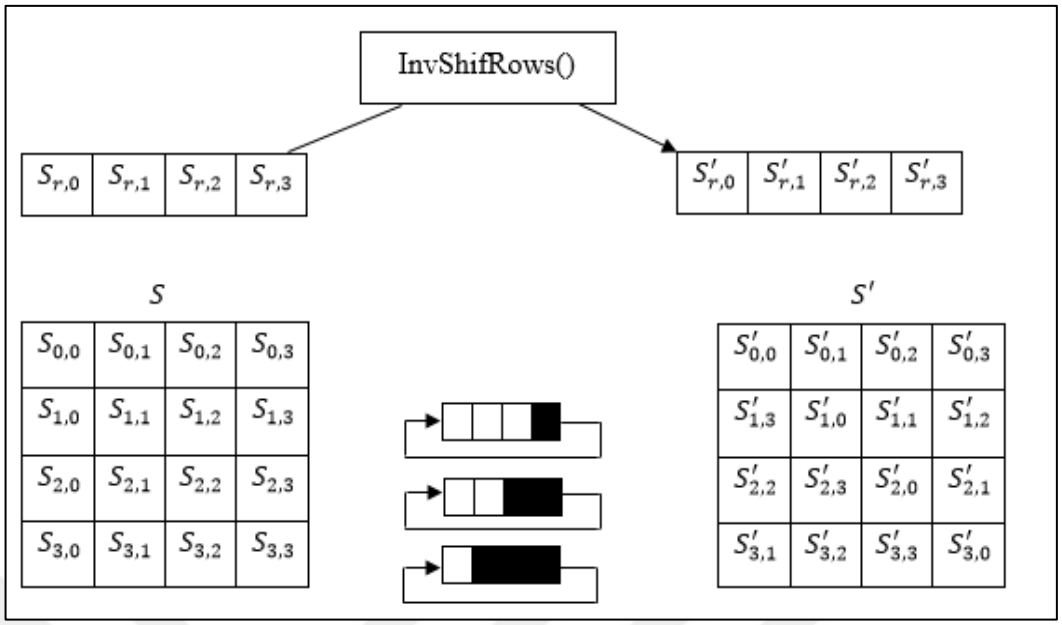
Şekil 3.13. Ters şifreleme için pseudo kod

3.3.8. Ters satır kaydırma (InvShiftRows())

Ters satır kaydırma ise, satır kaydırma işleminin tam tersi ve sağa kaydırarak yapılmasıdır. Satır kaydırma işleminde benzer durumunda ilk satır r=0 kaydırılmaz, kalan satırlar kaydırılacaktır. Ters kaydırma işlemi şekil 3.14'te gösterilmiştir. İfadesi Denklem (3.22)'deki gibi;

$$S'_{r,(c+shift(r,Nb))modNb} = S_{r,c} \text{ ve } 0<r<4,\ 0\leq c<Nb \tag{3.22}$$

şeklinde olacaktır.



Şekil 3.14. Durum matrisindeki ters kaydırma işlemi

3.3.9. Ters bayt değiştirme (InvSubBytes())

Şifreleme işlemindeki bayt değiştirme işlemini sağlayan S-kutusu tablosunun ters simetriği olan ters S-kutusu tablosu da bulunmaktadır. Eski değerlere geri dönebilmek için ters S-kutu tablosundan değerler alınır. S-kutusundaki değerler 3.9'de gösterildiği gibi GF(2^8) sonlu alana dayanmış olduğu matematiksel işlemlerin alt modüllerden oluşur. Sonuç olarak oluşan kutu; normal S-kutusunda elde ettiğimiz değeri tekrar ters S-kutusu girişine uyguladığımızda bize ilk değeri geri verecek şekilde düzenlenmiş halinden ibarettir.

Tablo 3.3. Ters S-kutu değerleri (onaltılık)

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
	1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
	2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
	3	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
	4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
	5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
	6	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
	7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
	8	30	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
	9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
	a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
	b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4

Tablo 3.4. (Devam) Ters S-kutu değerleri (onaltılık)

x	c	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
	d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
	e	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
	f	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

3.3.10. Ters sütun karıştırma (InvMixColumns())

Ters sütun karıştırma sütun karıştırmanın ters işlemidir. Bu işlem sütundan sütuna işlenerek (3.2.1.5)'de anlatıldığı gibi ters polinomla çarpılacaktır. Ters polinom ifadesi Denklem (3.23)'deki gibi;

$$a^{-1}(x)=\{0b\}x^3+\{0d\}x^2+\{09\}x+\{0e\}. \tag{3.23}$$

şeklinde olacaktır.

Matris halinde yazılırsa Denklem (3.24)'deki gibi;

$$\begin{bmatrix} S'_{0,c} \\ S'_{1,c} \\ S'_{2,c} \\ S'_{3,c} \end{bmatrix} = \begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix} \begin{bmatrix} S_{0,c} \\ S_{1,c} \\ S_{2,c} \\ S_{3,c} \end{bmatrix} \text{ ve } 0 \leq c < Nb \tag{3.24}$$

şeklinde olacaktır.

Çarpımını sonunda 4-baytlık sütundaki elemanlar Denklem (3.25), (3.26), (3.27) ve (3.28)'deki gibi;

$$S'_{0,c}=(\{0e\}\cdot S_{0,c})\oplus(\{0b\}\cdot S_{1,c})\oplus(\{0d\}\cdot S_{2,c})\oplus(\{09\}\cdot S_{3,c}) \tag{3.25}$$

$$S'_{1,c}=(\{09\}\cdot S_{0,c})\oplus(\{0e\}\cdot S_{1,c})\oplus(\{0b\}\cdot S_{2,c})\oplus(\{0d\}\cdot S_{3,c} \tag{3.26}$$

$$S'_{2,c}=(\{0d\}\cdot S_{0,c})\oplus(\{09\}\cdot S_{1,c})\oplus(\{0e\}\cdot S_{2,c})\oplus(\{0b\}\cdot S_{3,c}) \tag{3.27}$$

$$S'_{3,c}=(\{0b\}\cdot S_{0,c})\oplus(\{0d\}\cdot S_{1,c})\oplus(\{09\}\cdot S_{2,c})\oplus(\{0e\}\cdot S_{3,c}) \tag{3.28}$$

şeklinde olacaktır.

3.3.11. Tur anahtar ile çözme (AddRoundkey())

Bit işleminden ibaret olması sebebi ile tekrar bit işlemine tabi tutulması eski haline dönmesine sebep olmaktadır. Sonuç olarak şifreleme işlemindeki tur anahtarı ile toplama işleminin aynısı yürütülür. Şifreleme ve çözmede ilk olarak anahtar üretim gerçekleşir ve böylece şifrelemenin tam tersi çözüm işleminde yapılmaktadır. Şifreleme işleminde oluşan son anahtar, çözme işleminin ilk anahtarı olarak kullanılır. Anahtar üretiminin tersi işlemler ile ters anahtar üretimi gerçekleştirilir. Dolayısıyla çözmede şifreleme gibi anahtar üretim işlemi gerçekleşmektedir. Şekil 3.9'da gördüğümüz gibi önce elimizdeki anahtarın sütunlarını kayıt altına alırız. Sütunlara baştan itibaren w0, w1, w2, w3 dersek; önce w2 ve w3'ü XOR ile toplar, yeni w0 değerini elde ederiz. Aynı şekilde w1 ve w2 işleminin sonucu da yeni w1 değerini verir. Bu şekilde geriye doğru giderek yeni durum oluşturulur. Son olarak yeni oluşan w0 sütununu 1 sola öteledikten sonra her sütunu S-kutusundan geçiririz ve ters Rc katsayısı ile çarpılır. Böylece yeni anahtar oluşmuş olur.

4. BLUETOOTH DÜŞÜK ENERJİ NEDİR?

Bu bölümde Bluetooth düşük gücünün genel tanıtımı, özellikleri ve BLE protokolünün katmanları açıklanmaktadır. BLE, Bluetooth protokolünün 5'inci neslidir. Bluetooth'un bu sürümü önceki sürümlerinden farkı ise, veri aktarma hızı, veri aktarma mesafesi ve bağlanma hızı daha fazladır. Aşağıda Bluetooth ve BLE' nin farkları ve özelliklerine değinilmektedir.

BLE (Bluetooth düşük enerji), klasik Bluetooth teknolojisinin ardından yapılmış olan düşük güçlü yeni bir teknolojidir. İlk 2006 yılında Nokia şirketi tarafından akıllı Bluetooth olarak tanıtılmıştır. Klasik Bluetooth ayrı dizüstü bilgisayar ve cep telefonları bağlantısı için tasarlanmıştır. Simdi ise, müzik, yayma ve ayrı ayrı dosyaları aktarılabilen bir teknolojisi olarak bilinmektedir. Bluetooth veri haberleşme fiziksel katmanda maksimum veri saniyede 1 megabit (Mbps) hızına kadar ulaşabilir. Gelişmiş veri hızı (EDR) ve bluetooth 2'inci sürümünde fiziksel katman veri hızı saniyede 3 megabit(Mbps) olarak artırılmıştır. Bluetooth sürümü 3.0 alternatif MAC PHY eklenerek IEEE 802.1 kullanıldığı için fiziksel katman hızı saniyede yüzlerce megabit'e kadar hızlanmıştır. Ancak bluetooth sürümü 4 ve BLE (Smart bluetooth) tamamen farklı yöne amaçlanmıştır [20]. Yani, hız artırma yerine düşük güç tüketimi için tasarlanmıştır.

Tablo 4.1. İletişim hızı [20]

Modem	Ethernet	Wi-Fi	Bluetooth
V.21: 0,3kbps	802.3i:10Mbps	802.11:2Mbps	v1.1:1Mbps
V.22: 1,2kbps	802.3u:100Mbps	802.11b:11Mbps	v2.0:3Mbps
V.32: 9,6kbps	802.3ab:1000Mbps	802.11g:54Mbps	v3.0:54Mbps
V.34: 28,8kbps	802.3an:10000Mbps	802.11n:135Mbps	v4.0:1Mbps-2Mbps

Tablo 4.1'de gösterildiği gibi kablolu iletişim kablosuz iletişimden daha hızlı olduğu anlatılmıştır. BLE ise, pil ile çalıştığı için az güç tüketir ve bundan dolayı veri aktarma

hızı yavaştır. BLE ile çalışabilmesi için maliyet tasarımına bakmak çok önemli ve bunda üç temel unsur vardır;

1. ISM bant;

2.4GHz ISM bant ile çalıştığı için daha az maliyetsizdir. Çünkü diğer teknolojilere göre serbest kullanılan bir banttır.

2. IP Lisans;

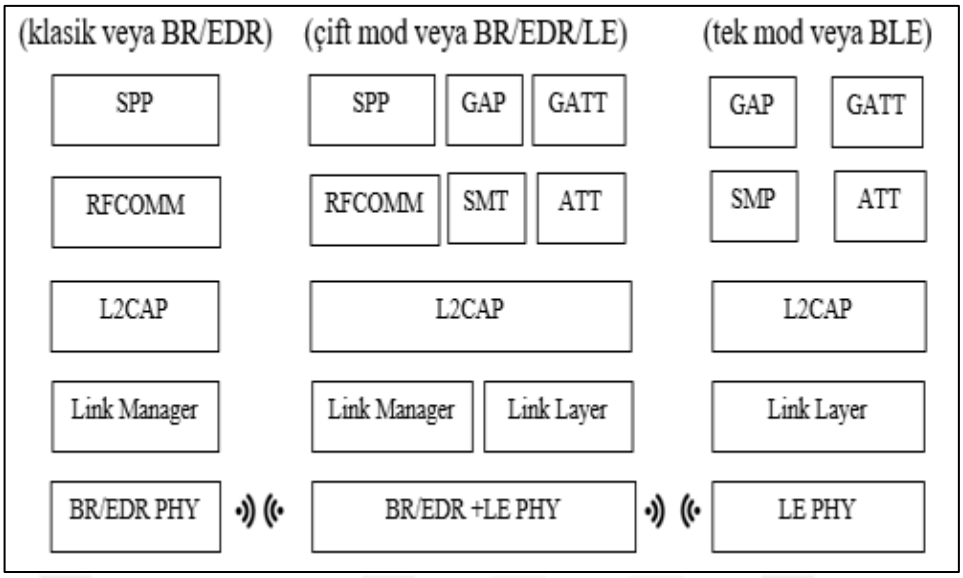
Nokia şirketi tarafından tasarlanmış Wibree teknolojisi diğer kablosuz iletişimler gibi 2,4 GHz ISM bant ile çalışır. 2,4 GHz ile çalışan cihazların lisans maliyeti daha azdır.

3. Düşük güç;

BLE donanımı daha az pil harcanacak şekilde tasarlandığı için daha az güç ile çalışabilir.

4.1. Donanım Çeşitleri

BLE, iki farklı modda çalışabilir: çift mod (dual mode) ve tek mod (a single mode). Çift modda Bluetooth klasik BLE'yi destekler ama tek modda sadece BLE'yi destekler. Çift modda aygıt yeni donanım ve yazılım kullanıldığı için her çeşit cihazlar birbiriyle bağlanabilmektedir. Tek modda ise, sadece tek mod ile çalışan donanımlar desteklenir. Dolayısıyla daha az güç ve ultra düşük güç ile çalışabilmektedir. BLE tek modda ses aktarımı ve yüksek kapasiteli veri aktarma desteklenmediği için şuan bluetooth klasik kullanılmaktadır [21]. Tablo 4.1'de aygıtların hangi cihazlarla uyumlu olduğu gösterilir ve tek modda çalışan BLE aygıtı, diğer tek modda çalışan aygıtlara bağlanabilir. Çift modda çalışan donanımlar diğer çift modda çalışan cihazlara bağlanabilmesinin yanında BR/EDR kullanılan bluetooth klasik cihazlara da bağlanabilmektedir [22]. Diğer bir deyişle, tek modda çalışan aygıt klasik aygıtlarla veri iletişimi kuramazlar. Şekil 4.1'de Bluetooth sürümlerinin ayarları ve cihaz türleri gösterilmiştir. Şekil 4.1'de görüldüğü gibi bluetooth klasik, çift mod ve tek mod sürümlerinin katman farkları var ve bir biriyle iletişim kurabildikleri gösterilmiştir.



Şekil 4.1. Bluetooth sürümleri ve türleri arasındaki yapılandırmaları [23]

Şekil 4.1 ve tablo 4.2'de çift mod ve BR/EDR/LE ise BR/EDR ve BLE ikisi de uygulandığı için her bluetooth cihazla iletişim kurabilmektedir. 4.3, 4.4, 4.5, 4.6, 4.7, 4.8, 4.9, 4.10 ve 4.11 bölümlerde çift moddaki çalışma mantığı ve çalışma profilleri açıklanacaktır.

Tablo 4.2. Tek mod, çift mod ve klasik modların çalışma uyumluğu

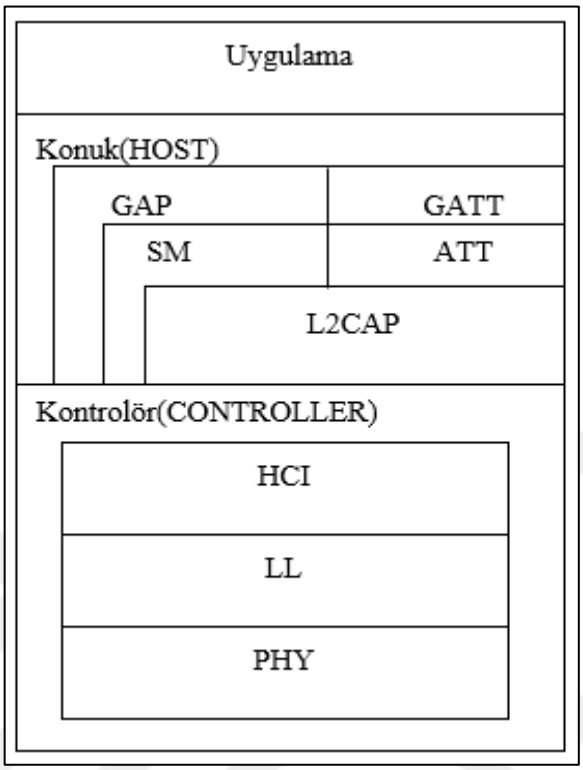
№	Tek moda	Çift moda	Klasik
Tek moda	Düşük güç	Düşük güç	
Çift moda	Düşük güç	Klasik	Klasik
Klasik		Klasik	Klasik

4.2. Bluetooth Düşük Enerji Protokol

Bu bölümde şekil 4.2'de belirtildiği gibi bloklar her Bluetooth cihazın içinde bulunan uygulama, konuk, kontrolör kısımları açıklanacaktır. Önceki bölümde bulunan donanım çeşitlerinin katmanları, bu 3 bloğun içinde bulunmaktadır. Aşağıda 3 bloğun görevi yazılmıştır;

- Uygulama – Bluetooth protokolü kullanıcı uygulaması ile bağlanmaktadır. Bunun yardımıyla, kullanıcı bilgileri daha anlaşılabilir şekilde aktarılmaktadır. Ve veriler üzerinde işlenen tüm kontroller burada bulunur.
- Konuk – Bluetooth protokolünün üst profillerden oluşan kısmıdır. Bu kısım, bluetooth fiziksel kısmı ve uygulama kısmını bir biriyle bağlamaktadır.

- Kontrolör – Bluetooth protokolünün alt ve taban kısımlarından oluşmaktadır. Bu kısımda bluetooth fiziksel katmanda nasıl çalıştığına karar verilmektedir.



Şekil 4.2. Bluetooth düşük güç protokol [24]

BLE protokolü genel mimarisi şekil 4.2'de gösterilmiştir. Uygulamada kullanılan BLE cihaz, bir çip üzerine bütün sistem yerleştirilmiş olan BLE cihazıdır. Avantajı ise, düşük güç CPU ile uyumlu yapıldığı için en az güç harcanan sistem elde edilmiş demektir. 4.3 -4.11 bölümüne kadar Bluetooth düşük güç protokol içindeki profiller anlatılmaktadır. Bu bölümlerde hangi profilin nasıl çalıştığı incelenmektedir.

4.3. Genel Erişim Profili (GAP)

GAP, tüm profiller arasındaki iletişimin nasıl sağlanacağını ve kullanıcı tarafından verilen bilgileri BLE üzerinden nasıl aktaracağından sorumludur. Bu sorumluluğun içinde cihazın bulunması, bağlantı kurulması, bağlantı sonlandırması, güvenlik özelliklerin başlatması ve cihazın ayarın yapılmasıdır. GAP katmanı 4 farklı rol almaktadır. Bu rollerden;

- Yayın yapan (broadcaster) – Duyuru yapar ancak bağlanmaz.
- Gözleyici (observer) – Duyuları tarar ancak bağlantı başlatamaz.

- Çevrebirimi (peripheral) –Bağlanabilir ve tek bir bağlantı bağlantısında yardımcı olarak çalışmaktadır.
- Merkezi (central) – Duyuruları tarar ve bağlantıları başlatır; tek veya çoklu bağlantı katmanı bağlantılarında ana olarak çalışır.

Bundan sonra bluetooth düşük enerji – BLE olarak yazılmaktadır. BLE çoklu-rol birleştirmesi olarak tanımlanabilmektedir. Örnekleme uygulamaların ilk ayarı çevrebirimi desteklenir. Kaynak kodlara göre çevrebirimi ve yayın yapan rolleri birleştirerek çalıştırabilmektedir.

Benzer BLE sistemlerde, çevrebirim cihazlar belirli verileri duyurur ve her hangi bir merkez cihaz ona bağlı olduğunu bilir. Bu duyuruda cihazın adresi, adı ve bazı ilave verileri içermektedir. Merkez cihaz, duyuru aldıktan sonra çevrebirim cihaza "tarama isteği" gönderir. Çevrebirim cihazı "tarama yanıtı" ile yanıt verir. Bu, cihaz bulunma işlemi ve o anda merkez cihaz çevrebirim cihazını bulup bağlantı kurabilmektedir. Daha sonra merkez cihaz çevrebirim cihaz ile bağlantı kurmak için bir istek gönderir. Bir bağlantı isteği birkaç bağlantı parametrelerini içerir ve aşağıda yazılmıştır;

- Bağlantı aralığı – iki cihaz arasındaki BLE bağlantı, frekans – atlamalı bir düzeni kullanır. Bu durumda iki cihaz her veri gönderme ve alma işlemi için belirli bir kanal üzerinden haberleşmektedir. Belirli bir zaman sonra yeni kanalda karşılaşırlar. Yani, BLE, bağlantı katmanına kanal geçişi yapar. Bu karşılaşmada iki cihazın veri gönderme ve alma işlemi "bağlantı olayı" olarak bilinir. Uygulamada hiçbir veri gönderme veya alma yoksa iki cihaz bağlantı kurmak için hala bağlantı katmanında veri tamamlanıyor demektir. Bağlantı aralığı iki bağlantı arasındaki süredir ve 1,25ms birimdir. Bu bağlantı aralığı 7,5 ms'den 4,0 saniye kadar değişebilmektedir.

Uygulamaya göre farklı bağlantı aralıkları gerekebilir. Uzun bağlantı aralığının avantajı güç tasarrufu modudur, çünkü uzun zaman veri alışverişi yapmayan cihazlar uyku moduna girer. Dezavantajı ise cihazdan veri göndermek gerekirse, bir sonraki bağlantı olayını beklemek zorundadır.

Az bağlantı aralığının avantajı ise iki cihaz daha sıkı bağlantı kurabilir ancak daha fazla güç harcanmaktadır.

- Yardımcı gecikme – Bu parametre çevrebirim (Periferi-yardımcı) cihazın bağlantı olaylarının atlama sayısı seçeneği verir. Bu yardımcı gecikme cihazlara bazı esneklik verir, yani, herhangi bir veri gönderilmeyecekse bağlantı olay atlama ve uyku moduna geçilir. Dolayısıyla güç tasarrufu olur. Bu seçenek yardımcı cihaza bağlıdır. Yardımcı gecikme olayları maksimum atlanabilir sayıyı gösterir. Bu en az 0 (Yani, bağlantı olay yok demektir ve atlanabilir) ve en fazla 499 olur, Yani, bu işlem 0 - 499'a kadar herhangi bir değer alabilir.
- Zaman aşımı – Bu iki cihaz arasındaki başarılı bağlantı olayının maksimum süresinin miktarıdır. Eğer başarılı bir bağlantı olmadan süre miktarı geçilirse cihazın bağlantısı kesilmiş ve bağlantısız duruma geri dönmüş demektir. Bu parametre değeri 10ms içinde karara bağlanmış olur. Zaman aşımı değeri 10 (100ms) - 3200(32,0s) aralığında değişir. Ek olarak, zaman aşımı etkili bağlantı aralığından daha fazla olması gerekir. Bu olay aşağıda açıklanmıştır.

"Etkili bağlantı olayı" iki bağlantı olayı arasındaki süreye eşittir. Eğer yardımcı gecikme sıfırsa, etkili bağlantı olayı gerçek bağlantı aralığına eşit olur ve bu aşağıdaki Formül (4.1)'de göstermiştir.

Etkili bağlantı aralığı = (bağlantı aralığı)·(1+(yardımcı gecikme)) (4.1)

Örneğin:
Bağlantı aralığı: 80(100ms)
Yardımcı gecikme: 4 birim
Etkili bağlantı aralığı: (100ms)·(1+4)=500ms

Yukarıdaki olayda, yardımcı cihazdan merkez cihaza veri gönderilmez ve yardımcı cihaz her 500ms'de bağlantı olayını iletir demektir. Dolayısıyla bağlantı aralığı BLE cihazlarının güç konusunda önemli rol aldığı anlamına gelir:

Bağlantı aralığı azaltılırsa;

- Her iki cihaz için güç tüketimi artar.
- Her iki yönde verim artar.
- Her iki yönde gönderilecek olan veriler için gereken süre azalır.

Bağlantı aralığı artırılırsa;

- Her iki cihaz için güç tüketim azalır.
- Her iki yönde verim azalır.
- Her iki yönde gönderilecek olan veriler için gereken süre artar.

Yardımcı gecikme azaltılırsa (sıfıra yakın olursa);

- Yardımcı cihaz için güç tüketimi artar.
- Merkez cihaz tarafından yardımcı cihaza gönderilen veriler için gereken süre azalır.

Yardımcı gecikme artırılırsa;

- Yardımcı cihazdan merkez cihaza gönderilecek veriler olmadığı sürece yardımcı cihazın güç tüketim azalır.
- Merkez cihaz tarafından gönderilen veri yardımcı cihaz tarafından alınırken gereken süre artar.

Bazı durumlarda, merkez cihaz ile yardımcı cihazın bağlantı parametrelerini içeren talep yardımcı cihaza gönderilir. Diğer durumlarda, yardımcı cihaz üzerinde uygulanan uygulamaya göre bağlantı ortasında parametrelerini değiştirmesi için istek gönderebilir.

GAP ayrıca bir BLE bağlantı sırasında güvenlik özelliklerini başlatılmasını yönetir. Bazı veriler yalnızca kimliği doğrulandığı durumda okunabilir ve yazılabilir. Bir bağlantı oluşturulduktan sonra, iki cihaz eşleştirme olarak adlandırılan bir sürece geçer. Eşleştirme yapıldığında, bağlantı doğrulanması için anahtar girilmesi gerekir. Tipik durumda, çevrebirim cihaz merkez cihaz ile eşleştirmesini tamamlamak için geçiş anahtarı ister. Bu anahtar genelde “000000” ve ayarlanabilmektedir.

4.4. Genel Özellik Profili (GATT)

BLE protokolü içindeki GATT katmanı bağlantı kurulmuş iki cihaz arasında veri iletişimi için uygulama tarafından kullanmak üzere tasarlanmıştır. GATT açısından bakıldığında, iki cihaz bağlandığında, iki farklı durumda çalışmaktadır.

- GATT alıcı – Bu cihaz, GATT sunucudan yazma veya okuma işlemlerini alır.
- GATT sunucu – Bu cihaz, GATT müşteri tarafından yazılacak/okunacak olan verileri içermektedir.

GATT profilindeki alıcı ve sunucu durumları BLE bağlantı-katmanının yardımcı ve başrol veya GAP profilindeki merkez ve çevrebirimlerinden bağımsızdır. Diğer bir deyişle, BLE bağlantı-katmanının yardımcısı, GATT profilindeki GATT alıcısı veya GATT sunucusu olur. Bununla aynı durumda, BLE bağlantı-katmanın başı, GATT alıcısı veya GATT sunucusu olabilir.

GATT sunucusu birden fazla GATT tarafından hizmet verebilmektedir. Bu, belirli bir işlevi veya özellikleri taşıyan verilerin koleksiyondur.

Her hizmetlerin "özellikleri" vardır. Bu bir hizmet tarafından kullanılan değerler demektir. GATT, BLE bağlantı üzerine alt işlemleri bulma, okuma ve yazma niteliklerini tanımlar. Bu değerler, GATT sunucusu içinde yer alan özellikleri tabloda "tanımlayıcı" olarak adlandırılan kısmında yerleşiktir. Özellikler tabloda basit veri parçaları içeren veri tabanlarıdır. Her değer aşağıdaki özelliklere sahiptir;

- Algılanma – Tablodaki özelliğin adresidir. Her özellikte benzersiz hata algılanma vardır.
- Tip – Bu veriler neyi temsil ettiğini belirtir. Bu Bluetooth SIG tarafından ayarlanan "UUID" (evrensel benzersiz tanımlayıcı) olarak adlandırılır. "UUID" 128 bit veya 16 bayt benzersiz adrestir.
- İzinler – GATT alıcı cihaz hangi verilere erişilebileceğini gösterir.

GATT, GATT sunucu ve alıcı arasındaki iletişim için alt işlemler tanımlanır. Bu işlemler aşağıda gösterilmiştir;

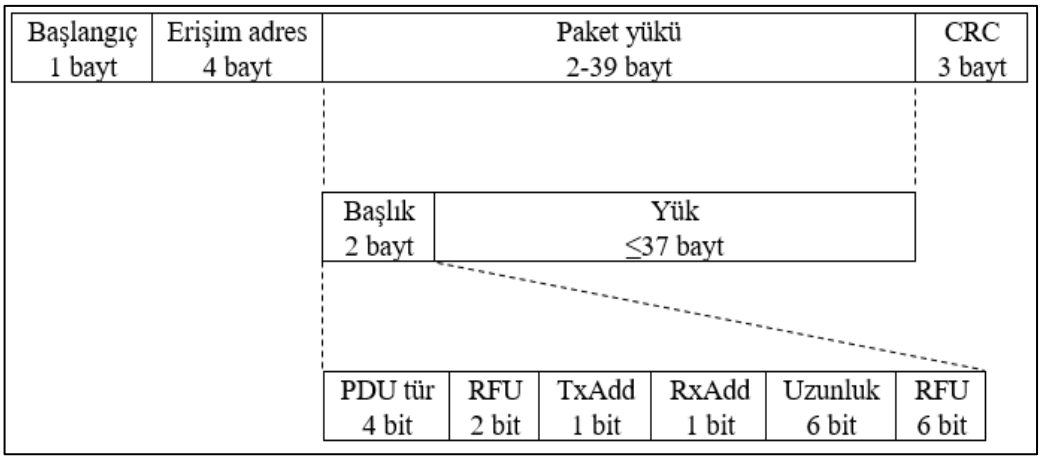
- Karakteristik değer okuma – Alıcı karakteristik değeri okumak için belirli bir algılanma talep eder ve sunucu değerlere yanıt verir. (Hangi verileri okuma izni olduğu varsayılarak.)
- UUID kullanarak karakteristik okuma – Alıcı belirli bir türdeki tüm karakteristikleri okumak için talep eder ve sunucu eşleşen tüm karakteristik ile değerleri algılayıp yanıt verir. Alıcı tarafından bu özelliklerin algılanmasının bilinmesi mecburi değildir.
- Birden fazla karakteristik değer okuma – Alıcı tek talep içinde birkaç tane algılanmadan karakteristik okumak için talep eder. Sunucu alıcıya değerler ile yanıt verir. Bu durumunda, alıcı farklı karakteristiğin değerler arasındaki verilerin ayrıştırmalarını bilmeleri mecburidir.

- Karakteristik tanımlayıcı okuma – Alıcı, belirli bir algılanmadaki karakteristik tanımlayıcıyı okumak için talep gönderir ve sunucu tanımlayıcı değerler ile alıcıya yanıt verir.
- UUID ile karakteristik bulma – Alıcı kendi türüne göre belirli bir karakteristik algılanmasını bulması için talep gönderir. Sunucu karakteristik değeri içeren bildirim ile yanıt verir.
- Karakteristik değer yazma – Alıcı belirli bir algılanmada karakteristik yazma talebini sunucuya gönderir ve sunucu sonuç hakkında belirli bir yanıt verir (Sonuç başarılı olup olmadığına ait.)
- Karakteristik tanımlayıcı yazma – Alıcı belirli bir algılanmada sunucusuna karakteristik tanımlayıcı yazmak için talep gönderir. Sunucu sonuç hakkında belirli bir yanıt verir (Sonuç başarılı olup olmadığına dair).
- Karakteristik değer bildirim – Sunucu, alıcıya karakteristik değerler hakkında bildirim verir. Ancak ilk bildirim için gereken ayarların yapılması gerekir.

Her profil ona karşılık gelen hizmet ile başlar. Her profil ona karşılık gelen bütün profillerin sunucusu olan cihaz üzerine kaydedilir.

4.5. Veri Paketleri

Bu bölümde BLE üzerinden aktarılan verinin nasıl paketlendiği açıklanmaktadır. Bununla birlikte, paketlerin hakkında daha ayrıntılı bilgi verilmektedir. Cihazdan aktarılan veri paketi şekil 4.3'de gösterilmiştir. Başlangıçta, alıcı ile senkronizasyon yapılır. Yani, zaman değerlendirmesi için kullanılan değer 1 bayttır. Bu, yayın paket için her zaman 0xAA olur. Erişim adresi de yayın paketi için 0x8E89BED6'ya sabittir. Paket yükü başlık ve yük diye 2 paketten oluşur.



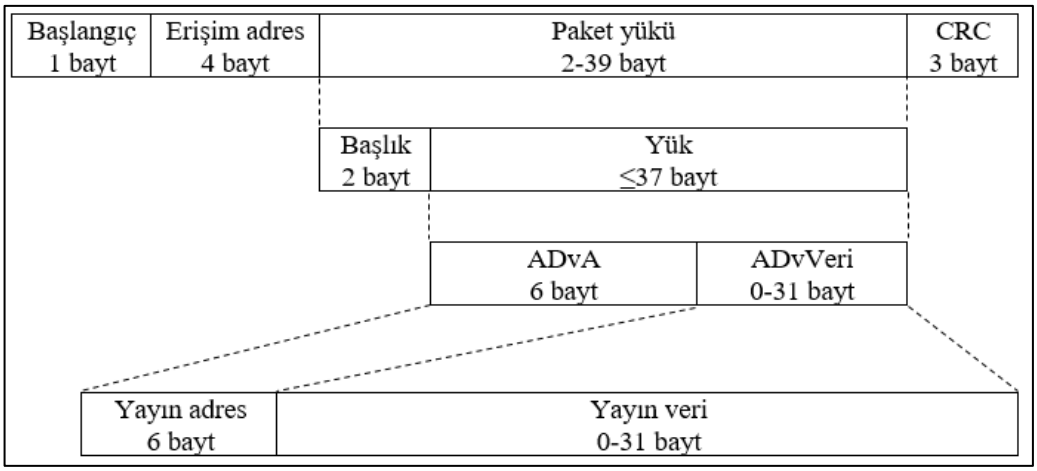
Şekil 4.3. BLE veri paketi

Başlık, paket türünü açıklar ve PDU tip ise cihazın amacını tanımlar. Yayın uygulamalar için üç farklı PDU tipleri vardır ve tablo 4.3'de gösterilmektedir. PDU tipte olan bu üç tip, önceki bölümlerde açıklandığı tarama durumunda yapılan duyuruların yanıtlarıdır.

Tablo 4.3. Yayın tür bağlantı için PDU duyurma tipleri

PDU tip	Paket adı	Açıklama
0000	ADV_IND	Bağlanabilir adresiz duyurma olayı
0010	ADV_NONCONN_IND	Bağlantı olmayan adresiz duyurma olayı
0110	ADV_SCAN_IND	Adresiz taranabilir duyurma olayı

TxAdd bit, duyurma için adresi gösteren bittir. Genel olarak (TxAdd) = 0 veya rastgele (TxAdd) = 1'dir. RFU yedek bittir. PDU tipe göre değişen ve iletilen paketin son bölümü CRC bittir. CRC bit istemeyen değişkenler ve hata tespiti için kullanılan bittir. Bunlar havada iletilen paketlerin veri bütünlüğünü sağlar. Duyuru adres ve kullanıcı tarafından iletilen veri içeren paketin yükü Şekil 4.4'te gösterilmiştir.



Şekil 4.4. BLE yayın veri

BLE veri paketinin yük kısmı, yayın adres ve yayın veriden oluştuğu şekil 4.4'te gösterilmiştir. Bu yayın ağdaki yayın adres ve yayın veridir;

- ADV_IND (0000): Bağlanabilir yönlendirilmemiş duyuru.
- ADV_DIRECT_IND (0001): Bağlanabilir yönlendirilmiş duyuru. Yönlendirilmiş duyuru bir cihaz hızlı bir şekilde başka cihaza bağlanmaya gerektiğinde kullanılır. Bağlantı isteği aldıktan sonra, bir başlatma cihazı hemen bağlantı gönderir.
- ADV_NONCONN_IND (0010): Bağlanmamış yönlendirilmemiş duyuru. Cihaz yayın yapmak için kullanılır bu durumda bağlanabilme veya taranabilme isteğine gerek kalmaz. Dolayısıyla bu seçenek sadece veri aktarmada kullanılır.
- ADV_SCAN_IDN (0110): (eskiden ADV_DISCOVER_IND): Taranabilir yönlendirilmemiş duyuru.
- SCAN_REQ (0011): yapılan işlemi geri alır. Duyuru paketi ve aktif tarama üzerinde bu tarama isteği paketi yayınlanır.
- SCAN_RSP (0100): Tarama isteği (SCAN_REQ) aldıktan sonra duyuru böyle yanıt verir.
- CONNECT_REQ (0101): Bağlantı isteği.

4.6. Nitelik Protokol (ATT)

Nitelik protokol verisidir. UUID değeri taşır. Okuma ve yazma izni ile adres yapılabilir. ATT sunucu ve istemci arasında noktadan noktaya peer-to-peer protokolü yapar. ATT sunucu ve istemci protokolü içindekileri aşağıda belirtilmiştir;

- Sunucu: nitelikleri içerir, istekleri alır, çalıştırır, yanıt verir ve değerleri gösterir

- İstemci: istek gönderir, komutları, yanıtları bekler ve onaylar

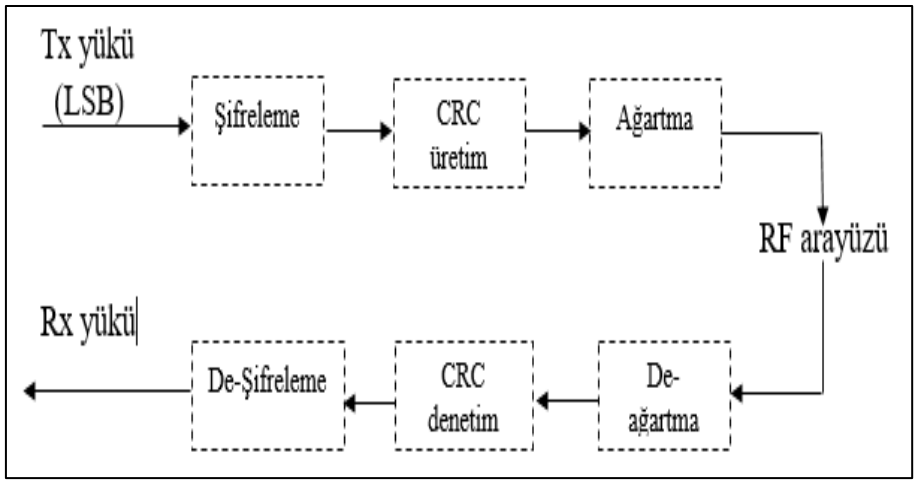
Nitelik üzerinde yapılan işlemler;

- İt: Ayarlar yapıldığında veya değiştiğinde sunucu istemciye veri gönderir.
- Çek: İstemci ihtiyaç duyduğunda sunucudan veri ister.
- Set: Bir sunucu yapılandırması
- Yayın: sunucu sürekli verileri yayınlar (PDU kullanarak bölüm 4.5).
- Alma: istemci nitelikleri ayırmak için ve sunucudan gelen hizmetleri bulması için gerek duyan isteği gönderir.

4.7. Güvenlik Yöneticisi Protokolü (SMP)

BLE iletişim üzerinde verilerin güvenlik yazılımı donanım tarafında yapılmıştır. Aşagıda BLE iletişim üzerinde işlenen aşamalar aşağıda belirtilmiştir;

- AES – 128 CCM Kriptografi
- Donanımsal ve yazılımsal çözüm
- Mesaj bütünlük denetimi her PDU şifrelemede yapılır
- Bit akışı süreci



Şekil 4.5. Bit akışı süreci

Ağartma (whitening) dönüşümü, bir kovaryans matrisinin vektörünü yeni kümesine dönüştüren doğrusal bir dönüşümdür [24]. Güvenlik yöneticisi protokolü yardımıyla bağlantı esnasında ilk olarak merkez cihaz bağlantı isteği gönderir ve yardımcı cihaz ona yanıt verir. Böylece iki cihaz arasında eşleştirme yapıldıktan sonra güvenlik yöneticisi protokolünün (SMP) güvenlik ve tanımlama bilgisi iki cihaz arasında gider.

4.8. Mantıksal Bağlantı Kontrolü ve Adaptasyon Protokolü (L2CAP)

L2CAP yönlü bağlantı (connection-oriented) ve bağlantısız veri hizmeti üst katmanlara sağlamaktadır. Gerçek gönderilmek istenilen veri aktarılmadan önce iki cihaz arasında kontrol paketleri yollar, buna yönlü bağlantı denir. Bu, BLE için aşağıdaki avantajları sağlar;

- Bölümlendirme ve yeniden birleştirme
- Kanal başına akış denetimi ve yeniden iletim (akış kontrolü)
- Güvenli veri aktarma

4.9. Ana Kontrolör Arabirim (HCI)

Ana kontrolör arabirim, Bluetooth denetleyicinin kontrol kısmına ulaşmasını sağlayan birimdir. HCI, donanımlara göre değişebilir. Bazı donanımlarda ayrı ayrı olabilirler. Bu durumlarda ana kontrolör diğer kontrolör ile haberleşmesi için senkron alıcı/verici (USART), SPI ve başka herhangi bir haberleşme arabirimi kullanarak birbiriyle anlaşabilmektedirler. Denemede bir çip üzerine konulmuş BLE modül kullanılmıştır.

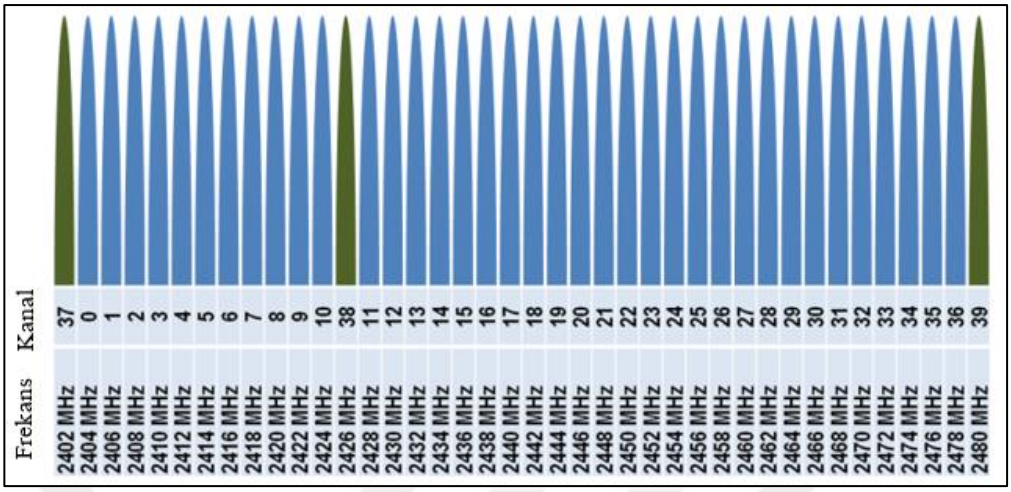
4.10. Bağlantı Katmanı (LL)

Bağlantı katmanı BLE hangi modda çalıştığını anlayıp bir sonraki katmana veri aktarır. LL katmanda karar verildiğine göre fiziksel katmanda olan veri aktarmasının frekansı değişmektedir. Bölüm 4'ün başlangıçta bahsedildiği gibi BLE 2,4GHz ile çalışır ve toplamda 40 kanaldan oluşur. Kanallar, duyuru kanalı ve veri kanalı olmak üzere 2 farklı kanalda bölünür. BLE duyuru modda çalışırken 3 kanal kullanmaktadır. Bunun sayısında BLE daha az enerji harcar ve daha az veri aktarır.

Şekil 4.6'da duyuru ve veri kanalı verilmektedir. Bağlantı katmanında 6 farklı durum oluşmaktadır ve aşağıda anlatılmaktadır;

- Bekleme – cihaz üzerinden hiçbir veri aktarılmaz ve gönderilmez, herhangi bir cihaza bağlı değildir.
- Duyuru – sürekli duyuru yayınlama.
- Tarama – duyuruları aktif olarak arar.
- Başlatma – aktif olarak başka bir cihazla bağlantı kurmaya çalışır.
- Merkez – başka bir cihaza merkez olarak bağlanır.

- Yardımcı - başka bir cihaza yardımcı olarak bağlanır.



Şekil 4.6. Bağlantı katmanı kanallar [24]

Şekil 4.6'daki yeşil renkle çizilen kanal duyuru kanallarıdır. Mavi ile çizilen kanallar veri kanallarıdır [24].

4.11. Fiziksel Katmanı

Bu katman Bluetooth kontroller de işlenen dijital sinyali analog sinyale dönüştürüp hava üzerinden aktarılan iletişimi temsil eder. Bluetooth protokolünün en alt katmanıdır ve bağlantı katmanı ile beraber çalışmaktadır [25]. 2,4 GHz radyo iletişimi kullanarak veri aktarılır. Toplamda 40 kanal üzerinde 2 MHz aralıkla, yani 2,400 GHz'ten 2,4835 GHz arasında çalışır. Alt bölüm 4.10'da değinildiği gibi 2 kanala ayrılmaktadır. BLE radyo 1Mbps ile 1 bit gönderir ve veri küçük parçalara bölünerek hızlı veri gönderilme optimize edilmiştir. BLE, radyo Gauss frekans kaydırmalı anahtarlama (GFSK) kullanır ve böylece taşıyıcı frekansları değiştirmeden önce veri darbeleri Gauss ile filtrelenir. Bunun sayesinde frekans geçişleri daha pürüzsüz olmaktadır.

5. ANDROID STUDIO

Android Studio, IntelliJ IDEA dayanıklı, Android uygulaması geliştirme için resmi Entegre geliştirme ortamıdır [28]. IntelliJ'nin güçlü kod editörü ve geliştirme araçları sağlayan Android uygulamaları oluştururken verimliliği artıran özellikler sunabilmektedir [28]. Android Studio'nun özellikleri aşağıda yazılmaktadır [28];

- Esnek Gradle tabanlı bir yapı sistemi
- Hızlı ve fazla özelliklere sahip olan emülatör
- Tüm Android cihazlar için geliştirebilecek birleşik bir ortam
- Yeni bir APK oluşturmadan çalışan uygulamada değişiklik uygulanabilmekte
- Ortak uygulama özellikleri ve örnek kodu içe aktarılması için GitHub entegrasyonu ve kod şablonları vardır
- Kapsamlı test araçları ve çerçeveleri
- Performansı, kullanışlılığı, sürüm uyumluluğu ve diğer sorunları yakalamak için 'Lint' araçlar vardır
- C++ ve NDK desteği
- Google bulut platformu desteği

5.1. Proje Yapısı (ANDROID)

Android Studio'daki her proje, kaynak kodu dosyaları ve kaynak dosyaları içeren bir veya daha fazla modül içerir [28]. Modül türleri aşağıda yazılmaktadır;

- Android uygulama modülleri
- Kütüphane modülleri
- Google uygulama makine modülleri

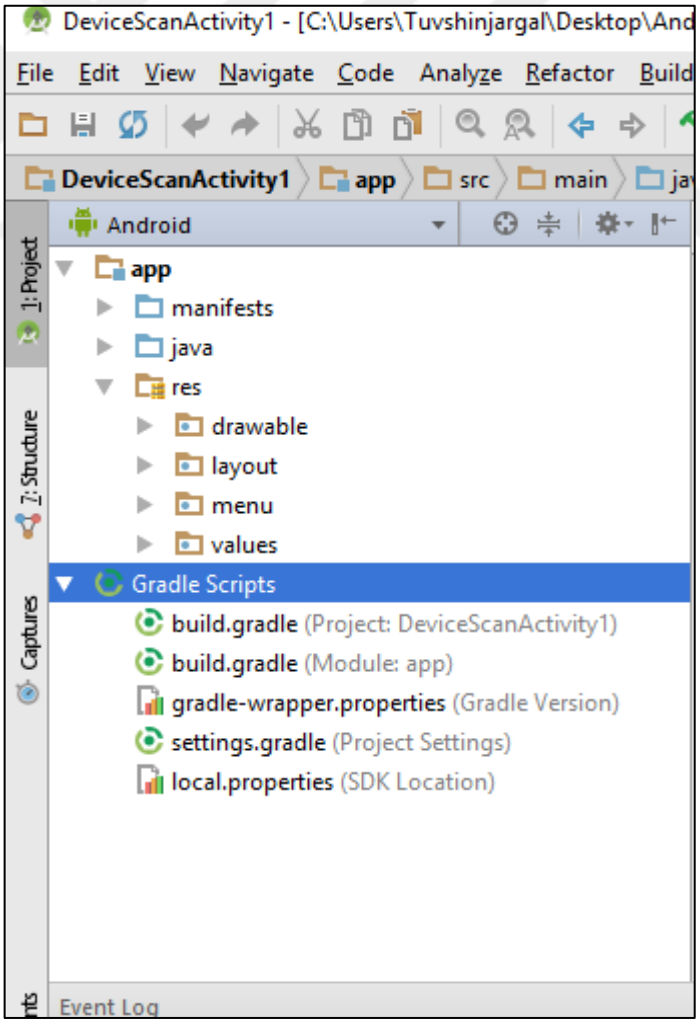
Genel olarak, Android Studio, proje dosyaları Şekil 5.1'de gösterildiği gibi Android proje görünümünde görüntülenir. Şekil 5.1'deki görünüm, projenin kaynak dosyalarına hızlı erişimi sağlamak için modüller tarafından otomatik olarak düzenlenir.

Tüm derleme dosyaları “Gradle Scripts” dosyanın altındaki en üst seviyede görünür ve her uygulama aşağıdaki klasörleri içerir [28];

manifests: AndroidManifest.xml dosyaları içerir. Bunun içinde genel ayarları yazılır. Örneğin, Bluetooh için ona erişebilmesi izni yönetim tarafından etkinleştirilir. Bölüm 5.3’de ayrıntılı anlatılmaktadır;

- java: JUnit test kodu dahil olmak üzere java kaynak kodu dosyalarını içerir
- res: XML düzenleri, bitmap görüntüleri gibi kod’da olmayan tüm kaynakları içerir. Örneğin, Uygulamanın her bir sayfasına ait resimler olabilir

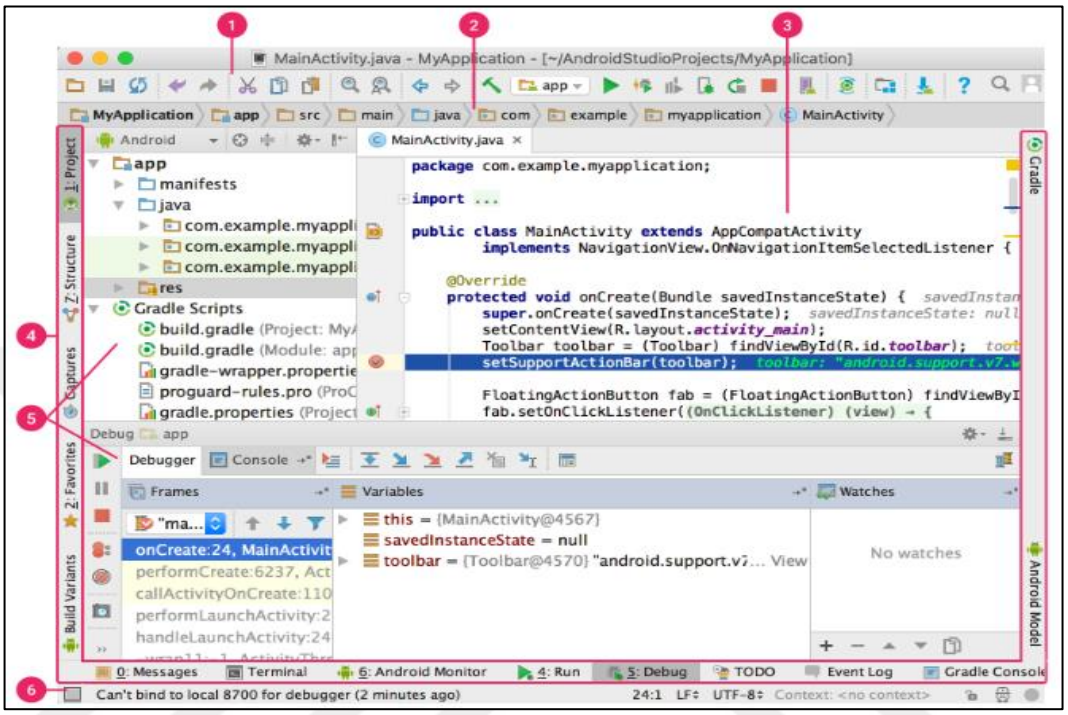
Üstelik kullanıcı isteğe göre proje dosyalarındaki görüntülerde değişim yapabilmektedir. Örneğin, kaynak kodda herhangi bir hata oluştuğunda onları göstermek amacıyla yapılan düzen “xml” dosyaları “AndroidManifest.xml” olabilir.



Şekil 5.1. Android görünümdeki proje dosyaları

5.2. Kullanıcı Arabirim

Android Studio'nin ana pencere şekli 5.2'de tanımlanan çeşitli mantıksal alanlardan oluşur [28].



Şekil 5.2. Android Studio ana pencere [28]

Şekil 5.2'deki kırmızı ile işaretlenen kısımlar aşağıda anlatılmaktadır.

1. Araç çubuğu, uygulamanın çalıştırılması ve Android araçlarının başlatılması da dahil olmak üzere geniş eylemlerin gerçekleştirilmesini sağlar.

2. Gezinti çubuğu, proje üzerinde dolaşma, dosyaları açma ve düzenleme için yardımcı olur. Bu proje penceresi daha kompakt bir görünümünü sağlar.

3. Düzenleyici penceresi, kodun oluşturulup değiştirildiği yerdir. Geçerli dosya türüne bağlı olarak, düzenleyici değişebilir. Örneğin, bir düzen pencere görüntülerken, diğer düzenleyici sabit düzenleyicisini görüntüler.

4.Araç pencere çubuğu, IDE penceresinin etrafında çalışır ve tek tek takım pencerelerinin genişletilmesi veya daraltılmasını sağlayan düğmeleri içerir.

5. Araç pencereleri, proje yönetimi, arama, sürüm kontrolü gibi belirli görevlere erişmenizi sağlar. Bunlar genişletilebilir ve daraltılabilmektir.

6. Durum çubuğu, projedeki durumları, yani uyarıları görüntüler.

Ana pencere kullanıcı tarafından düzenlenebilmektedir. Yani, IDE özellikleri kullanarak klavyenin tuşlarından araçlara hızlıca erişebilmektedir. Tezde kullanılan BLE platformu Android sürümü 4.3 ve sonraki sürümleri destekler.

5.3. Android Uygulamalarda Manifest Ayarı ve Oluşturma

Her uygulama AndroidManifest.xml (bu adına sahip olmak üzere) dosyanın ana yönetiminde bulunur [29]. Manifest dosya, Android sistemde uygulamanın hakkında gerek duyulan bilgileri içerir [29]. Yani, sistem uygulamayı çalıştırmadan önce olması gereken bilgileri sağlar. Manifest dosyada aşağıdakileri içerir;

- Uygulama için Java paketinin adını verir. Projede: com.example.bluetooth.le, bu projenin hangi paketin içinde olduğunu sisteme gösterir. Uygulamada paketin adının tek olması gerekir.
- Uygulamayı oluşturan etkinlikler, hizmet, yayın alıcıları ve içerik sağlayıcıları içeren bileşenleri açıklar. Ayrıca bileşenlerin her birini kapsayan sınıfları adlandırır.
- Uygulama bileşenlerini barındıran işlemleri bilirler. Örneğin: ana kodun niyet mesajlarını belirler.
- API'nın korunan bölümlerine erişmesini sağlar. Diğer uygulamalarla etkileşimde bulunması için uygulamada gereken izinleri vermektedir. Örneğin uygulamada Bluetooth çalıştırabilmesi için izin yazılması gerekir.

Şekil 5.3'de Projede yapılan manifest ayarları gösterilmektedir.

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
          package="com.example.bluetooth.le"
          android:versionCode="1"
          android:versionName="1.0">
    <uses-sdk android:minSdkVersion="18"
              android:targetSdkVersion="18"/>
    <!-- Declare this required feature if you want to make the app available to BLE-capable
    devices only.  If you want to make your app available to devices that don't support BLE,
    you should omit this in the manifest.  Instead, determine BLE capability by using
    PackageManager.hasSystemFeature(FEATURE_BLUETOOTH_LE) -->
    <uses-feature android:name="android.hardware.bluetooth_le" android:required="true"/>

    <uses-permission android:name="android.permission.BLUETOOTH"/>
    <uses-permission android:name="android.permission.BLUETOOTH_ADMIN"/>

    <application android:label="BLE"
                 android:icon="@drawable/emblem_kocaeli"
                 android:theme="@android:style/Theme.Holo.Light">
        <activity android:name="ble.ble.bluetooth.ble.DeviceScanActivity"
                  android:label="BLE"
                  android:screenOrientation="portrait">
            <intent-filter>
                <action android:name="android.intent.action.MAIN"/>
                <category android:name="android.intent.category.LAUNCHER"/>
            </intent-filter>
        </activity>
        <activity android:name="ble.ble.bluetooth.ble.DeviceControlActivity"
            android:screenOrientation="portrait"/>
        <service android:name="ble.ble.bluetooth.ble.BluetoothLeService" android:enabled="true"/>
    </application>
</manifest>
```

Şekil 5.3. Manifest ayarı ve gösterimi

Şekil 5.3'de uygulamada yer alan BLE ve onun hizmetlerine erişebilmesi için gereken izni belirtmektedir. Eğer izin kısmı ve paketleri yoksa uygulama simülasyondayken hata verip izin istemektedir. Yani, Android akıllı cihazlarda olan bluetooth'e erişmesini sağlar.

5.4. Android Uygulamalarda Düzen Ayarı ve Oluşturulması

Bir düzen kullanıcı arabirimin görsel yapısını tanımlar [30]. İki yolla düzen oluşturulabilir;

- Kullanıcı, arayüzü öğelerini XML'de bildirir. Android, gösterim sınıflarına karşılık gelen basit bir XML sunar
- Çalışma zamanında öğelerini tanımlar. Uygulama, gösterim ve nesnelerini yazılım kullanarak oluşturabilir. Burada özelliklerini yönlendirebilir

Düzen uygulamada oluşturulursa, kullanıcılara kullanım için daha iyi bir ortam sağlamakta olup uygulamada işlenen süreçleri daha verimli şekilde çalıştırmaktadır. Yazılımın adım adım çalışmasına yardım eder. Projede 3 tane genel düzen vardır. Bunlar ise, uygulamanın başlamasından bitimine kadar olan sürede kullanıcılara uygulamanın nerede ne yaptığını gösteren içeriktir. Ayrıntıya girerse, uygulama ilk açılırken BLE'nin cihazları bulabilmesi için gereken düzen görülür. Tarama yapıldıktan sonra mevcut olan BLE cihazları gösteren düzene kavuşur. En son düzen ise, cihazlara bağlı olup olmadığına ait bilgileri gösterir.

5.5. Android Uygulamalarda Metin Alan Ayarı ve Oluşturulması

Kullanıcı tarafında uygulamanın her hangi bir değişkenin veya bir kısmının kontrol edilmesini sağlayan kısmi metin alanıdır. Metin alanında yazılan değişken türü yazılımda direkt yazılmalı aksi takdirde doğru türün ne olduğunun gösterildiği kısım eklenmelidir. Android giriş özelliklerinden biri eğer metin girişi tasarlanmışsa otomatik yazım denetimi düzeltmesi "textAutoCorrect"değeriyle etkinleştirilmelidir [31]. Metin alanı xml dosya üzerinde hazır "widget"in yardımıyla oluşur. Bu yazılımcı tarafındaki işleri kolaylaştırır [32]. Örneğin eklenmiş olduğu metin veya herhangi bir "widget"in konumunun değiştirilmesini isterse, xml dosyasının içine girip onun üzerinde kaydırma veya başka bir konuma yerleştirme yapılabilir. Bu "widget"ler yazılımda kendi hakkındaki tüm bilgileri verir ve bunun sayesinde yazılımcı onunla bilgi kurup yönlendirebilir.

5.6. Android Uygulamalarda Düğme Ayarı ve Oluşturulması

Bir düğme, kullanıcı dokunduğunda gerçekleşen eylemi bildiren bir metin veya bir simge içerir [33].Bir düğme metin, simge veya her ikisi ile oluşabilir. Düğme oluşturmanın 3 yolu aşağıda gösterilmektedir;

- Metinle, "Button" sınıfını kullanarak
- Simge ile "ImageButton" sınıfını kullanarak
- Metin ve simge ile "Button" sınıfı "drawableLeft" özelliği kullanarak

Bunların dışında oluşturulmuş olan düğme basılı olup olmadığına ait "OnClickListener" işleyici kullanarak düğmenin o andaki durumunu algılayıp işlem yaptırabilir. Projede 3 tane düğme kullanılmaktadır. Birinci düğme tarama için, ikinci

düğme bağlantı kurması veya bağlantı koparması için, üçüncüsü ise, metin alanında yazılmış olan verileri BLE üzerinden aktarılmasını sağlayan düğmedir.

5.7. Android Uygulamalarda Toasts Ayarı ve Oluşturulması

Bir "toast" uygulamasında olan biten işlemler hakkında basit bir geribildirim sağlar [34]. Yalnızca ileti için gereken alan miktarını doldurur [34]. Aynı zamanda mevcut etkinlik görünür. Yani, her hangi bir işlemin bitmiş olduğuna dair bir "toast" yapılmışsa, o işlem gerçekleştiğinde ona dair bir bilgi verir. Bu işlem basit bir baskı işlemidir. Belirli bir süreden sonra "toast" yazısı otomatik olarak kaybolur. Düğme ve metin gibi konum değiştirme fonksiyonları vardır. Ancak "toast" yazılımda yazılarak gerçekleşir. Eğer kullanıcı tarafından durum mesaja her hangi bir yanıt verilmesi gerekiyorsa bunun yerine bildirim kullanılabilmektedir [35]. İlk olarak, Toast nesne, makeTest() yöntemi ile çalışır. Bu yöntemin içinde 3 tane parametre vardır: uygulama denen bağlam, metin mesajı ve toast'un süresidir. Çalıştığı anda bir toast nesneye dönüşür. Aşağıdaki örnekte gösterildiği gibi toast bildirimini show() ile görüntüleyebilir. Aşağıda görünen örnek projenin android kısmından alınmış koddur. Örnek;

Toast.makeText(DeviceControlActivity.this, "No characters have been written, please write character", Toast.LENGTH_SHORT).show();

Yukarıda yazılan örneği ise, Kullanıcı tarafından herhangi bir karakter yazılmadan gönderme düğmesine basılırsa, kullanıcıdan karakter yazmasını isteyen küçük bir pencere çıkıp görüntülenmektedir. Android'daki diğer yöntemlerin içerdiği gibi konum ve boyut yöntemleri vardır.

5.8. Android Uygulamalarda LOG nedir? Nasıl Oluşur?

Android Monitor, hata ayıklama mesajlarını görüntüleyen bir "logcat" ekran içerir [36]. Genelde Log.v(),Log.d(),Log.i(),Log.w() ve Log.e() yöntemleri kullanılır [36]. "Log"lar sınıflarına göre farklı uygulama işlemleri yapar. Yani, ilgilenen bilgileri görüntülemek için, filtreler oluşturabilir, mesajlarda ne kadar bilginin görüntülenebileceğini değiştirebilir, öncelik düzeyini ayarlayabilir, yalnızca uygulama kodu ile üretilen mesajları görüntüleyebilir [36]. "Log" mesajlarını farklılığı aşağıda gösterilmektedir [36];

- v ve verbose – tüm “log” mesajlarını gösterir.
- d ve debug – geliştirme sırasında düşük ileti düzeylerini gösterir.
- i ve info – Düzenli kullanım için beklenen “log” mesajını gösterir.
- w ve warn – henüz hata olmayan olası sorunları gösterir.
- e ve error – hatalı nedenli olan sorunları gösterir.
- a ve assert – yazılımcının beklemediği sorunları gösterir.

5.9. Android Cihazlarda Bluetooth Düşük Güç Bağlantısı

Android 4.3 (API seviye)’den başlayan sürümler Bluetooth düşük enerjisini destekler. Cihazı keşfetmesi, bağlantı kurulabilmesi, hizmetler için sorgulama yapmak ve karakterleri okuma/yazma için kullanılır [37]. Klasik Bluetooth’un aksine, BLE daha düşük güç tüketimi sağlamak üzere tasarlanmıştır [37]. Dolayısıyla Android uygulamalar BLE cihazıyla bağlantı kurabilmektedir. Bunlar düşük güç gereken yakınlık algılayıcıları, kalp atışı hızı denetleyici ve spor cihazlarıdır. Bağlantı kurmanın 4 basamağı vardır;

1. Bluetooth Adaptörü alması: Android API, temel Bluetooth görevleri tamamlanan Bluetooth adaptörünü içeren sınıfa sahiptir. Yani, Bluetooth adaptörü BLE ile desteklenip desteklenmediğini belirler. Bu aşamada uygulama, android cihazın Bluetooth’u destekleyip desteklenmediğine dair bilgiyi kullanıcıya verir. Eğer desteklerse 2’inci aşamaya geçer.
2. BLE cihazı Bluetooth adaptörünün “scanLeDevice” yöntemiyle bulma: Eğer kullanıcı tarama düğmesini etkinleştirirse, “scanLeDevice” yöntem BLE cihazlarını tarar ve bulunan cihazları geri gönderir. 10 saniye devamlı tarar veya kullanıcı kendi isteğiyle taramayı sonlandırabilir. Bunu uygulandığında enerji tasarrufu elde etmiş olur. Tarama sonucunda bulunan cihazları sırayla ekler ve kullanıcıya gösterir. Bu aşamada kullanıcı tarafından bulunan cihazlardan birisini seçip bağlantı kurulabilmektedir. Kullanıcı bağlantı kurarsa, 3’üncü aşamaya geçilir.
3. GATT- sunucusuna bağlanır: Bağlantıyı etkinleştirmenin son aşamasıdır. Bundan sonra kullanıcı için tasarlanmış veri gönderme penceresi etkinleşir. Kullanıcı tarafından yazılan veri ve karakterler, gönderme düğmesine basıldıktan sonra AES ile şifrelenip BLE cihazına aktarılmaktadır.
4. GATT sunucusu kapanması: Uygulama BLE cihazını kullanmayı tamamladığında, sistemi serbest bırakabilmesi için “close” yöntemi çağırılıp çalıştırılmalı.

6. BLE ve AES'in ÇALIŞMASI

Bu bölümde önceki bölümlerde çıkmış olan teorileri birleştirerek bütün bir sistem olarak çalıştırıp inceleceğiz. Sistemin donanımlarına; 1, 2 ve 5'inci bölümlerde değinilmiştir. TinySine BLE modül, STM32F407 ile USART1'e bağlanmıştır ve ayarı yapılmıştır. Bu kısmında, devre ve yazılmış olan algoritma denetlemeye başlayacaktır. Akıllı telefondan gelen verilerin hatasız gelip gelmediğine bakmak için bilgisayar üzerindeki seri bağlantı, STM32F4'nin USART3'a bağlanıp yan taraftan kontrol edilmesi sağlamıştır. Aynı zamanda Android akıllı cihaz ve TinySine BLE modül arasında BLE bağlantısı kurulup veri aktarılmaktadır. Alt bölüm 6.1 ve 6.2'lerde Android akıllı cihazlarında işlenen uygulamanın çalışma mantığını ve uygulamanın içindeki yazılımın algoritmasını anlatmaktadır. Alt bölüm 6.3'de STM32F4 denetleyici içinde çalışan algoritmayı anlatmaktadır. Alt bölüm 6.4 ve 6.5'lerde Android akıllı cihazın ve STM32F4 denetleyiciler arasında aktarılan şifrelenmiş verilerin zaman analizini anlatmaktadır.

Uygulama'da kullanılan AES şifreleme standardı ile verinin en kısa zamanda işlenmesi için ve iki aygıt arasında verinin en kısa zamanda iletilmesini sağlamak için yazılımda aşağıdaki iki yön takip edilmiştir;

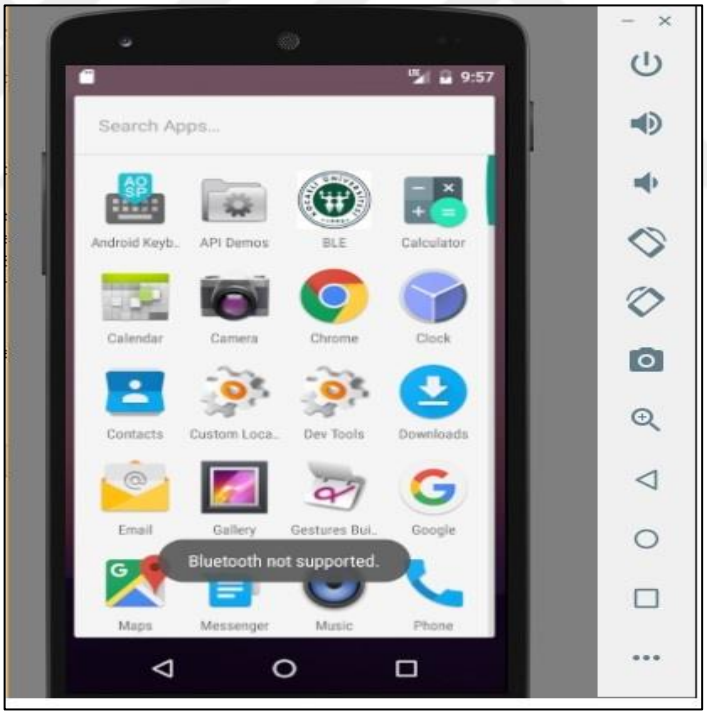
- Gelişmiş şifreleme standardı algoritma çalışırken en kısa zamanda işlenecek şekilde kod yazılması için matris şekli kullanılmıştır.
- Diğer aygıta veri aktarılırken en az zaman harcanarak şekilde aktırılmasını sağlamak.

Bölümün sonunda, BLE üzerinde iletilen verilerde gelişmiş şifreleme standardı algoritma uygulanarak, veri şifrelenirken harcanan zaman ve iletirken harcanan zamanı şifrelenmemiş verilerde harcanan zamanla karşılaştırılmıştır.

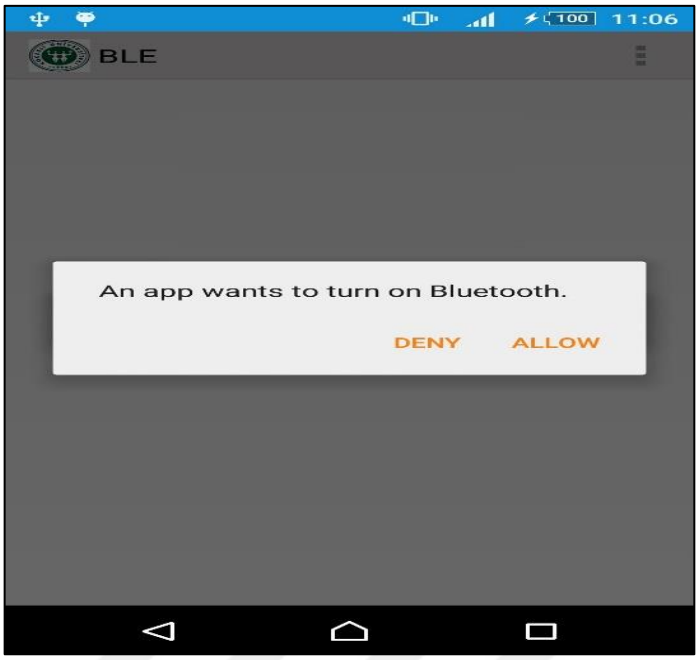
6.1. Android Akıllı Cihazda Uygulama Nasıl Çalışır ve Veriler Nasıl Gönderilir?

Bu bölümde android akıllı cihazlarda yazılan algoritmanın nasıl çalıştığı anlatılmaktadır. Algoritma içinde kullanıcı grafik arayüzünü de kapsamaktadır. Kullanıcıda gösterilen ara sayfalar ve algoritmayı sırayla gösteren yazılar aşağıda yazılmaktadır.

Android akıllı cihazın, BLE'yi destekleyip desteklemediğini belirtir. Desteklenmiyorsa şekil 6.1'de gösterilmiş olduğu sayfa kullanıcıda görünmektedir. Aksi takdirde, bir sonraki adıma geçer. Böylece kullanıcılarda uygulama hakkında ve kendi cihaz hakkında kısaca bilgi edinmektedir. Android akıllı cihazda, BLE desteklenirse, onu etkinleştirmesi için kullanıcı tarafından izin istenir. Şekil 6.2'de gösterilmiştir.

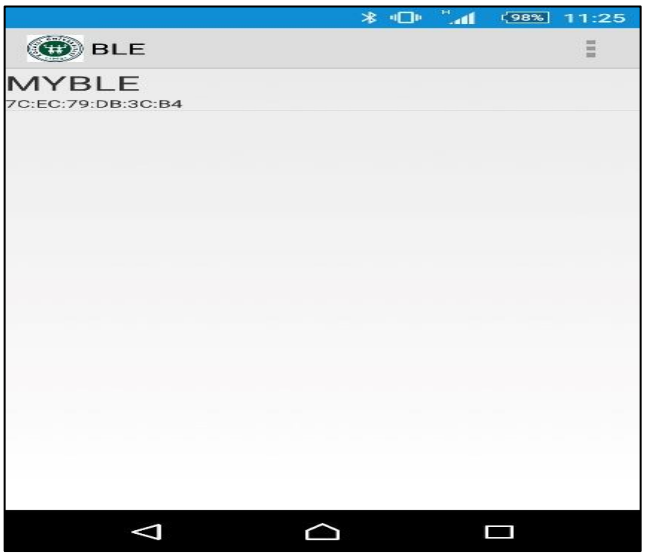


Şekil 6.1. Uygulamanın ilk önlemesi android akıllı cihaz Bluetooth desteklenmiyorsa, görünür



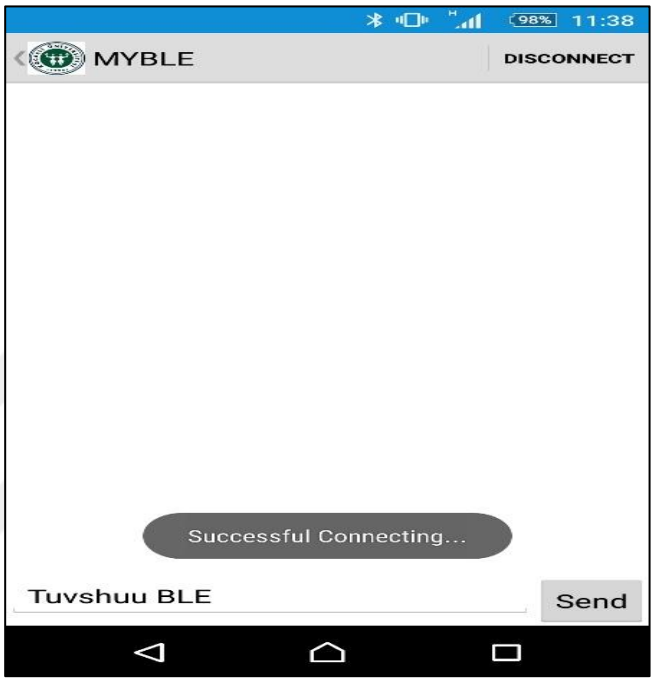
Şekil 6.2. Bluetooth açması için izin isteyen pencere

- BLE etkinleştirildikten sonra mevcut olan BLE cihazları taraması için uygulamada yapılan sayfa gösterilir.
- Tarama düğmesine basıldığında mevcut olan BLE cihazlar görünür ve bağlanmak istediği cihazı seçip bağlantı kurabilir. Şekil 6.3'de tarama düğmesine basıldıktan sonraki hali gösterilmektedir.



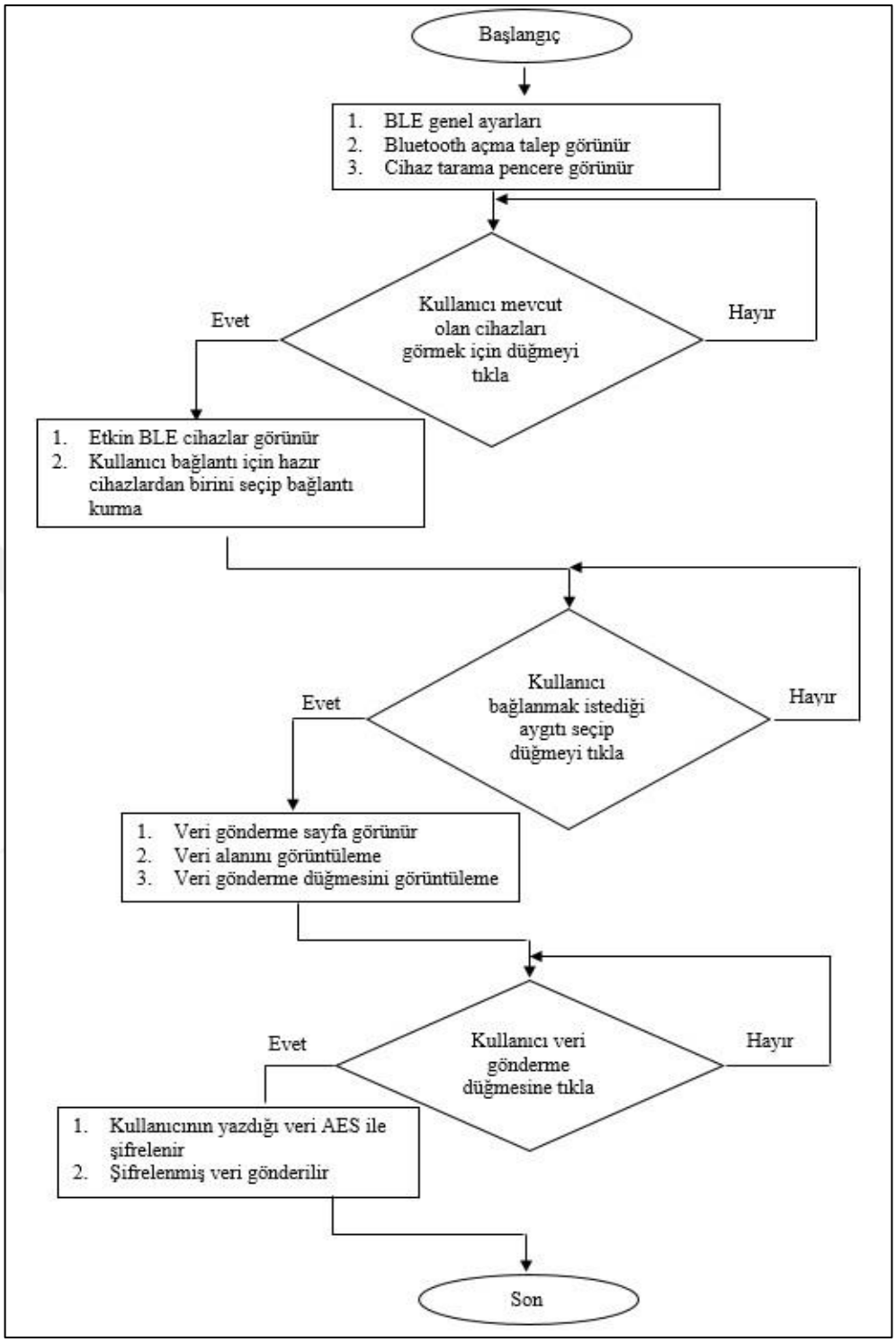
Şekil 6.3. BLE bağlantı için mevcut olan cihazları gösteren pencere

- Bağlantı kurulduktan sonra göndermek istediği verileri yazıp, gönderme komutu verildiğinde veriler AES ile şifrelenip diğer BLE aygıtına aktarılır. Bölüm 6.2'de aygıtta çalışan şifreleme algoritmasından bahsedilecektir. Şekil 6.3'de bağlantı kurulduktan sonra kullanıcıda görünecek olan pencereyi göstermektedir. Bağlantı kurulduğuna dair anlık bilgi görünür.



Şekil 6.4. Bağlantı kurulduktan sonraki pencere

Yukarıdaki adımlar gerçekleştikten sonra kullanıcı, metin alanında olan başlangıç yazısını "Tuvshuu BLE" silip göndermek istediği veriyi yazıp aktarabilmektedir. Gönderme düğmesine basıldıktan sonra kullanıcının yazdığı metin uygulamanın içinde yazılan AES algoritması ile şifrelenip gönderilmektedir.

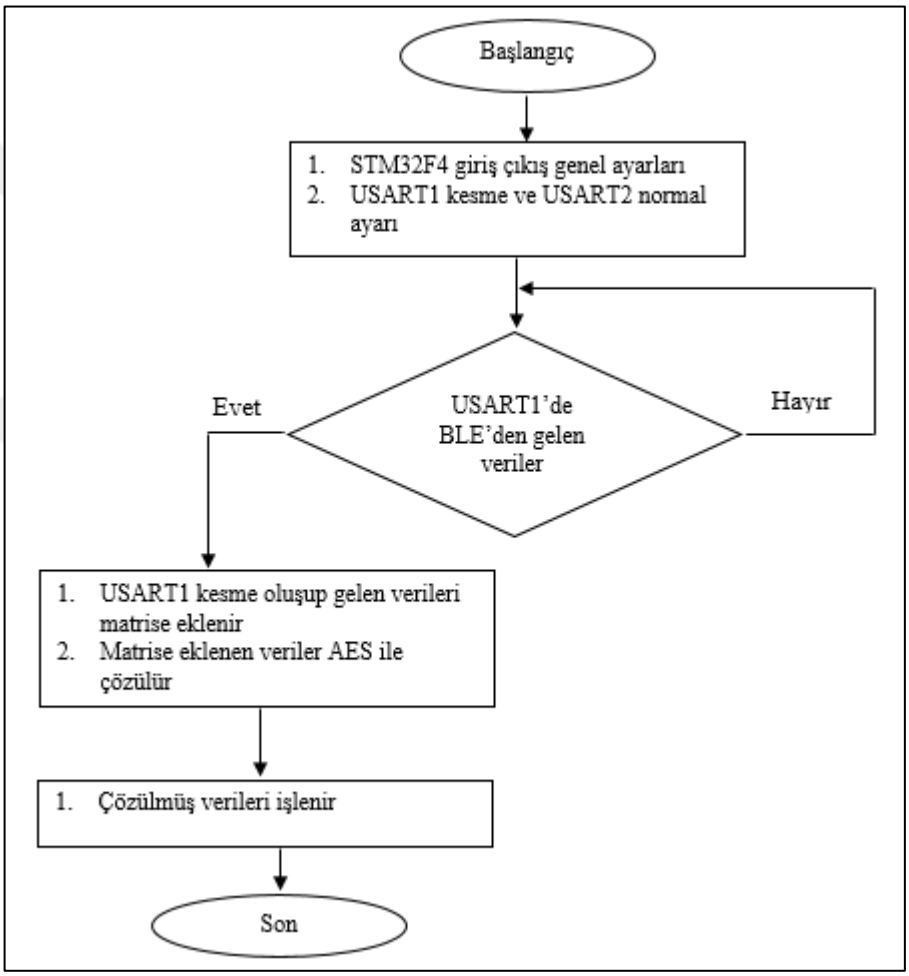


Şekil 6.5. Android akıllı cihazda gerçekleşen uygulamanın akış diyagram

Android akıllı cihazda çalışan uygulamanın akış diyagramı şekil 6.4'te gösterilmektedir. Algoritmada kullanıcı Bluetooth etkinleştirilmesi talep göründüğü anda onun etkinleştirildiği varsayılmıştır.

6.2. BLE üzerinden Aktarılan Veriler STM32F4'te Nasıl Alınır

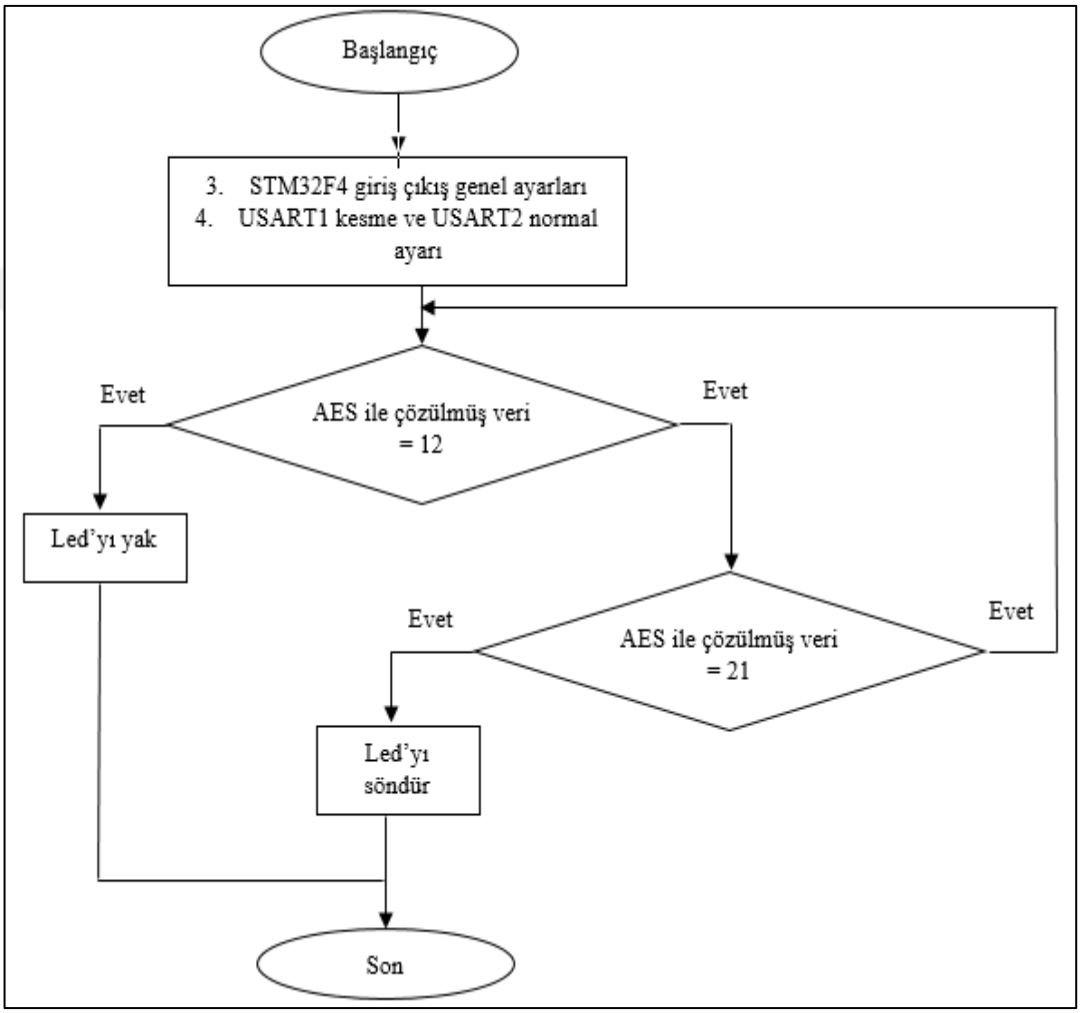
Bu bölümde STM32F4 mikro denetleyicide işlenen algoritmayı anlatmaktadır. Android akıllı cihazında BLE üzerinden gönderilen veriler TinySine BLE tarafından teslim alınıp kendi USART modül üzerinden STM32F4'nun USART'ya aktarmaktadır. Projenin bağlantı blok diyagramı bölüm 2'de gösterilmiştir. STM32F4'ne aktarılan verileri kaçırmamak için USART'A dış kesme kullanılmıştır. Sonunda gelen veriler toplanıp AES ile çözülüp işlenmektedir. STM32F4'te çalışan yazılımın algoritmasını şekil 6.6'te gösterilmektedir.



Şekil 6.6. STM32F4'te işlenen algoritmanın akış diyagramı

İşlenen algoritmada STM32F4 ikinci USART'ın çalıştığını göstermemektedir. Çünkü STM32F4'nun USART3, BLE üzerinden giden gelen verilerin sadece kontrol edilmesi için çalıştırılmaktadır. Bu yüzden kullanıcı istediği zaman BLE üzerinden aktarılan verileri görmek istiyorsa, direkt STM32F4'nun USART3'yı bilgisayarın USART'ına

bağlanıp görebilmektedir. Bununla aynı zamanda USART3'ün kesmesinin dışında çalışan ana kodda BLE üzerinden gelen verilere göre onun üzerinde olan ledleri yanıp söndürmektedir. Bu ana kodun algoritması şekil 6.5'de görünmektedir. Ana kodun amacı ise BLE üzerinden aktarılan verilerin doğru düzgün çözülüp işlendiğini göstermektedir.



Şekil 6.7. AES ile çözülen verilere göre led'lerin çalıştırma akış diyagramı

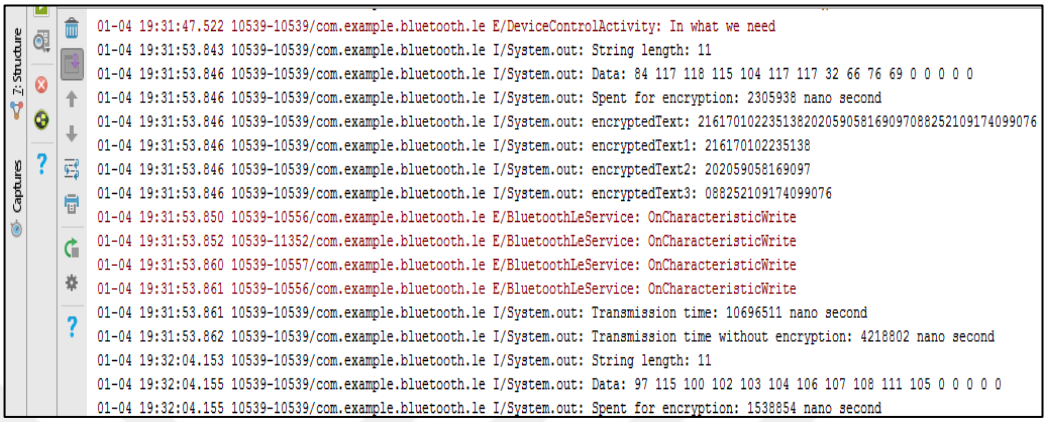
6.3. Android Cihazda AES Şifreleme Zaman Analizi

Bu bölümde Android cihazındaki veriler BLE üzerinden gönderilmeden önce AES ile şifrelenirken ne kadar zaman harcandığını anlatmaktadır. Android akıllı cihazında Sony Z2 seçilmiştir. Sony Z2'nin özellikleri aşağıda yazılmaktadır;

- Android 5.1.1
- CPU Quad-core 2,3 GHz
- 16GB RAM + 3GB dahil

Aşağıda yapılan deneylerin her biri 3 defa yapılmıştır. Deneylerin ortak ölçümü alıp yazılmıştır. Deneylerde şekiller alınırken;

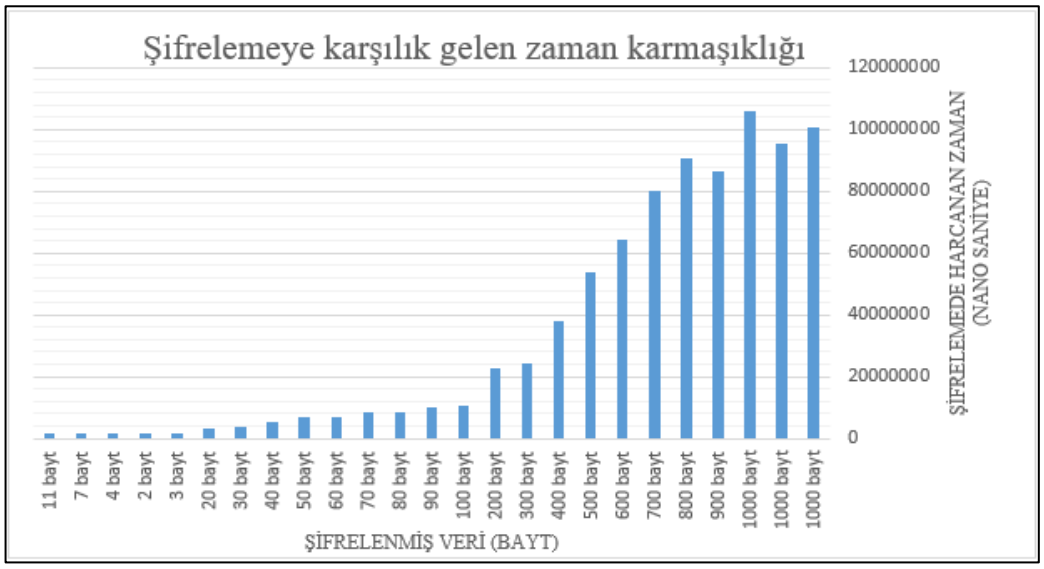
DENEY 1:11 bayt veri gönderilirken;



Şekil 6.8. 11 bayt gönderilirken alınan pencere

- Şifreleme yapılmadan veri gönderilirken: 42188802 nano saniye harcanmıştır.
- Şifreleme yaparken: 1538854 nano saniye harcanmıştır

Şekil 6.8'da BLE üzerinden veri gönderilirken ne kadar zaman harcandığını ve şifreleme yapılırken ne kadar zaman harcandığını göstermektedir. Bundan sonraki deneylerin sonuçları şekilsiz yazılmaktadır. Verileri miktar ve veri miktara düşen zaman karşılığını toplayarak şekil 6.9'de gösterilmektedir.



Şekil 6.9. Şifrelemeye karşılık gelen zaman karmaşıklığı

Şekil 6.9'da Android akıllı cihaz Sony Z2'de BLE üzerinden aktarılan verilerde AES şifreleme uygulanarak farklı boyutta verileri şifrelenerek ona karşı gelen zamanı gösterilmiştir. Tablo 6.1'de Android akıllı cihaz Sony Z2'de BLE üzerinden aktarılan verilerde şifreleme yapmadan gönderilirken ve şifreleme yapıp gönderilirken ne kadar zaman harcadığını karşılıklı göstermektedir.

Tablo 6.1 Android akıllı cihazda BLE üzerinden aktarılan verilerde uygulan AES şifreleme denetimleri

Veri miktar (Bayt)	Şifreleme yapılmadan veri gönderilirken harcanan zaman (nano saniye)	Şifreleme yapılırken harcanan zaman (nano saniye)
2	1804948	1681302
3	2224740	1818177
4	920260	1518594
7	10160209	1737136
20	1552865	3377083
30	3048022	3486950
40	3061612	5454530
50	4556771	6754164
60	4187292	6827916
70	4428438	8406510
80	7056401	8442705
90	7656412	10087812
100	8056425	10630158
200	16521621	22582638

Tablo 6.2 (Devam) Android akıllı cihazda BLE üzerinden aktarılan verilerde uygulan AES şifreleme denetimleri

300	24168810	28853286
400	40731640	37964850
500	498089200	54033312
600	62759400	64163532
700	68099325	79999788
800	72199440	90908850
900	80530290	86559857
1000	104599050	105922026
1000	114689025	95688432
1000	106564896	100922026

7. SONUÇLAR VE ÖNERİLER

Tez çalışması için öncelikle AES şifreleme algoritması ve BLE hakkında araştırmalar yapılmıştır. Daha sonra BLE'de AES uygulanarak literatür çalışması yapılmıştır. Böylece, en az bir haberleşmenin gerçekleştirilmesi için iki tarafa ihtiyaç duyulduğundan bir tarafta TinySine BLE modül ile STM32f4 üzerinde AES uygulanmakta ve diğer tarafta Android akıllı cihazında BLE ile AES uygulanarak simülasyon yapılmıştır. Simülasyonda BLE haberleşmede AES uygulanırsa ne kadar zaman harcandığına dair bilgi alınmıştır. Akabinde; AES, BLE haberleşmede uygulanılırsa, uygulanılmadığı halinden ne kadar daha fazla zaman harcandığını simülasyon da değerlendirilip zaman karşılığı alınmıştır. Aynı zamanda şuanda kullanılan cihazlarda AES'e ne kadar yük verildiği tespit edilmiştir.

Simülasyon yapılırken farklı miktarlarda veriler üzerinde zaman analizi yapılmıştır. Verilerin miktar büyümesine rağmen ona karşılık gelen zaman harcama lineerliği artırılmıştır. Çünkü 16 bayt veya ondan küçük miktarlarda olan veriler üzerinde işlenen adım aynı adımla gerçekleştirilir. Yani, 32 bayt veride AES ile şifreleme yapılırsa 16 bayt veride'de şifreleme yapılırken harcanan zamandan yaklaşık olarak 2 kat fazla zaman harcandığı simülasyonda görülmüştür.

İkinci bir konu, android akıllı cihazı ve STM32F4 geliştirme üzerinde AES uygulanırsa her iki cihaz için ne kadar gecikme olduğuna dair analiz yapılmıştır. Buradan, android cihazlar GHz hızında çalıştığı için, STM32F4 geliştirme board MHz hızında çalıştığı için AES her iki cihazda'da o kadar fazla yük oluşturmamıştır. Yani, kullanıcılarda veri aktarılırken hiçbir gecikme fark edilmemiştir. Dolayısıyla, AES'in haberleşmelerde kullanılması hem verimliliği hem de veri güvenliği artırır.

KAYNAKLAR

[1] Joan D., Vincent R., *The Design of Rijndeal*, 3rd ed., Springer-Verlag, Berlin, 2002.

[2] https://www.mouser.com.tr/new/stmicroelectronics/stm32f4discovery/, (Ziyaret tarihi: 10 Ağustos 2016)

[3] ST, *Stm32f405xx stm32f407xx*, 1st., STM, Shanghai, 2016.

[4] ST, *Reference manual*, 1st., STM, Shanghai, 2017.

[5] ST, *User manual discovery kit with STM32F407VG MCU*, 1st ed., STM, Shanghai, 2016.

[6] STM, *Programming manual*, 1st ed., STM, Shanghai, 2016.

[7] http://www.keil.com/arm/mdk.asp, (Ziyaret tarihi: 3 Ekim 2016).

[8] https://www.arm.com/products/tools/software-tools/mdk-arm/index.php, (Ziyaret tarihi:10 Ekim 2016).

[9] TinySine, *Bluetooth 4.0 BLE module User Manual*, 1.1st ed., TinySine Electronics, Anhui, 2015.

[10] TinySine, *Serial Bluetooth 4.0 Smart Ready dual-mode module User Manual*, 1.1st ed., TinyeSine Electronics, Anhui, 2014.

[11] TinySine, *Bluetooth Low Energy (BLE) 1.20*, 1st ed., Cypress Semiconductor Corporation, San Jose, 2015.

[12] Nicholas G. M., *Past present and future methods of cryptography and data encryptıon*, 1st ed., Utah University, Utah, 2009.

[13] Nigel S., *Cryptograhy: an ıntroductıon*, 3rd ed., Mcgraw-Hill College, University of Bristol, Bristol, 2013.

[14] Joan D., Vincent R., *AES submission document on Rijndael*, 2nd ed., ESAT-COSIC, Belgium, 1999.

[15] FIPS 197, *National Institute of Standards and Technology*, 1st., Springer-Verlag, Belgium, 2001.

[16] Rudolf L., *Polynomials over Finite Fields*, Harald Neiderreiter, *Introduction to finite fields and their applications*, 1st ed., Press Syndicate, Melbourne, 76-81, 1986.

[17] Alfred J. M., Paul C. O., Scott A. V., *Finite Fields,* Ronald L.Rivest, *Handbook of Applied Cryptography,* 1st ed., CRC Press, New York, 81-83, 1997.

[18] Avi K., *The Advanced Encryption Standard*, Avi Kak, *Computer and Network Security,* 1st ed., Avinash Kak, Purdue, 10-15, 2016.

[19] Murat A., Ali A. S., Atlamalı Aralık Yayın Şifreleme Sisteminde Bedava Alıcıların İyi Yerleştirilmesi, *IV. Ağ ve Bilgi Güvenliği Sempozyumu*, Atılım Ünivsersitesi-Orhan Zaim Konferans Salonu, Ankara, 25-26 Kasım 2011.

[20] Texas Instrument, *LPRF San Diego Bluetooth Low Energy Deep Dive*, 2nd ed., Texas Instruments, Dallas, 2011.

[21] Robin H., *Blueooth Low Energy The Developer's handbook*, 1st ed., Crawfordsville, Indiana, 2012.

[22] http://www.bluetooth.com/Pages/Fast-Facts.aspx, (Ziyaret tarihi: 10 Agustos 2016).

[23] http://microchip.wikidot.com/wireless:ble-link-layer-channels, (Ziyaret tarihi: 9 Eylül 2016).

[24] Tervakangas S., Bluetooth Low Energy Development Environment, Bachelor's Thesis, Oulu University, Institute of Science and Technology, 2013.

[25] https://developer.bluetooth.org/TechnologyOverview/Pages/GATT.aspx, (Ziyaret tarihi: 13 Eylül 2016).

[26] https://www.bluetooth.com/specifications/bluetooth-core-specification, (Ziyaret tarihi: 9 Eylül 2016).

[27] https://developer.android.com/studio/intro/index.html, (Ziyaret tarihi: 10 Eylül 2016)

[28] https://developer.android.com/guide/topics/manifest/manifest-intro.html, (Ziyaret tarihi: 11 Eylül 2016).

[29] https://developer.android.com/guide/topics/ui/declaring-layout.html, (Ziyaret tarihi: 11 Eylül 2016).

[30] http://glosbe.com/en/en/layout%20container, (Ziyaret tarihi: 11 Eylül 2016).

[31] http://developer.android.com/guide/topics/ui/controls/text.html, (Ziyaret tarihi: 12 Eylül 2016).

[32] http://developer.android.com/guide/topics/ui/controls/button.html, (Ziyaret tarihi: 12 Eylül 2016).

[33] http://developer.android.com/guide/topics/ui/notifiers/toasts.html, (Ziyaret tarihi: 13 Eylül 2016).

[34] http://developer.android.com/guide/topics/ui/dialogs.html, (Ziyaret tarihi: 12 Eylül 2016).

[35] http://developer.android.com/studio/debug/am-logcat.html, (Ziyaret tarihi: 13 Eylül 2016).

[36] https://developer.android.com/guide/topics/connectivity/bluetooth-le.html, (Ziyaret tarihi: 10 Eylül 2016).

[37] Simon J., *Head First Android Development*, 1st ed., O'Reilly Media, Sebastopol, 2011.

EKLER

EK-A:

Tablo A.1. AT komutlar

Gönderilen komut	Geri gelen komut/dönüş	Görev	Parametre
AT	OK	Deneme için	Yok
AT+ADDR?	OK+ADDR: MAC adres	Cihazın MAC adresi görmek için	Yok
AT+ADTY?	OK+Get:[para]	Duyuru türü görmek için	Para:0~3 0: Herhangi bir cihazla bağlanabilir 1: En son başarılı bağlanmış cihazla bağlanabilir (28s) sonra 2: yayın ve tarama izin 3: sadece duyurma
AT+ADTY[para]	OK+Set:[para] 0: (seçilmiş)	Duyuru türü seçmek için	
AT+ADVI?	OK+Get:[para]	Duyurunun zaman aralığını görmek için	Para: 0~F 0: 100ms 1: 152,5ms 2: 21125ms 3: 318,75ms 4: 417,5ms 5: 546,25ms 6: 760ms 7: 852,5ms 8: 1022,5ms 9:1285ms A: 2000ms B: 3000ms C: 4000ms D: 5000ms E: 6000ms F: 7000ms
AT+ADVI[para]	OK+Set:[para] 0: (seçilmiş)	Duyuru zaman aralık	
AT+BAUD?	OK+Get:[para]	USART hızı görmek için	Para: 0~8 0: 9600

Tablo A.1. (Devam) AT komutlar

AT+BAUD[para]	OK+Set[para] 0: (seçilmiş)	USART hızı seçmek için	1: 19200 2: 38400 3: 57600 4: 115200 5: 4800 6: 2400 7: 1200 8: 230400
AT+COMI?	OK+Get:[para]	En az bağlantı katmanın bağlantı	
AT+COMI[para]	OK+Set[para] 3: (seçilmiş)	En az bağlantı katmanın bağlantı aralığı seçmek için	Para: 0~9 0: 7,5ms 1: 10ms 2: 15ms 3: 20ms 4: 25ms 5:30ms 6: 35ms 7: 40ms 8: 45ms 9: 4 saniye
AT+COMA?	OK+Get[para]	En fazla bağlantı katmanın bağlantı aralığı görmek için	Para: 0~9 0: 7,5ms 1: 10ms 2: 15ms 3: 20ms 4: 25ms 5:30ms 6: 35ms 7: 40ms 8: 45ms 9: 4 saniye
AT+COMA[para]	OK+Set[para] 7: (seçilmiş)	En fazla bağlantı katmanın bağlantı aralığı seçmek için	
AT+MODE?	OK+Get:[para]	Çalışma modayı görmek için	Para: 0~2 0: iletim moda 1: iletim moda + PIO moda 2: iletim moda + uzaktan kumanda
AT+MODE[para]	OK+Set[para] 0: seçilmiş	Çalışma modayı seçmek için	
AT+NAME?	OK+NAME[para]	Cihazın adı görmek için	

Tablo A.1. (Devam) AT komutlar

AT+NAME[MyBLE]	OK+Set[para] [MyBLE] yazılmıştır	Cihazın adı seçmek için	Para: modül adı, en fazla 11 karakter
AT+PASS?	OK+Get[para]	Cihazla bağlantı kurmak için gereken şifreyi görmek için	Para: 000000 ~ 999999
AT+PASS[para]	OK+Set[para] 000000: seçilmiş	Cihazla bağlantı kurmak için gereken şifreyi koymak için	
AT+PWRM?	OK+Get[para]	Uyku moda bakmak için	Para: 0~1 0: otomatik uyku 1: otomatik uyuma
AT+PWRM[para]	OK+Set[para] 1: seçilmiş	Uyku moda seçmek için	
AT+POWE?	OK+Get[para]	Modül güç bakmak için	Para: 0~3 0: -23dbm 1: -6dbm 2: 0dbm 3: 6dbm
AT+POWE[para]	OK+Set[para] 2: seçilmiş	Modül güç seçmek için	
AT+RESET	OK+RESET	Cihazı tekrar başlatmak için	Yok
AT+ROLE?	OK+Get[para]	Hangi rolde çalıştığını bakmak için	Para: 0,1 0: Çevresel 1: Merkez
AT+ROLE[para]	OK+Set[para] 0: seçilmiş	Hangi rolde çalışacağını seçmek için	

KİŞİSEL YAYIN VE ESERLER

[1] Ertürk S., **Ulzii-Utga T.**, Implementation and complexity analysis of embedded advanced Encryption standard (AES) in bluetooth low energy, *Communication journel Izvestıya No.41 of KSTU named after I.Razzakov*, DOI: 621.397.132.122.

ÖZGEÇMİŞ

1992 yılında Ulaanbator'da doğdu. İlk, orta ve lise öğrenimini Ulanbator'da tamamladı. 2009 yılında girdiği Moğolistan'ın Bilim ve Teknoloji Üniversitesinin Bilgisayar Bilim ve Yönetimi Okulu Mühendislik Fakültesi Bilgisayar donanımı ve ağ mühendisliği Bölümü'nden 2013 yılında Bilgisayar donanımı ve ağ mühendisi olarak mezun oldu. 2013-2017 yıllar arasında Kocaeli Üniversitesi Fen Bilimleri Enstitüsü, Elektronik ve haberleşme Mühendisliği Anabilim Dalı'nda Yüksek Lisans Öğrenimine devam etmektedir.