T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

KABLOLU İLETİŞİM AĞLARINDA YENİ BİR ŞİFRELEME TABANLI GÜVENLİK UYGULAMASI

DOKTORA TEZİ

Ahmet KARACA

Enstitü Anabilim Dalı	**:**	**ELK. ELKTR. MÜHENDİSLİĞİ**
Enstitü Bilim Dalı	**:**	**ELEKTRİK MÜHENDİSLİĞİ**
Tez Danışmanı	**:**	**Yrd. Doç. Dr. Halil İbrahim ESKİKURT** **Doç. Dr. Özdemir ÇETİN**

Temmuz 2012

T.C.
SAKARYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

KABLOLU İLETİŞİM AĞLARINDA YENİ BİR ŞİFRELEME TABANLI GÜVENLİK UYGULAMASI

DOKTORA TEZİ

Ahmet KARACA

Enstitü Anabilim Dalı : ELK. ELKTR. MÜHENDİSLİĞİ

Enstitü Bilim Dalı : ELEKTRİK MÜHENDİSLİĞİ

Bu tez 30 / 07 / 2012 tarihinde aşağıdaki jüri tarafından Oybirliği ile kabul edilmiştir.

Prof. Dr. Hüseyin EKİZ	**Y. Doç. Dr. Halil İbrahim ESKİKURT**	**Doç. Dr. Ahmet Turan ÖZCERİT**
Jüri Başkanı	**Tez Danışmanı**	**Üye**
Y. Doç. Dr. Feyzi Akar	**Y. Doç. Dr. Gürsel DÜZENLİ**	
Üye	**Üye**	

TEŞEKKÜR

Tez çalışmam süresince çalışmalarımı teşvik eden, her türlü yardımlarını esirgemeyen danışmanım Yrd. Doç. Dr. Halil İbrahim ESKİKURT ve eş danışmanın Doç. Dr. Özdemir Çetin'e minnet borçluyum. Tez konusunun belirlenmesinde verdiği fikirler nedeniyle Yrd. Doç. Dr. Feyzi AKAR'a, MATLAB ortamında yazılım geliştirme aşamalarında desteklerini esirgemeyen değerli mesai arkadaşlarım Arş. Gör. Barış BORU ve Arş. Gör. Sezgin KAÇAR'a teşekkürü bir borç bilirim. Ayrıca bu günlere gelmemi sağlayan anne ve babama, başından sonuna kadar çalışmalarımı sabırla destekleyen eşime sonsuz teşekkür ederim.

Ahmet KARACA

İÇİNDEKİLER

SİMGELER VE KISALTMALAR

ADPCM : Adaptive Differantial Pulse Code Modulation
CELP : Code Excited Lineer Prediction
SEA : Scalable Encryption Algorithm
XTEA : Extended Tiny Encryption Algorithm
ADC : Analog Digital Convertor
DAC : Digital Analog Convertor
PSTN : Public Switchwed Telephone Network
POTS : Plain Old Telephone System
ITU : International Telecommunication Union
SNR : Singnal Noise Ratio
TEA : Tiny Encryption Algorithm
AES : Advenced Encryption Standard
DES : Data Encryption Standart
VoIP : Voice Over IP
GSM : Global System for Mobile Communications
PCM : Pulse Code Modulation

ŞEKİLLER LİSTESİ

TABLOLAR LİSTESİ

ÖZET

Anahtar kelimeler: Güvenli konuşma, güvenli iletişim, ADPCM, PSTN, SEA, XTEA, stenografi

Günümüzde haberleşme teknolojileri hızla gelişmekte ve güvenli iletişim önemli bir kavram olarak ortaya çıkmaktadır. Gerek askeri gerek sivil uygulamalar da yapılan bir görüşmenin, düşmanın ya da ticari bir rakibin eline geçmemesi için kullanılan ve geliştirilen birçok güvenli haberleşme sistemi mevcuttur. Bu sistemler üzerindeki çalışmalar günümüzde hala güncelliğini korumaktadır. İnternet ortamında, uydu haberleşmesinde, telsiz haberleşmesinde, kablolu telefon hatları ve mobil telefonlarda bu teknolojiler kullanılmakta ve yeni teknikler geliştirilmektedir.

Bu çalışmada Public Switched Telephone Network (PSTN) üzerinden güvenli haberleşme için bir sistem tasarlanmış ve gerçekleştirilmiştir. Konuşma sinyali dijital sinyale çevrilip şifrelendikten sonra bir dial up modem aracılığıyla PSTN üzerinden alıcıya gönderilmektedir. Burada şifreleme için düşük bellek ve işlem kapasitesine sahip sistemler için geliştirilmiş SEA ve XTEA algoritmaları kullanılmıştır. Gerçekleştirilen sistem ile konuşma sinyali şifrelenmeden önce bu sinyalin üzerine bir metin gizlenebilmektedir ve bu sayede sistemin güvenliği arttırılmıştır. Metin dosyası ses sinyali üzerine gömüldüğünde ses sinyalindeki bozulmaların kulakla algılanmayacak seviyede olmasına dikkat edilmiştir. Hattı dinleyen yetkisiz kişiler şifrelemeyi çözüp ses sinyalini elde etseler bile ses içindeki gizlenmiş veriyi algılayıp steganaliz yöntemleri ile elde etmeleri gerekmektedir. Ayrıca yapılan ikinci bir uygulamada, kısa aralıklarla şifreleme algoritmaları ile bu algoritmaların kullandıkları anahtarları değiştirilmiş ve bu değişimlerle ilgili bilgiler alıcı ile eş zamanlı çalışmanın sağlanması için gizli metin içinde gönderilmiştir. Böylece şifrenin kırılıp yapılan gizli haberleşmenin içeriğinin elde edilmesi daha da zorlaştırılmış ve sistem güvenliği arttırılmıştır.

A NEW ENCRYPTION BASED SECURITY LEVEL IMPLEMANTATION FOR PULIC SWITCHED TELEPHONE NETWORK (PSTN)

SUMMARY

Keywords: Secure speech, secure communication, ADPCM, PSTN, SEA, XTEA, steganography

Today, communication technologies are developing rapidly and secure communication appears to be an important concept. There are many secure communication systems developed and used for protecting any conversation from an enemy or a trade rival in both military and civil applications. The works performed for these systems still remain up to date today. In the internet, the satellite communication, the radio communication, the wired telephone lines and the mobile phones, these technologies are being used and new techniques are being developed.

In this work, a system for secure communication over Public Switched Telephone Network (PSTN) has been designed and realized. After the speech signal is converted to digital signal, the digital signal is encrypted and it is sent to receiver over PSTN by the dial-up modem. The SEA and XTEA algorithms which have been developed for low memory and process capacity, have been used for the encryption. With the developed system, before the encryption of the speech signal, a text can be hide into the signal. So the security of the system has been increased. When the text is embedded to the speech signal, it has been considered that the distortions in the signal are in unsensiblelevel fort he ears. Even if unauthorized persons listening the line can decrypt the encryption and obtain the speech signal, they have to detect and obtain the hidden data in the signal by using the steganalysis methods. In addition, the encryption algorithms and the keys of them have been changed in short periods and the information related with the changes has been sent in the hidden text for synchronization with the receiver. Thus, it is become harder to decrypt the encryption and to obtain the context of the hidden communication and the security of the system is increased.

BÖLÜM 1. GİRİŞ

Bilişim teknolojilerinin hızla geliştiği günümüzde bireyler veya kurumlar arasında güvenli haberleşme araştırmacıların üzerinde durduğu önemli bir konu haline gelmiştir. Günümüzde ses, veri, görüntü şifreleme ve veri gömme ile bilgi güvenliği üzerine birçok çalışma yapılmıştır [1–26]. Geliştirilen bütün bu tekniklere bakıldığında en önemli amacın haberleşme mahremiyetinin sağlanması olduğu görülmektedir. Örneğin, günümüzde hayatın önemli bir parçası haline gelmiş olan sesli haberleşme imkânı sayesinde insanlar özel görüşmelerinden bankacılık işlemlerine kadar birçok önemli işlerini bu iletişim kaynaklarını kullanarak gerçekleştirmektedirler. Bu nedenle iletişim ortamlarının güvenirliliğinin sürekli sağlanması önemli bir konudur. GSM gibi kablosuz hatlar üzerinden gerçekleştirilen sesli haberleşme için geliştirilen güvenlik teknikleri kablolu hatlar için geliştirilen tekniklerden farklılık gösterir. Kablosuz hatlar üzerinden gerçekleşen haberleşme kablolu hatlara göre saldırılara daha açıktır. Kablosuz iletişim hattının herhangi bir fiziksel bağlantıya sahip olmamasından dolayı kötü niyetli kişiler tarafından fark edilmeden dinlenmesi daha kolaydır. Kablosuz ortam güvenliğinin kablolu ortama göre daha kırılgan olmasından dolayı üst düzey güvenlik gerektiren iletişimlerde kablolu ortam kullanılması daha uygun olacaktır.

Bu alan da yapılan çalışmaları iletişim ortamı açısından düşündüğümüzde dört başlık altında toplayabiliriz. Bunlar kablolu telefon şebekesini (PSTN) kullanan uygulamalar, GSM şebekesini kullanan mobil telefon uygulamaları [27–37], internet bağlantısı üzerinden gerçekleştirilen (VoIP) Ethernet bağlantılı uygulamalardır [38–61].

GSM ağlar ve internet bağlantısı üzerinden yapılan çalışmalara günümüzde daha çok rastlanmaktadır. Bu uygulamalar ek donanım gerekmeden bilgisayara ya da cep telefonuna yüklenecek bir yazılım üzerinden kolaylıkla yapılabilecektir. Özellikle GSM uygulamaları cep telefonlarının taşınabilirliği sayesinde her istenilen yerden kullanılması bu uygulamalarda büyük kolaylık sağlayacaktır. Ama bu iki sistemde de kayıtlı bir hattın olması zorunludur ve aynı zamanda kullandığımız telefon yada bilgisayarda yüklü bir güvenlik programı olmalıdır. Başka bir telefon kullanmak istenildiği takdirde bu yazılımın o telefona da yüklenmesi gerekecektir. Yada bilgisayarı bir internet hattına bağlayıp konuşmak istenildiği takdirde bu bilgisayarın internete erişimi için ağ yöneticisine başvurulması gerekmektedir. Bu durumda da kimin kullanılan hattan gizli bir görüşme yapıldığının tespiti ve bu görüşmenin kimin tarafından yapıldığının tespiti kolay olacaktır. Kablolu telefon hatlarında ise, günümüzde azalmış olmasına rağmen hala kullanılmakta olan bir telefon kulübesindeki telefon bile rahatlıkla kullanılabilir ve o saatte o görüşmeyi kimin yaptığının bir tespiti yapılamaz. Kablolu telefon hatları için geliştirilmiş bu sistemi telefon hattına kablo ile bağlamadan kullanmamız da mümkündür. Modem tarafından üretilen sinyaller bir hoparlör ile telefon mikrofonuna aktarılarak ta iletişim sağlanabilir. Bu durumda da yanımızda taşıdığımız cihaz sayesinde herhangi bir telefondan aynı cihaza sahip olan birisiyle rahatlıkla görüşebiliriz.

Geliştirilen yöntemler aynı zamanda uygun bir yazılım kullanılarak rahatlıkla cep telefonlarına da aktarılabilecektir. Ya da kullanılan dial up modem modülü yerine kullanılacak bir ethernet bağlantı modülü ile kolay bir şekilde VoIP uygulamasına dönüştürülebilecektir. Ama iletişim ortamı olarak kablolu telefon hatları (PSTN) seçildiğinden bu hattın kısıtlamaları göz önünde bulundurularak yeni bir sistemin tasarımı gerçekleştirilecektir. İkinci kısıtlama ise, kullanılacak mikrodenetleyicinin seçimi ve mikrodenetleyicinin işlem gücüne uygun ses kodlama ve şifreleme algoritmalarının seçimi olacaktır.

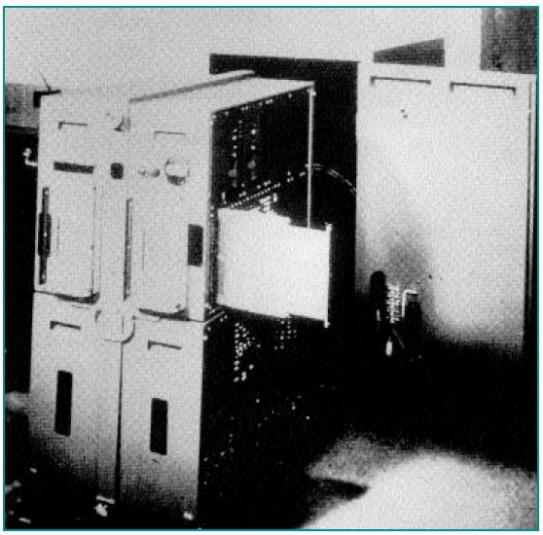
1.1. Güvenli Konuşmanın Tarihçesi

Ses kodlamanın tarihi Tom Tremain adlı bilim adamının kariyeri ile yakın bir şekilde bağlantılıdır. 1959 yılında National Security Ajansına havacı teğmen olarak katılan Tremain ses kodlamanın geleceğini de şekillendiren bu atama ile, A.B.D. hükümetinin kidemli ses bilimcisi olmuştur. Ses biliminde bir lider ve uzman olarak tanınan Tremain A.B.D. ve NATO nun taktik ve stratejik güvenli iletişim programları için kritik çalışmalar yapmıştır.

Ulusal Güvenlik Ajansı (National Security Agency, NSA) şifrelenmiş ses uygulamaları için ses kodlamanın sorumluluklarını, geleneklerini ve uzmanlığını 1952'de Ordu Güvenlik Ajansı'ndan (Army Security Agency, ASA) miras olarak almıştır. D-Day istilasının planlanması için Roosevelt ve Churchill tarafından kullanılan ünlü SIGSALY ses kodlayıcısının geliştirilmesine katılmıştır. Şekil 1.1'de gösterilen SIGSALY ses kodlama tabanlı bir sistemdir. Bant geçiren filtreler kullanılarak yapılmış bu sistem analog olarak çalışmaktaydı [62,63].

Şekil 1.1. Sigsaly güvenli konuşma sistemi

Tremain, SIGSALY'den NSA'ya gelene kadar Bell Labs ile birlikte ses kodlayıcılarının birçok versiyonu geliştirilmiştir. KO-6 ses kodlayıcı 1949'da geliştirildi ve 1200 bps SIGSALY ses kodlayıcıya yakın sınırlı kaliteye sahipti. Bunu 1953'te Şekil 1.2 de gösterilen 1650 bps KY-9 takip etti. 12 kanal bir ses kodlayıcı ve el yapımı transistörler kullanan KY-9 yarı iletken teknolojinin ilk uygulamalarından biridir. Bu SIGSALY'nin vakum tüp teknolojisinin ağırlığını 55 tondan sadece 565 pound da indirdi. 1961'de, Tremain'in ilk projesi U.S. kanal ses kodlayıcı teknolojisinin son nesli olan HY-2 ses kodlayıcısının geliştirilmesiydi. Şekil 1.3'te gösterilen HY-2, ağırlığı 100 pound da azaltmak için "Flyball" renkli-kodlama modüler lojik devrelerini kullanan 16 kanal 2400bps bir sistemdi. 1964-1962 arasında, bir dijital bilgisayarda kanal formant ses kodlayıcının ilk simülasyonu oluşturuldu ve 1966-1968 arasında ilk dijital kanal kodlayıcının geliştirilmesine yardım etti. Bu zamanda Amerikan ses kodlayıcı teknolojisinin en iyisi bile analog teknoloji kullanılması nedeniyle sınırlı bir kaliteye sahipti. Analog filtre ve yükselteçlerin çalışması zamanla ve sıcaklıkla değiştiğinden ses analizörü ve ses sentezleyici arasında hassas ayarlamalar yapılırdı. Sahadaki performans asla laboratuvar performansına ulaşamazdı. Kullanıcıların sentetik bir "Donald Duck" kalitesine sahip sistemlerini kullanmayı istememeleri ve düşük kalitesinden dolayı HY-2'de ilk başlarda tercih edilmedi. HY-2, 1961 yılında geliştirilmiş ve Wietnam Savaşında kullanılmıştı. Sonuç olarak bu cihazların yayılması sınırlıydı. Tremain'in ses kodlamaya en önemli katkısı, ses için dijital sinyal işlemenin mümkün olduğunu görmesiydi. Analog ayar devrelerinin sanat durumunda olduğu bir devirde sesin bilgisayar tabanlı işlenmesi bir hayaldi. Bu ses işlemede çok önemli bir dönüm noktasıydı. Tremain ses kodlayıcıların Linear Predictive Coding (LPC) versiyonunu geliştirmek için tekrar Bell Labs ile birlikte çalışarak bu yeni yaklaşıma öncülük etmiştir.

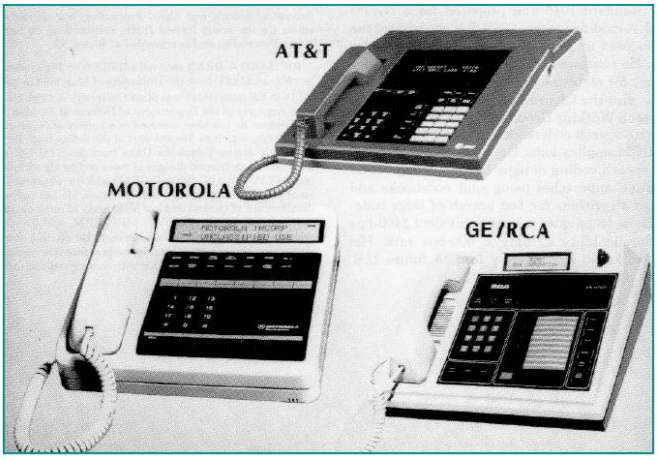


Şekil 1.2. KY-9 transistörlü güvenli konuşma sistemi

Şekil 1.3. HY-2 ses kodlayıcı

O dönemde ses kodlamada bilgisayarların daha hızlı çalışması için, günümüzde de yaygın olarak kullanılan birçok teknik ve yeni yapılar geliştirilmiştir. Örnek olarak yoğun çarpma işlemlerinin yapıldığı oto korelasyon fonksiyonu yerine Average Magnitude Difference Function (AMDF) fonksiyonu verilebilir. 1974'te CSP-30 bilgisayarı üzerinde çalışan LPC-10 nun ilk gerçek zamanlı simülasyonu sunulmuştur. Bu sinyal işlemede bir dönüm noktasıydı ve NSA'nın ses kodlama ürünlerinin tamamen yeni bir ailesi olan STU ürünlerinin geliştirilmesine neden oldu.

STU ürünleri ilk nesil AMD2901/TRW sinyal işlemcisi ile yapıldı ve ses kodlamanın geleceğinin değişmesini sağladı [62,63].

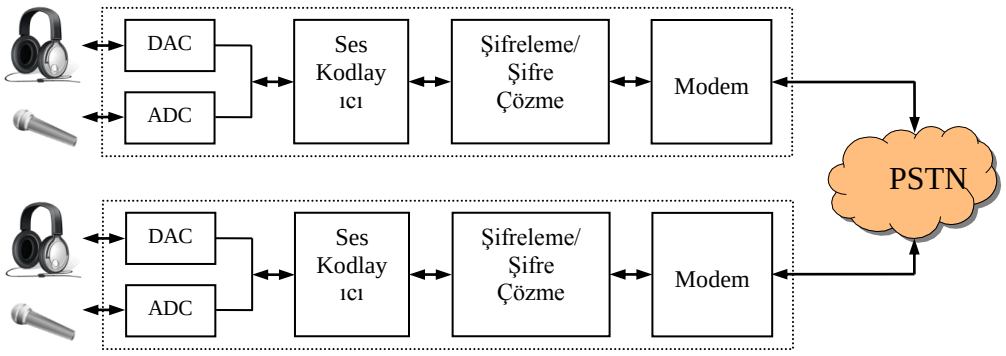


Şekil 1.4. STU-III güvenli konuşma cihazı ailesi

Şekil 1.4'te gösterilen STU-III LPC-10 nun geliştirilmiş bir versiyonunu kullanan üçüncü nesil masa telefonudur ve güvenli konuşma kullanıcıları tarafından desteklenmektedir.

Ses kodlayıcılar, uzun süre egzotik şifreleme şemaları ile ilgilenmiştir ve günümüzde kablosuz iletişim, sesli posta ve sentetik ses uygulamaları için birçok uygulama alanı bulmaktadır. Günümüzde uydu haberleşmesinde, cep telefonlarında kullanılan ses kodlayıcıları bahsedilen ilk çalışmaların devamı niteliğindedir. Değişken eğimli delta modülasyonu (continuously variable slope delta modulation, CVSD), doğrusal öngörü kodlayıcı (linear predictive coders, LPC), konuşma kodlayıcılar (vocoders), uyarlanabilir kestirimci kodlama (adaptive predictive coders), kod uyartımlı doğrusal öngörü kodlama (code excited linear predictors CELP), modem teknolojileri üzerine orijinal çalışmaları günümüzde kullanılan çoğu ses iletişiminin temelidir. İlerleyen zamanlarda Code Excited Linear Prediction (CELP) ses kodlayıcısı, STU-III te kullanıldı ve Ulusal Standard 1016 olarak kabul edildi ve bir NATO standardı olarak önerildi. CELP, modifiye edilmiş ve bir formu Kuzey Amerika cep telefonu şebekesinde kullanılan algoritmanın temelidir. Bunlar ses kodlama konusunda birçok önemli çalışmalardan bir kısmıdır.

1.2. PSTN Üzerinden Güvenli Konuşma Uygulamaları



Şekil 1.5. PSTN üzerinden güvenli konuşma sistemi blok diyagramı

PSTN üzerinden güvenli konuşma çalışmalarını Şekil 1.5'te verilen blok diyagramı kullanılarak inceleyebiliriz. Bu çalışmalar Şekil 1.5'te gösterildiği gibi sesin dijitale çevrilmesini sağlayan ADC ve DAC bloğu, ses kodlama bloğu, şifreleme bloğu ve modem bloğu olarak dört bloktan oluşmaktadır. Uygulamanın kullandığı iletişim ortamına göre modem bloğu yerine ethernet ve kablosuz iletişim modülleri gibi farklı bloklar kullanılması söz konusudur.

Bu konuda yapılmış çalışmalardan birisi Nuzli Mohamad Anas tarafından yapılmıştır [64]. Bu çalışmada ses kodlayıcı olarak ICELP (improved code-excited linear prediction) olarak isimlendirdikleri 4,8Kbps bit akış hızına sahip olan bir algoritma kullanılmıştır. Kullanılan bu yeni algoritma ses kodlamada iyi bilinen CELP (code-excited linear prediction) algoritmasının iyileştirilmiş halidir. CELP kodlama algoritmasının kalitesini koruyup işlem gücünü azaltan bu yeni algoritma ICELP olarak isimlendirilmiş ve bu çalışmada kullanılmıştır. Şifreleme bloğu olarak da simetrik şifreleme algoritması olan DES (Data Encryption Standart) algoritması kullanılmıştır. Bu işlemlerin gerçekleştirilme ortamı olarak da Texas Instruments (TI) firmasının üretmiş olduğu TMS320C54CST isimli bir DSP(digital signal processing) işlemcisi kullanılmıştır. Telefon uygulamaları için geliştirilmiş bu işlemci 120MIPS işlem gücüne sahiptir ve bir DSP işlemcisi olması nedeniyle CELP gibi sinyal işleme teknikleri gerektiren bir algoritmayı çalıştırmak için gerekli özelliklere sahiptir [65]. Ayrıca bu işlemci bir dial up modem de içermektedir.

Kullanılan şifreleme algoritması DES in günümüzde geçerliğini yitirmiş, kırılabilen bir algoritma olması ise en büyük dezavantajlardan biridir.

Javier Calpe tarafından yapılan ve secraphone ismi verilen bir uygulamada ses kodlayıcı olarak CELP kodlayıcı kullanılmıştır [66]. Celp kodlayıcı 7200 bps ve 9600 bps hızlarında çalışmaktadır. İletişim ortamına göre veri aktarım hızı seçilmektedir. Şifreleme bloğu olarak ta RSA algoritması kullanılmıştır. Bu uygulamada gerçekleştirme ortamı olarak yine DSP işlemci kullanılmıştır.

Luis Diez-del- Rio, tarafından yapılan bir diğer çalışmada ise sadece konuşma güvenliği değil aynı zamanda fax cihazından gönderilen verilerinde güvenliğinin sağlanması amaçlanmış ve bu sisteme Tiche ismi verilmiştir [67]. Bu tasarlanan sistemin diğer uygulamalardan farkı, konuşma yanında faks cihazını da işin içine katmış olmasıdır. Bu çalışmada kullanılan ses kodlayıcı 4800 ve 9600 bps bit akış hızlarında çalışan CELP ses kodlayıcı kullanılmıştır. Şifreleme bloğu olarakta bu çalışmada RSA kullanılmıştır. Bu algoritmaların yürütülmesi için Lucent Technologies firmasının ürettiği 20 MIPS işlem gücüne sahip DSP32 DSP işlemcisi kullanılmıştır [68].

Bu konuda Wu Zhi-Jun tarafından Speech Information Hiding Telephone (SIHT) ismi verilen bir çalışmada, güvenli konuşmanın yanında konuşma sinyalinin üzerine stenografi teknikleri kullanılarak gizli bir ses dosyası eklenmektedir [69]. Konuşma sinyalinin kodlanması için 32 Kbps veri akış hızına sahip ADPCM algoritması kullanılmıştır. Bu konuşma sinyali üzerinden gönderilecek gizli ses bilgisi ise 2,4 Kbps veri aktarım hızına sahip bir CELP kodlayıcı tarafından kodlanarak stenografi teknikleri ile konuşma sinyalinin üzerinden gönderilecektir. Burada gizli olarak gönderilecek olan ses sinyali kayıtlı bir ortamdan alınıp kodlanarak, gerçek zamanlı konuşma hızı olmadan gönderilecektir. Bu çalışmada, ses kodlama ve ses çözme algoritmaları ayrı DSP işlemcileri tarafından gerçekleştirilmiştir. Bu işlemleri gerçekleştirmek için Texas Instrument firmasının ürettiği bir adet TMS320C54X ve üç adet TMS320C31 DSP işlemcisi kullanılmıştır.

Bu çalışmada da, bu makaleler referans alınarak PSTN üzerinden güvenli bir konuşma sağlama amaçlanmıştır. Yapılan çalışmada ses kodlama ve şifreleme blokları bulunmaktadır. Bu bloklara ilave olarak stenografi bloğu kullanılmıştır. Stenografi ile, veri göndermek yerine sistemin güvenirliğinin arttırılması için gönderilen gizli bilgi şifreleme algoritmasının ve kullanılan anahtarın belirli aralıklarla değiştirilmesi işlemi gerçekleştirilmiştir.

Geliştirilen yöntemler aynı zamanda uygun bir yazılım kullanılarak rahatlıkla cep telefonlarına da aktarılabilecektir. Ayrıca kullanılan dial up modem modülü yerine kullanılacak bir ethernet bağlantı modülü ile kolay bir şekilde VoIP uygulamasına dönüştürülebilecektir. İletişim ortamı olarak kablolu telefon hatları (PSTN) seçildiğinden bu hattın kısıtlamaları göz önünde bulundurularak sistem tasarlanmış ve yine kullanılacak mikrodenetleyicinin işlem gücüne uygun ses kodlama ve şifreleme algoritmalarının seçimi yapılmıştır.

Güvenli konuşma için kablolu telefon hatlarının seçilmiş olması beraberinde bazı sınırlamaları da getirmektedir. Telefon santrallerinde ses iletişimi için ayrılan bant genişliği 300Hz ve 3500Hz arasındadır. Bu hat üzerinden dijital veri haberleşmesi yapmak için dial up modemler kullanılır. Bu sınırlı bant genişliğinde yapılacak dijital veri iletişimi de sınırlıdır. Bu nedenle, dijitale çevrilen ses bilgisi sıkıştırılarak daha az sayıda bit göndererek bu kısıtlama aşılır. Örnekleme sayısının düşürülmesine neden olan sıkıştırma işlemi ses kalitesini düşürerek gizli şüphe uyandırmayan haberleşme yapma imkanını da azaltmaktadır. Bu çalışmada, ses bilgisinin şifrelenerek karşı tarafa gönderilmesinin yanında, stenografi teknikleri kullanılarak gizlenmiş bir metnin de iletim hattı üzerinden gönderilmesi, bu gizli metin sayesinde şifreleme algoritmasının ve kullanılan anahtarın eş zamanlı değiştirilmesi sağlanmıştır. Bu yöntem sayesinde konuşmanın güvenliği arttırılmıştır.

Tez organizasyonu aşağıda özetlenen 5 bölümden oluşmaktadır:

Bölüm 1'de, güvenli konuşma ile tarihte yapılmış olan çalışmalar anlatılmış ve bu konuda önemli çalışmaları olan Tom Tremain'in güvenli konuşma alanına yaptığı katkılar anlatılmıştır. Ayrıca PSTN üzerinden güvenli konuşma ile ilgili yapılan çalışmalar incelenmiş ve bunların tez çalışmasından farklılıkları ortaya konmuştur.

Bölüm 2'de, çalışmada gerçeklenen sistemin blok diyagramı verilmiş ve her bir blok diyagramla ilgili genel bilgiler verilmiştir. Kullanılan haberleşme ortamı olan PSTN, ses kodlama algoritmaları, veri gömme ve şifreleme işlemleri hakkında genel bilgiler bu bölümde verilmiştir.

Bölüm 3'de sistemin gerçeklenmesi sırasında kullanılan donanımlar ve sistem tanıtılmıştır. Ayrıca testler sırasında kullanılan arayüzün tanıtımı da bu bölümde yapılmıştır.

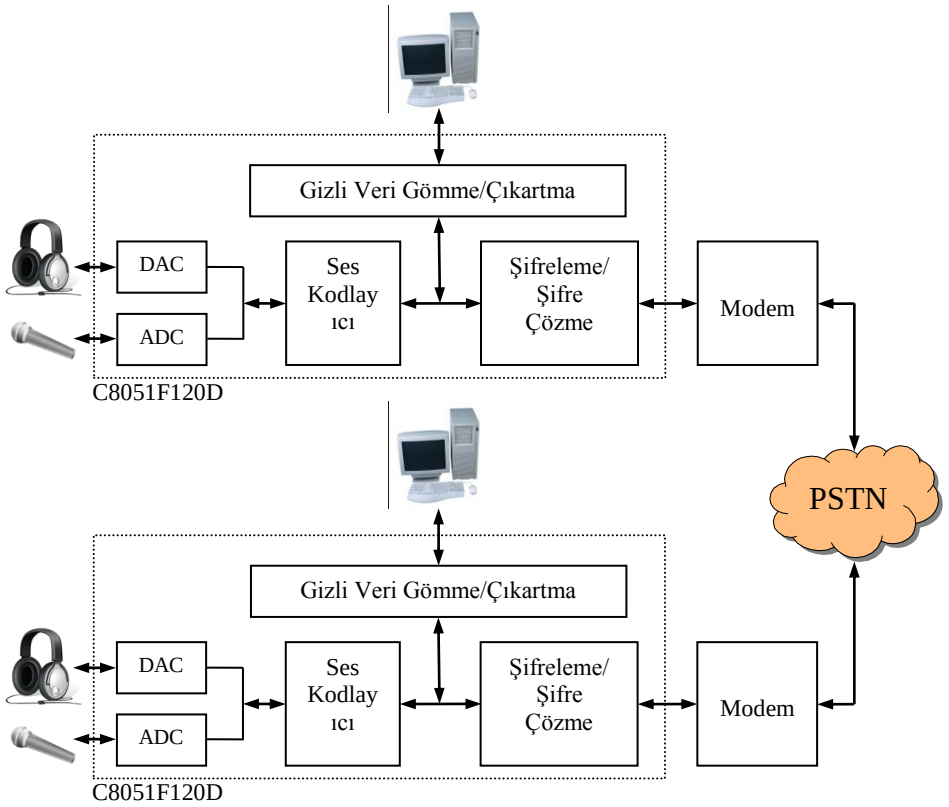
Bölüm 4'de ADPCM algoritmasının ve ADPCM veri gömme algoritmasının başarım analizini içeren bu bölüm de orijinal konuşma sinyali ile kodlanmış konuşma sinyali karşılaştırılarak SNR oranları verilmiştir. Yine veri gömmenin etkilerini incelemek amacıyla veri gömülmüş ve gömülmemiş konuşma sinyali karşılaştırılarak SNR oranı ile veri gömme işleminin meydana getirdiği bozulmalar incelenmiştir. Yapılan uygulamalar da bu böülümde tanıtılmıştır.

Bölüm 5'te ise, yapılan deneysel çalışmalardan elde edilen sonuçlar değerlendirilerek çalışmanın katkıları tartışılmıştır. Ayrıca gelecekte yapılması düşünülen, tez çalışmasının devamı niteliğini taşıyabilecek yeni çalışmalar da önerilmiştir.

Tez çalışmasında kullanılan standart algoritmaların kodları ekler bölümünde verilmiştir. ADPCM kodlayıcı ve kod çözücü ile SEA ve XTEA şifreleme algoritmaları ana program içerisinden bir fonksiyon olarak çağrılmıştır.

BÖLÜM 2. SİSTEMİN TANITILMASI

2.1. Genel Blok Diyagram



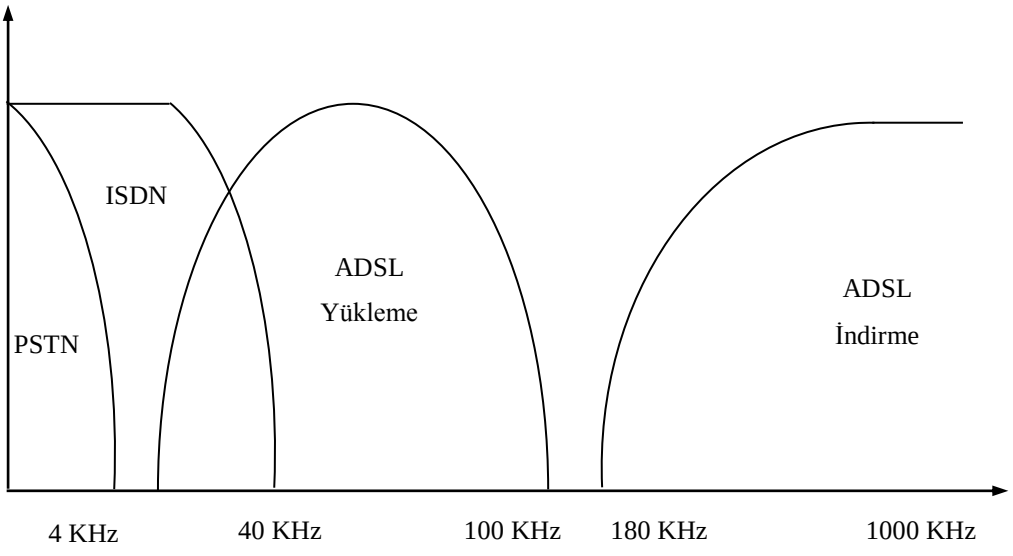
Şekil 2.1. Genel blok diyagram

Oluşturulan sistem PSTN üzerinden güvenli haberleşme sağlamak amacıyla yapılmıştır. Şekil 2.1'deki blok diyagram incelendiğinde, birinci bölümdeki incelenen çalışmalarla arasındaki fark gönderilen ses bilgisinin üzerine stenografi teknikleri ile gizli bir metin gömen ayrı bir bloğun daha olmasıdır. Bu bölümde sistemi oluşturan bloklar ayrı ayrı tanıtılmıştır.

2.2. Kamu Anahtarlamalı Telefon Ağı (PSTN)

Haberleşmek için kullandığımız telefonların bağlı olduğu ağ, kamu anahtarlamalı telefon ağı (PSTN – public switched telephone network) ya da basit eski telefon sistemi (POTS – plain old telephone system) olarak adlandırılır. PSTN aboneleri bir çift bükülü kablo ile santraller üzerinden bu ağa bağlanır. PSTN' de kullanılan ağ teknolojisi devre anahtarlamalı ağ olarak bilinir. Bu ağ teknolojisinde bizim telefon numarası dediğimiz numaralar adres olarak kullanılır. Bağlantı bir istasyonun diğerini aramasıyla iki sabit nokta arasında kurulur. Bağlantı kurulduktan sonra hat, bağlantı sonlandırılana kadar sadece bu iki istasyona aittir ve diğer kullanıcılar tarafından kullanılamaz.

Şekil 2.2'de gösterildiği gibi PSTN konuşma bant genişliği 300Hz-3500Hz'dir ve bu sınırlı bant genişliği kullanılarak ses kalitesinden ziyade konuşmanın anlaşılması söz konusudur. Daha üst frekanslar ise veri aktarımı (internet) amacıyla ISDN ve ADSL gibi bağlantılar için kullanılmaktadır.



Şekil 2.2. PSTN bant aralıkları

2.2.1. PSTN üzerinden veri aktarımı ve dial-up modemler

Telefon şebekesinde analog ses haberleşmesi için kullanılan 4KHz'e kadar olan analog ses bandını veri haberleşmesi için de kullanmak mümkündür. Bunun en büyük örneği geçmişte internet bağlantısı için evlerde kullandığımız dial-up modemlerdir. Bu ve benzeri cihazlar telefon hattını kullanarak uzak mesafedeki bilgisayar yada mikro işlemcili sistemlerin birbiriyle haberleşmesine imkan sunmaktadır.

Telefon hattına bir cihazın bağlanması bazı kurallar çerçevesinde yapılmaktadır ve bağlanılacak cihazın bu işle görevli bir kurum tarafından onaylanması gerekmektedir. Onaylanmamış bir cihazı hatta bağlamak bir çok ülkede yasal değildir. Bunun temel nedeni güvenliktir. Bağlayacağımız cihaz telefon hattına zarar verebilir. Telefon hattına onaylanmamış bir cihaz bağlamak hatta oluşan sinyalleri zayıflatabilir ve hattın çalışmasını bozabilir. Hatta, en kötü durumda santraldeki sistemde çalışan mühendislerin elektriğe çarpılmasına neden olabilir. Bu nedenle üreticiler tarafından PSTN hattına bağlanmak için güvenlik ve elektriksel işlevsellik bakımından kesin standartları sağlayan entegreler ve devreler üretilir. Direk erişim modülleri olarak isimlendirilen DAA'lar bunlardandır. Bu entegreler yada modüller, içerdikleri yalıtım devreleri sayesinde telefon ağı ile bu ağa bağlamak istediğimiz sistemler arasında bir ara birim (interface, line driver) olarak kullanılırlar. Bu modülleri kullanmak, tasarlanan donanımın onay gereksinimi olmadığını farz etmek anlamına gelmez. Tasarım hala onay gerektirir ve PCB yerleşiminde ve birleştiricilerde dikkatli olunmalıdır. Gömülü bir sistemde böyle bir modül kullanmakla kolayca bilgisayar ağına bağlanılabilir. Bu bir mikro işlemcinin arama yapmasına ve aramalara cevap vermesine olanak tanır ve DTMF tonları kullanarak (yada sentezlenmiş konuşma) uzak sistemlerle iletişim kurmasını sağlar.

PSTN hattı üzerinden veri iletişiminde bir sonraki basamak modemlerdir. Bunlar DAA erişiminin bütün olanaklarını içerirler fakat aynı zamanda daha yüksek hızlarda veri şifreleme ve iletimi olanaklarına da sahiptirler. Modemler dijital bir veri katarını PSTN üzerinde taşınabilen tonlara ve sinyal darbelerine çeviren aygıtlardır.

Modemler dışarıyı arama ve gelen aramalara cevap verme gibi yeteneklere de sahiptirler. Direkt olarak telefon hattına bağlanırlar ve telefonun yerini alırlar. İlk modemler ‘0’ ve ‘1’ dijital durumlarını ifade etmek için 300-3400 Hz frekans aralığında iki ayrı ton yada frekans kullanırdı. Bu modemler modern standartlara göre oldukça yavaştı fakat tipik bir oturumda karşılaşılabilecek gürültülere, voltaj ve faz değişimlerine karşı oldukça esnekti.

Tablo 2.1. Günümüzdeki kullanılabilir olan modem hızları

Veri Hızı	Standart İsmi
56 kbps	ITU-T V.90
54.666 kbps	ITU-T V.90
53.333 kbps	ITU-T V.90
52 kbps	ITU-T V.90
50.666 kbps	ITU-T V.90
49.333 kbps	ITU-T V.90
48 kbps	ITU-T V.90
46.666 kbps	ITU-T V.90
45.333kbps	ITU-T V.90
44 kbps	ITU-T V.90
42.666 kbps	ITU-T V.90
41.333 kbps	ITU-T V.90
40 kbps	ITU-T V.90
38.666 kbps	ITU-T V.90
37.333 kbps	ITU-T V.90
36 kbps	ITU-T V.90
34.666 kbps	ITU-T V.90
33.333 kbps	ITU-T V.90
32 kbps	ITU-T V.90
30.666 kbps	ITU-T V.90
29.333 kbps	ITU-T V.90
28 kbps	ITU-T V.90
33.6 kbps	ITU-T V.34
31.2 kbps	ITU-T V.34
28.8 kbps	ITU-T V.34
26.4 kbps	ITU-T V.34
24.0 kbps	ITU-T V.34
21.6 kbps	ITU-T V.34
19.2 kbps	ITU-T V.34
16.8 kbps	ITU-T V.34
14.4 kbps	ITU-T V.34 or V.32bis
12.0 kbps	ITU-T V.34 or V.32bis
9600 bps	ITU-T V.34, V.32bis, or V.29
7200 bps	ITU-T V.34 or V.32bis
4800 bps	ITU-T V.34 or V.32bis
2400 bps	ITU-T V.34 or V.22bis
1200 bps	ITU-T V.22bis, V.23, or Bell 212A
300 bps	ITU-T V.21
300 bps	Bell 103

Daha güvenilir ve öngörülebilir oldukça çoklu evreli modülasyon metotları kullanılmaya başlandı. Bu yeni çıkan modülasyon metotları veri aktarım hızını arttırdı.

Modem hızlarını tanımlamak için Uluslararası telekomünikasyon birliği (ITU, international telecommunications Union) tarafından tanımlanmış V serisi olarak bilinen standartlar kullanılır. Tablo 2.1'de günümüzde kullanılan dial-up modem hızları gösterilmiştir. PSTN bağlantısının teorideki veri taşıma kapasitesi ile mümkün olan en yüksek hız 56 kbps olmuştur [70–72].

Modemler yüksek hızlara ulaşmak için veri iletimi sırasında veri sıkıştırma yöntemleri uygularlar. En yaygın sıkıştırma metotlarından biri Microcom firmasının geliştirmiş olduğu MNP5 sıkıştırmasıdır. Her iki modemde MNP5 olduğunda veri iletim hızı ortalama olarak ikiye katlanabilir. V42b formatı standart olarak donanımsal veri sıkıştırmayı kullanır.

2.2.2. AT komut seti

1980'lerde Hayes Company tarafından modemi kontrol etmek için kullanılan özel kod dizileri ile belirlenmiş komutlar geliştirilmiştir. Bu kod dizileri AT komutları olarak tanımlanmış ve standart hale getirilmiştir. Bu nedenle modemler bazen Hayes uyumlu olarak tanımlanır. Modemler şu anda oldukça standart hale geldiği için telefon hattına bağlantıdaki fiziksel katman PSTN değil de AT komut setini içeren modem olarakta kabul edilebilir. Modemler bir bilgisayar yada mikroişlemci tarafından genellikle RS232 bağlantısı tarafından AT komut setini kullanarak kontrol edilir. Bu gün GSM telefonlarla da bu komut seti ile haberleşmek mümkündür [70–73].

AT komut seti bir modemin, seri hattan tam olarak kontrol edilmesine olanak verir. Tabloda sık kullanılan AT komutlarına örnekler verilmiştir. Her bir komut AT harfleri ile başladığı için komut setine AT komutları adı verilmiştir. Tablo 2.2'de sık kullanılan AT komutlarından bazıları verimişitir. Görülen bu komutlar ASCII

karakterler olarak RS232 bağlantısı aracılığıyla tek tek modeme gönderilir. Modemde yine ASCII karakterlerden oluşan bir mesaj ile komutun yürütülmesi hakkında bilgi verir. Modem iki modda çalışır. Bu modlardan biri modemin kontrol edilmesi yani AT komutlarının modemin özel işlevler yerine getirmesini sağladığı durumdur. Diğeri ise veri alma ve gönderme yani modeme gönderilen bütün verinin saydam olarak hatta iletildiği durumdur.

Tablo 2.2. Sık kullanılan AT komutlarından bazıları

AT komutu	Açıklama
ATDT123456	Verilen numarayı ara
ATH0	Telefonu kapat
ATH1	Ahizeyi kaldır
ATVn	Sonuçları rakam olarak göster
ATSnm	n yazmacını m değerine getir
ATSn =?	n yazmacının içeriğini sorgula
ATIn	Modem bilgisini rapor et ör: n=7 model ve yapım
ATZ	Modemi resetle

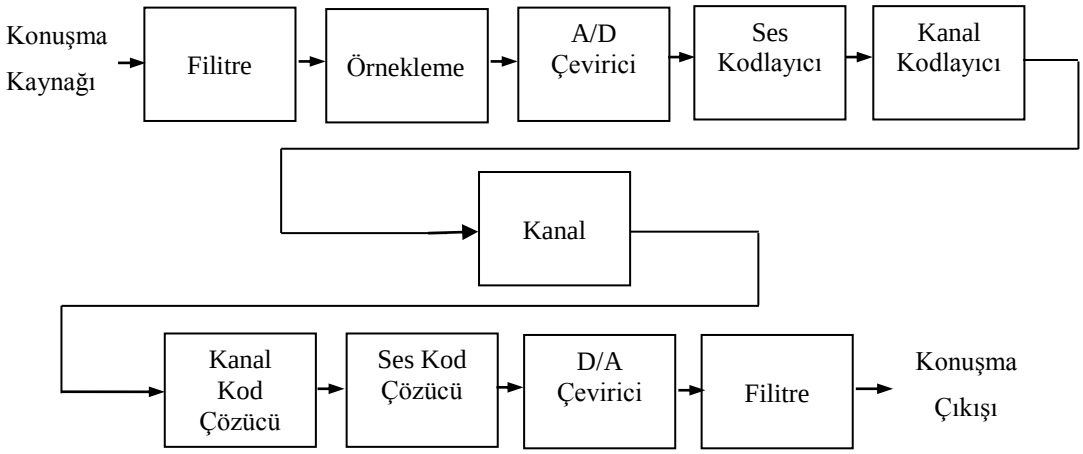
2.3. Ses Kodlama Algoritmaları

Konuşma kodlamanın diğer bir adı da konuşma sıkıştırmadır (speech compression). Konuşma sıkıştırma ismi çok kullanılmasa da yapılan işi daha iyi özetlemektedir. Ses kodlamasındaki asıl amaç veri sıkıştırmadaki gibi konuşma sinyalini ifade edecek bit sayısını azaltmaktır. Tabii burada kayıpsız bir sıkıştırma söz konusu değildir. Konuşma kodlama bir dijital konuşma sinyalini mümkün olduğunca az bit kullanarak ve aynı zamanda konuşma kalitesini makul bir seviyede tutarak temsil etmek için kullanılan bir işlemdir. Konuşmada meydana gelecek bir bozulma mutlaka olacaktır ama bu bozulma konuşmanın anlaşılmasını önlemeyecek boyutta olacaktır. Konuşma kodlama sinyal işlemenin en önemli alanlarından biridir. Bu alanda birçoğu günlük hayatımızda kullanılmakta olan birçok algoritma geliştirilmiştir ve bu algoritmalar ile ilgili birçok çalışma çalışma yapılmıştır [74–94]. Bu algoritmaların günlük hayatta kullanımına örnek olarak olarak GSM ve IP telefonları verebiliriz.

2.3.1. Bir konuşma kodlama sisteminin yapısı

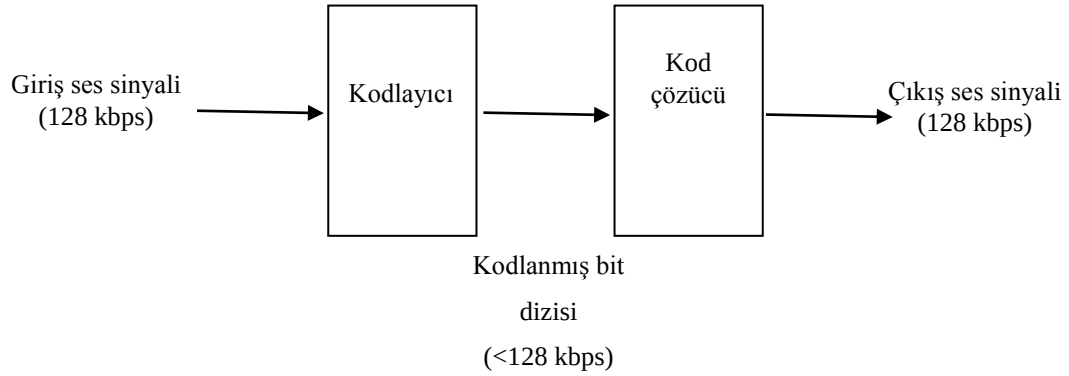
Şekil 2.3 bir konuşma kodlama sisteminin blok diyagramını gösterir. Geleneksel olarak, çoğu konuşma kodlama sistemleri 300 ve 3400 Hz arasında sınırlı frekans içeriği ile, telekomünikasyon uygulamalarını desteklemek için tasarlanmıştır. Bu nedenle kodlamak istediğimiz ses sinyalindeki istenmeyen diğer frekansları ve gürültüleri yok etmek için girişte bir alçak geçiren filtre kullanılır. Sonrasında bir ADC ile bu sinyal sayısal hale çevrilir. ADC çıkışı sayısal ayrık zamanlı bir konuşma sinyalidir. Bu sinyalin kalitesi ve veri miktarı örnekleme frekansına ve ADC nin bit sayısına bağlıdır. Nyquist teoremine göre, örnekleme frekansı birbirine karışmayı önlemek için sürekli zaman sinyali bant genişliğinin en az iki katı olmalıdır. Konuşma sinyalleri için standart örnekleme frekansı değeri genellikle 8 KHz olarak seçilmiştir. Uniform niceleme kullanarak ve toll quality [Jayant and Noll, 1984] yi koruyarak analog örnekleri dijital formata dönüştürmek için 8 bits/sample'dan daha fazlası gereklidir. 16 bits/sample kullanımı yüksek olarak kabul edilen bir kalite sağlar. Bu parametreleri kullanarak bit akış hızını hesaplarsak 8 KHz örnekleme frekansı saniyede 8000 örnek anlamına gelir. Her bir örneğinde 16 bit olduğunu kabul edersek ADC çıkışındaki bit çıkış hızı 128 Kbs olur.

$$\text{Bit-hızı} = 8\text{kHz} \times 16 \text{ bits} = 128 \text{ kbps}$$



Şekil 2.3. Konuşma kodlama sisteminin blok diyagramı

Giriş bit hızı olarak bilinen bu hız, kaynak kodlayıcı tarafından azaltılmaya çalışılır. Şekil 2.4'te gösterildiği gibi kaynak kodlayıcının çıkışı dijital konuşmayı gösterir ve genellikle bu, giriş bit hızından daha düşüktür.



Şekil 2.4. Ses kodlayıcının blok diyagramı

2.3.2. Ses kodlama algoritmalarının sınıflandırılması

Çeşitli yaklaşımlar arasında açık bir ayrım olmaması nedeniyle modern konuşma kodlayıcıların sınıflandırılması basit değildir ve genellikle kafa karıştırıcıdır. Bu bölüm, bazı mevcut sınıflandırma kriterleri sunmaktadır. Konuşma kodlamanın sürekli gelişen bir alan olduğu ve alternatif teknikler sunulduğunda kodlayıcıların yeni sınıflarının oluşacağı göz önünde tutulmalıdır.

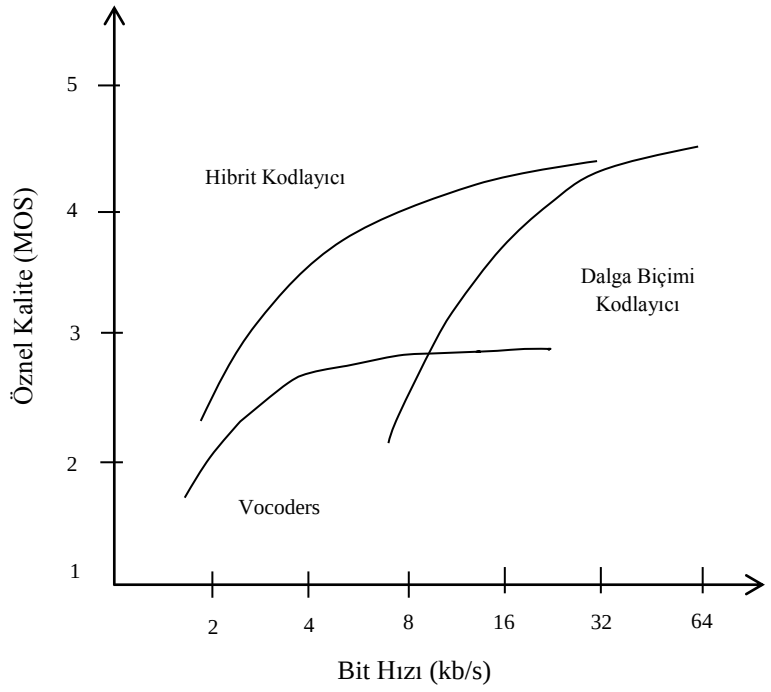
2.3.2.1. Bit hızına göre sınıflandırma

Konuşma kodlayıcılar bit hızını 128 kbps bit hızından daha düşük değerlere indirmek için tasarlanmıştır. Azaltılmış olan bit akış hızı farklı kodlama tekniklerine göre değişiklik gösterir. Belirli bir bit hızı aralığında iyi çalışan bir yöntemde, bit akış hızı belirli bir eşiğin altına düşerse sesin kalitesi kabul edilebilir sınırların altına düşmesiyle bozulmaya uğrayacaktır. Konuşma kodlayıcılar kodlanmış sinyalin bit akış hızına bağlı olarak Tablo 2.3'deki gibi sınıflandırılabilir [94].

Tablo 2.3. Bit Akış hızına göre konuşma kodlayıcıların sınıflandırılması

Category	Bit-Rate Range
High bit-rate	>15 kbps
Medium bit-rate	5 to 15 kbps
Low bit-rate	2 to 5 kbps
Very low bit-rate	<2 kbps

2.3.2.2. Kodlama tekniklerine göre sınıflandırma



Şekil 2.5. Kodlama tekniklerine göre konuşma kodlama algoritmalarının sınıflandırılması

Kodlama tekniklerine göre kodlayıcıları sınıflandırmak istersek iki genel başlık karşımıza çıkar. Bunlar dalga şeklini korumaya çalışan dalga biçimi kodlama ve sinyalin konuşma sinyalinin özelliklerini kullanarak sesin kodlanmasını sağlayan parametrik kodlama teknikleridir. Üçüncü bir sınıf olarak ta bu iki tekniğin bir arada

kullanıldığı hibrid kodlayıcılar bulunmaktadır. Şekil 2.5'te bu üç sınıf algoritmanın kalite ve bit akış hızı açısından karşılaştırılması verilmiştir.

2.3.2.3. Dalga biçimli kodlayıcılar (waveform coders)

Dalga biçimli kodlayıcılar sinyalin sadece dalga biçimini korumayı hedeflemektedir. Kodlama işlemi sırasında başka bir parametre ya da özellik kullanılmamaktadır. Bu yüzden sadece ses için değil herhangi bir sinyal için kullanılabilirler. Örneğin bir sensör çıkışındaki sinyali de bu yöntemle sıkıştırmak mümkündür. Bu teknik yüksek bit hızlı kodlamalarda daha iyi çalışır. Bu tür kodlayıcılarda kodlanmış sinyalin bit akış hızı düştükçe kalitedeki bozulma oldukça fazladır. Pulse Code Modulation (PCM) ve Adaptive Differantial PCM (ADPCM) gibi algoritmalar bu sınıf içinde yer almaktadır. Dalga biçimi kodlayıcıların kalitesi sinyal-gürültü oranı (SNR) ile kolaylıkla ölçülebilir. Uygulamada, bu kodlayıcılar 32 kbps ve daha yüksek bir bit hızlarında daha iyi çalışır.

2.3.2.4. Parametrik kodlayıcılar (parametric coders)

Parametrik kodlayıcılarda hedef dalga biçiminin orijinal şeklini korumak değildir. Ses sinyalinin oluşumunda kullanılan bazı özellikler ve parametreler kullanılarak konuşmanın tekrar elde edilebileceği bir model oluşturulur. Kod çözülerek tekrar elde edilen ses sinyali, dalga biçimi olarak orijinal sinyalden farklı olmasına rağmen anlaşılabilir bir konuşma sunmaktadır. Ama dalga biçiminin değişmiş olmasından dolayı kalite ölçümlerinde SNR gibi matematiksel ölçüm yöntemleri kullanılması doğru sonuçlar vermeyecektir. Bu tür kodlayıcıların kalite değerlendirmeleri için ACR gibi istatistiksel değerlendirme yöntemleri geliştirilmiştir. Literatürde pek çok önerilmiş model vardır. İçlerinde en başarılı yöntemler doğrusal öngörüye (lineer prediction) dayanmaktadır.

Kodlayıcıların bu sınıfı 2-5 kbps aralığı gibi düşük bit akış hızları için iyi çalışır. Bit hızı seçilen model ile sınırlandırılmış olmasından dolayı bit hızının artması normalde

daha iyi kalite anlamına gelmez. Bu sınıfın örnek kodlayıcıları linear prediction coding (LPC) ve mixed excitation linear prediction (MELP) olarak verilebilir.

2.3.2.5. Hibrid kodlayıcılar (hybrid coders)

Adından da anlaşılacağı gibi hibrid kodlayıcılar dalga biçimli kodlayıcıların ve parametrik kodlayıcıların birleşimi ile oluşmaktadır. Parametrik bir kodlayıcı gibi, kodlama sırasında model parametrelerinin bulunduğu bir konuşma üretim modeli üzerine kuruludur. Buna ek olarak modelin parametreleri kodu çözülmüş konuşmanın orijinal dalga biçimine mümkün olduğunca daha yakın olacak şekilde optimize edilmiştir. Bu yakınlık sık sık bir hata sinyali ile ölçülür. Dalga kodlayıcılardaki gibi bir girişimle kodu çözülmüş sinyal ile orijinal sinyalin zaman domeninde karşılaştırması yapılır.

Bir hibrid kodlayıcı yüksek bit hızlarında bir dalga biçimli kodlayıcı gibi davranır. Düşük bit hızlarında ise parametrik kodlayıcı gibi davranır. Orta bit hızlarında ise iyi bir kaliteye sahiptir. Bu sınıfta kod uyartımlı doğrusal öngörü (code-excited linear prediction CELP) algoritması ve onun türevleri ile orta bit hızlı kodlayıcılar bulunmaktadır.

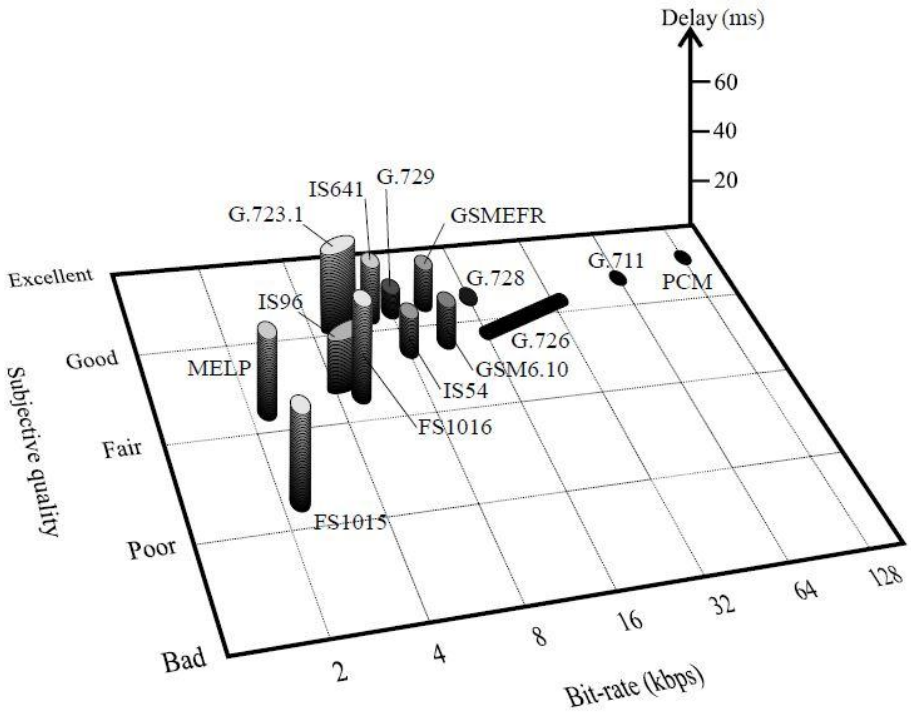
2.3.3. Konuşma kodlama standartları

Standartlar, iletişim için ortak bir araca sahip olma ihtiyacından ortaya çıkmıştır. Aldığımız veya kullandığımız bir cihazın sadece aynı marka ürünlerle değil aynı işi gören tüm ürünlerle birlikte kullanılması gerekmektedir. Bu herkesin yararına olan bir ihtiyaçtır.

Tablo 2.4 1999'a kadar geliştirilen en önemli standartları içerir. Bir standardın adı geliştirilmesinden sorumlu kurumun kısaltması ile başlar. Eğer varsa bunu bir etiket veya sayıdan oluşan kodlayıcı numarası takip eder ve sonunda algoritmanın kullanılan yöntemini belirten isminin kısaltması bulunur.

Tablo 2.4. Başlıca konuşma kodlama standartları

Yıl	Standardın Adı	Bit Hızı (kbps)	Uygulama Alanı
1972	ITU-T G.711 PCM	64	Genel amaçlı
1984	FS 1015 LPC	2.4	Güvenli haberleşme
1987	ETSI GSM 6.10 RPE-LTP	13	Dijital mobil radyo
1990	ITU-T G.726 ADPCM	16, 24, 32, 40	Genel amaçlı
1990	TIA IS54 VSELP	7.95	Kuzey Amerika TDMA dijital hücresel telefon
1990	ETSI GSM 6.20 VSELP	5.6	GSM
1990	RCR STD-27B VSELP	6.7	Japonya dijital hücresel telefon
1991	FS1016 CELP	4.8	Güvenli haberleşme
1992	ITU-T G.728 LD-CELP	16	Genel amaçlı
1993	TIA IS96 VBR-CELP	8.5, 4, 2, 0.8	Kuzey Amerika CDMA dijital hücresel telefon
1995	ITU-T G.723.1 MP-MLQ / ACELP	5.3, 6.3	Multimedia haberleşmesi, görüntülü telefon
1995	ITU-T G.729 CS-ACELP	8	Genel amaçlı
1996	ETSI GSM EFR ACELP	12.2	Genel amaçlı
1996	TIA IS641 ACELP	7.4	Kuzey Amerika TDMA dijital hücresel telefon
1997	FS MELP	2.4	Güvenli haberleşme
1999	ETSI AMR-ACELP	7.40, 6.70, 5.90, 5.15, 4.75	Genel amaçlı telekomünikasyon

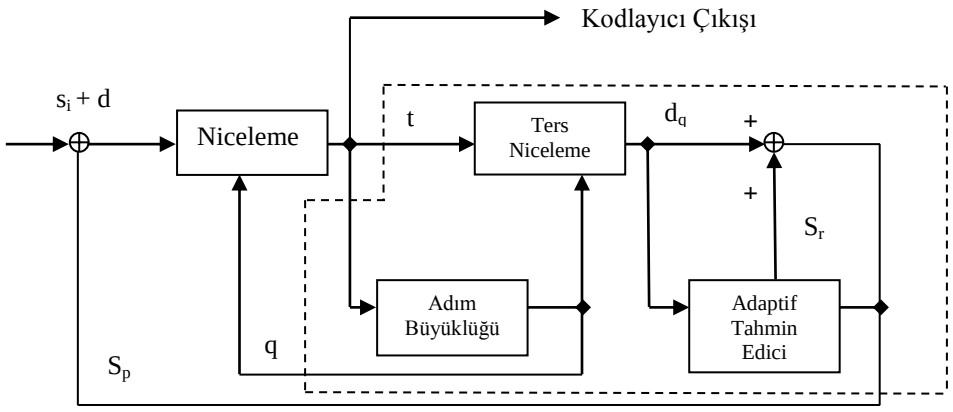


Şekil 2.6. Bazı standart kodlayıcılar için performans karşılaştırması

Şekil 2.6 çeşitli standartlar için kalite / bit hızı / gecikme karşılaştırma grafiği gösterilmiştir [94]. Burada G726 ADPCM algoritması işlem gücü açısından bakıldığında en az işlem gücü gerektiren algoritmalardan biridir. Buna rağmen bit akış hızı 16 kbps seviyelerinde çalışabilmektedir. Ayrıca kalite açısından da bakıldığında iyi bir ses kalitesi sağlamaktadır. Bu özelliklerinden dolayı da çalışmada ses kodlama algoritması olarak ADPCM ses kodlama algoritması tercih edilmiştir.

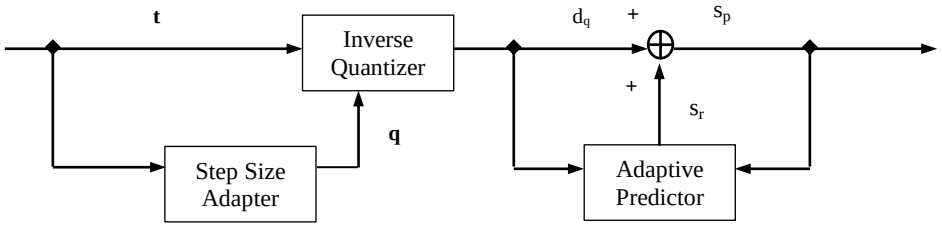
2.3.4. ADPCM ses kodlama algoritması

Interactive Multimedia Association (IMA) tarafından önerilen Uyarlanabilir Fark İşaret Kod Modlasyonu (ADPCM) algoritması örnek başına düşen bitlerin sayısını 4 kattan 1 kata kadar azaltan bir sıkıştırma faktörü sunmaktadır. IMA bilgisayar donanımı ve yazılımı satıcılarının bilgisayar multimedia verisi için geçerli bir standart geliştirmek için işbirliği yaptığı bir konsorsiyumudur. IMA'nın amacı iyi veri sıkıştırma performansı ile iyi sıkıştırılmış ses kalitesi sağlayabilen ses sıkıştırma algoritmalarının herkes tarafından kullanımı için seçilmesidir. IMA ADPCM algoritmasının basitliği, onun tahmin edicisinden kaynaklanmaktadır. Ses örneğinin tahmin edilmiş değeri basitçe hemen sonraki ses örneğinin kodu çözülmüş değeridir. Böylece Şekil 2.7'deki tahmin edici blok, bir zaman gecikmesi elemanıdır ve çıkışı sadece bir ses örneği aralığı kadar geciktirilmiş giriştir. Tahmin edicinin adaptif olmasından dolayı yan bilgi tahmin edicinin tekrar oluşturulması için gerekli değildir.



Şekil 2.7. ADPCM kodlayıcı

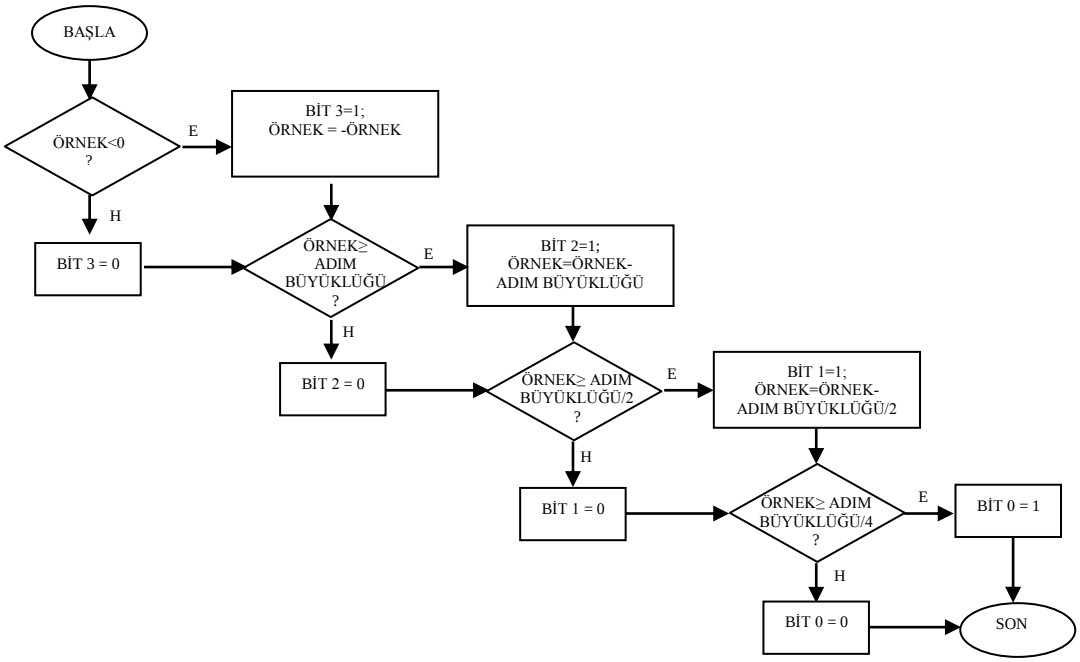
ADPCM konuşma kodlama algoritmasında kodlama işlemi sırasında her bir ses örneği bağımsız olarak ele alınmaz. ADPCM kodlayıcının temel amacı giriş işaretinin ardışık iki örneği arasındaki farkı vermesidir. PCM deki gibi her bir ses örneğinin bağımsız olarak gösterilmesi yerine bir ADPCM kodlayıcı bir sonraki örnek değerinin tahmin edilmesini mümkün kılan ardışık konuşma örnekleri arasındaki yüksek benzerliğin avantajına sahiptir. Kodlama işleminde de ADPCM kodlayıcı her bir ses örneği için tahmin edilen örnek ile konuşma örneği arasındaki farkı kodlar. Bu metot konuşma sinyalinin genel kalitesini koruyarak örnek başına bitlerin sayısında azalma ile daha etkili bir sıkıştırma sağlar. Şekil 2.7 ve Şekil 2.8 adaptive differantial pulse kod modülasyonu (ADPCM) kodlayıcının basitleştirilmiş bir blok diyagramını gösterir [95]. Şekil 2.7'deki ADPCM kodlayıcı tahmin edilmiş değeri hesaplamak için Şekil 2.8'deki ADPCM kod çözücü komponentlerinin çoğunu kullanmaktadır.



Şekil 2.8. ADPCM kod çözücü

Tahmin edilmiş örnek sp ve niceleyici adım büyüklüğü indeksi q kodlayıcının bir sonraki iterasyonu için bir yapıda saklanır. Başlangıçta, niceleyici adım büyüklüğü indeksi ve tahmin edilmiş örnek sp sıfırdır. Kodlayıcı fonksiyonu 16 bitlik bir konuşma örneği alır ve 4 bitlik işaretli ADPCM kodunu (t) geri döndürür. Şekil 2.9'da, IMA algoritması tarafından kullanılan niceleme işleminin bir blok diyagramı gösterilmektedir [79]. Tahmin edilmiş örnek 'sp', doğrusal giriş örneği 'si' den bir fark 'd' üretmek için çıkartılır. Adaptif niceleme bu fark ile gerçekleştirilir ve sonuçta 4 bit ADPCM değeri 't' elde edilir. Hem kodlayıcı hem de kod çözücü ADPCM değerine göre kendi iç değişkenlerini günceller. Kodlayıcı içerisinde tam bir kod çözücü gömülmüştür. Bu durum kodlayıcı ve kod çözücünün herhangi bir ek

veri göndermesine gerek kalmadan senkronize olmasını garanti eder. Gömülü kod çözücü Şekil 2.7'de kesikli çizgilerle gösterilmiştir. Gömülü kod çözücü ters niceleyiciyi güncellemek için ADPCM değerini 't' yi kullanır. Ters niceleyici fark 'd'nin dequantize edilmiş versiyonu olan 'dq'yu üretir. Şekil 2.7'de gösterildiği gibi dequantize edilmiş fark 'dq', yeni bir tahmin edilmiş örnek 'sr' yi üretmek için tahmin edilmiş örnek 'sp' ile toplanır. Sonunda yeni tahmin edilmiş örnek 'sr' bir sonraki iterasyonda kullanılmak için 'sp' ye atanır.



Şekil 2.9. ADPCM niceleme işlemi

Kod çözme girişi 't', 4 bitlik işaretli sayı olan ADPCM verisi olmalıdır. İzin verilen sınır aralığı 0-15 tir. Şekil 2.8 ADPCM kod çözücünün blok diyagramını gösterir. Tahmin edilmiş örnek 'sp' ve niceleyici adım büyüklüğü kod çözücünün bir sonraki iterasyonu için bir yapıda saklanır. Başlangıçta, niceleyici adım büyüklüğü indeksi ve öngörülen numune (sp) sıfırdır. Bu fonksiyon 4 bitlik ADPCM kodlarını alır ve 16 bitlik konuşma örneklerini geri döndürür. Bu kod çözücü kodlayıcı rutininde kullanılanın aynısıdır. Kod çözücü ADPCM değerini 'dq' fark değerini üreten ters niceleyiciyi güncellemek için kullanır. Fark 'dq' çıkış örneği 'sr' yi üretmek için

tahmin edilmiş örnek ‘sp’ye eklenir. Çıkış örneği ‘sr’, kod çözücünün bir sonraki iterasyonu için tahmin edilmiş örnek sp ye aktarılır.

Tablo 2.5. Adım büyüklüğü dizisi

İndeks	Adım büyüklüğü	İndeks	Adım büyüklüğü	İndeks	Adım büyüklüğü	İndeks	Adım büyüklüğü
0	7	22	60	44	494	66	4,026
1	8	23	66	45	544	67	4,428
2	9	24	73	46	598	68	4,871
3	10	25	80	47	658	69	5,358
4	11	26	88	48	724	70	5,894
5	12	27	97	49	796	71	6,484
6	13	28	107	50	876	72	7,132
7	14	29	118	51	963	73	7,845
8	16	30	130	52	1,060	74	8,630
9	17	31	143	53	1,166	75	9,493
10	19	32	157	54	1,282	76	10,442
11	21	33	173	55	1,411	77	11,487
12	23	34	190	56	1,552	78	12,635
13	25	35	209	57	1,707	79	13,899
14	28	36	230	58	1,878	80	15,289
15	31	37	253	59	2,066	81	16,818
16	34	38	279	60	2,272	82	18,500
17	37	39	307	61	2,499	83	20,350
18	41	40	337	62	2,749	84	22,358
19	45	41	371	63	3,024	85	24,623
20	50	42	408	64	3,327	86	27,086
21	55	43	449	65	3,660	87	29,794
						88	32,767

Kodlayıcı girişi $X_{[n]}$, ADC çıkışından alınan 16 bitlik konuşma verisidir. $X_{[n]}$ için izin verilen değerler aralığı 32,767 ile -32.768 dir. 16 bitlik bu değer sıkıştırma işlemi sonrasında 4 bitlik bir koda indirgenir ve 16 bitlik her bir giriş örneği için 4 bitlik bir kod elde edilmiş olur. Böylece giriş konuşma sinyali veri miktarı 4 kat azaltılmış olur. Elde edilen bu dört bitlik kod işaretli bir sayı şeklindedir. Bu kodun en yüksek değerlikli biti işaret biti olarak kullanılır ve iki ardışık konuşma örneği arasındaki farkın artı ya da eksi yönde olduğunu gösterir. Kalan üç bit ise Şekil 2.9’daki niceleme işlemi ile ADPCM kodlayıcı ya da kod çözücü içerisinde bulunan adım büyüklüğü hesaplaması ile birlikte ardışık iki konuşma örneği arasındaki farkı

belirler. Tablo 2.5 gösterilen adım büyüklüğü 80 elemanlı bir dizidir ve dizi elemanları yaklaşık olarak 1,1 oranında artar ve azalır.

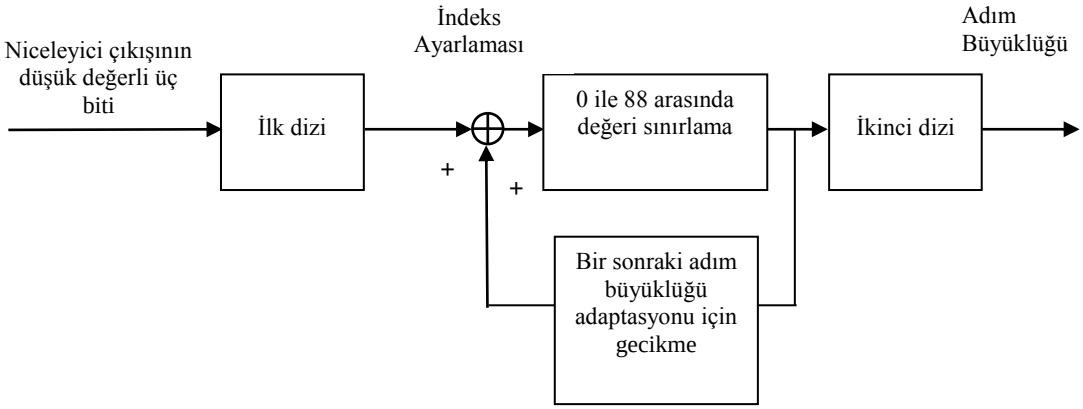
Adım büyüklüğünün belirlenmesi için Tablo 2.6'deki ikinci bir dizi kullanılarak bu dizi ile adım büyüklüğü dizisinin indekslenmesi sağlanmıştır. Bu dizi 4 bitlik ADPCM değerini kullanarak sinyaldeki değişim hızını tespit etmek için kullanılacaktır. ADPCM kodu ile indekslenen bu dizi sürekli olarak indeks değerinin üzerine eklenir. Burada ADPCM değeri iki örnek arsındaki farkı verdiği için bu değer indeks table dizisini adresler ve elde edilen değer index değerinin üzerine eklenir. Daha sonra index değeri adım büyüklüğü dizisini indeksler. Bu durumda sinyalin değişim hızı büyük ise adım büyüklüğü dizisinden büyük bir değer üretilecek, değişim hızı küçük ise adım büyüklüğü dizisinden küçük bir değer seçilecektir.

Tablo 2.6. İndeks dizisi

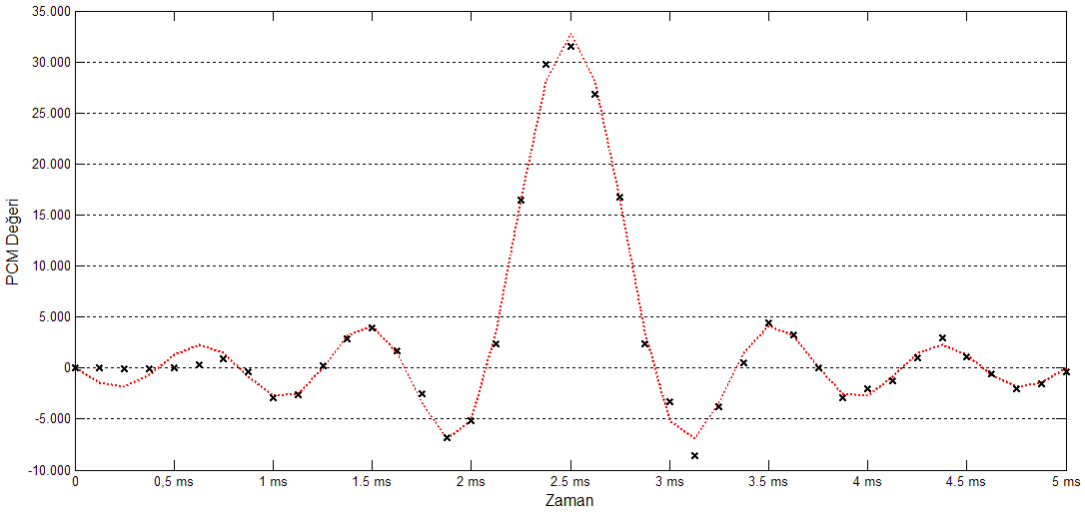
Üç bit niceleme değeri	İndeks ayarlaması
000	-1
001	-1
010	-1
011	-1
100	2
101	4
110	6
111	8

Ses sinyalinin adapte edilmesi sadece niceleyici bloğu içinde yer alır. Niceleyici adım büyüklüğünü, o anki adım büyüklüğüne ve hemen sonraki girişin niceleyici çıkışına bağlı olarak adapte eder. Bu adaptasyon iki dizi ile yapılabilir. Niceleyici seviyesinin sayısını gösteren 3 bit, ilk dizinin indeksini belirlemek için kullanılır. Bu ayarlama kaydedilmiş indeks değerine eklenir ve aralığı limitlenmiş sonuç ikinci dizi

için indeks olarak kullanılır. Toplanmış indeks değeri adım büyüklüğü adaptasyonunun bir sonraki adımında kullanılmak için kaydedilir. İkinci dizi çıkışı yeni niceleyici adım büyüklüğüdür. Şekil 2.10, adım büyüklüğü adaptasyon işleminin bir blok diyagramını göstermektedir.



Şekil 2.10. Adım büyüklüğü adaptasyonu



Şekil 2.11. ADPCM algoritmasının girişine uygulanan test sinyali

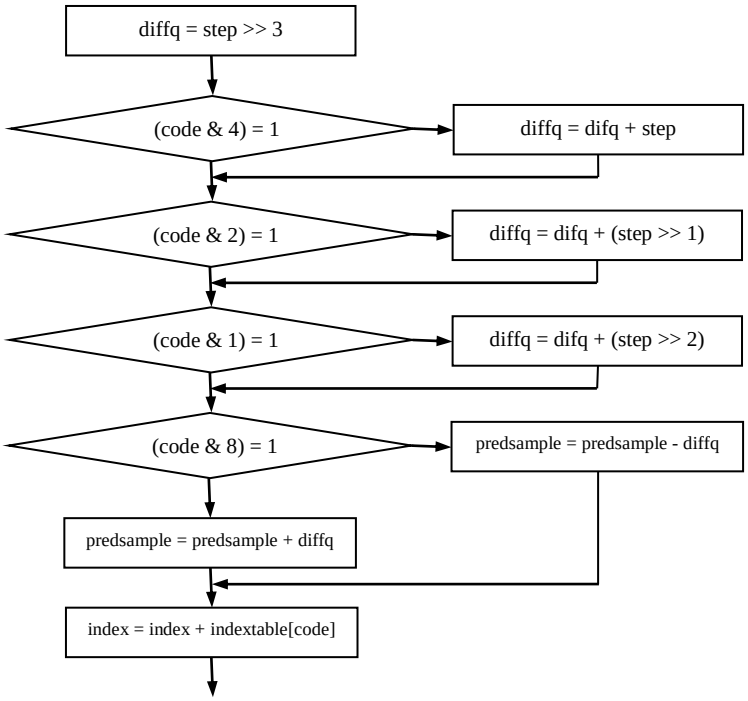
ADPCM kodlayıcının çalışmasını incelemek için Şekil 2.11'de verilen dalga şekli kodlayıcının girişine uygulanmış ve sonra tekrar kodu çözülerek elde edilen dalga şekli tekrar çizilmiştir. Bu şekilde giriş ve çıkış sinyali arasındaki farklar rahatça görülmektedir.

Tablo 2.7. ADPCM kodlama işlemi sırasında kullanılan değişkenlerin aldığı değerler

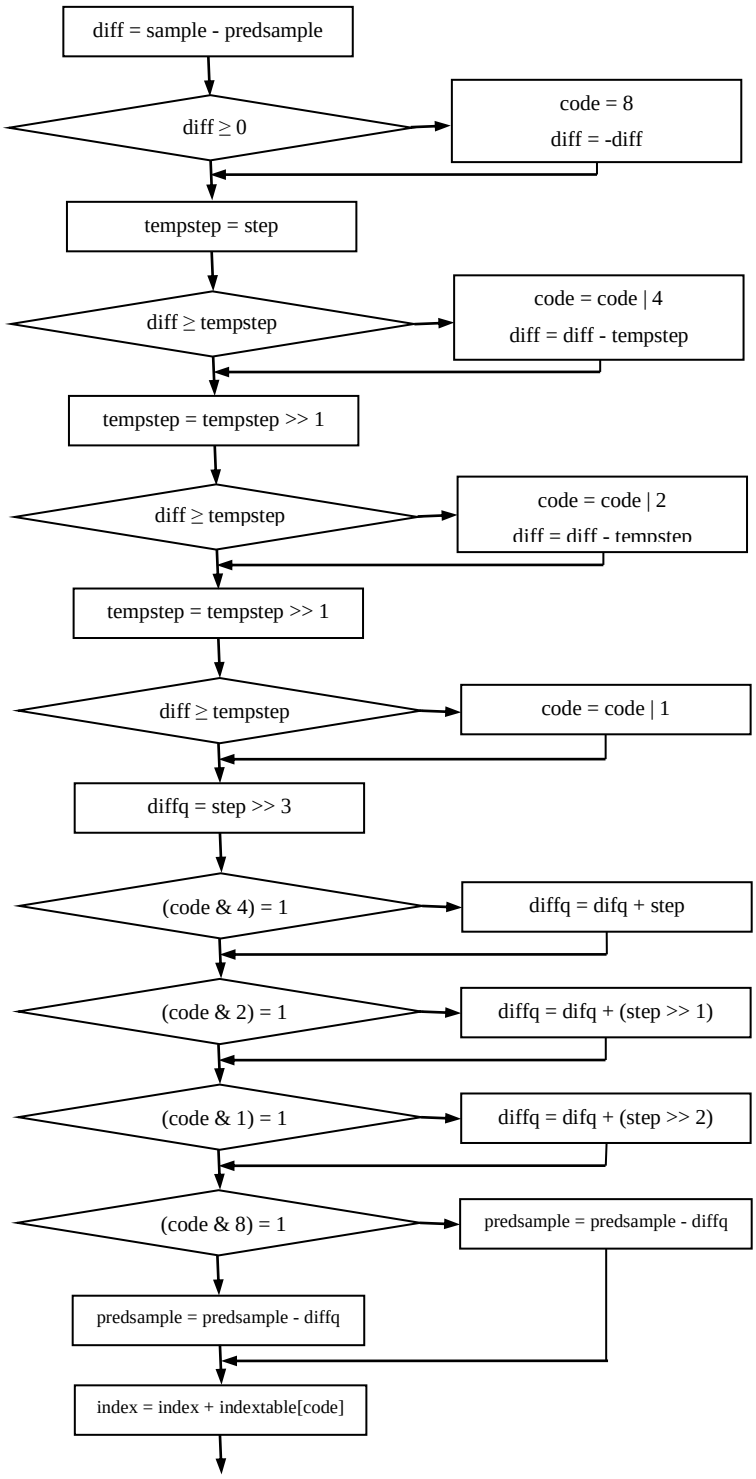
Örnek no	sample	pred sample	diff	step	code	index	İndex table
1	0	0	0	7	8	1	-1
2	-1480	-11	1470	7	15	9	8
3	-1837	-41	1802	16	15	17	8
4	-632	-104	540	34	15	25	8
5	1277	32	1272	73	7	33	8
6	2318	325	2051	157	7	41	8
7	1460	956	630	337	7	49	8
8	-826	-401	696	724	15	57	8
9	-2755	-2923	26	1552	14	63	6
10	-2557	-2580	366	2749	0	62	-1
11	0	231	81	2499	4	64	2
12	3125	2877	1382	3024	3	63	-1
13	4133	3907	1256	2749	1	62	-1
14	1535	1722	1123	2499	11	61	-1
15	-3406	-2538	1720	2272	15	69	8
16	-6953	-6798	1980	4871	11	68	-1
17	-5109	-5138	1689	4428	1	67	-1
18	3581	2410	2680	4026	7	75	8
19	16533	16433	1178	8630	6	81	6
20	28127	29810	4050	15289	3	80	-1
21	32767	31547	2957	13899	0	79	-1
22	28127	26810	3420	12635	9	78	-1
23	16533	16761	4534	11487	11	77	-1
24	3581	2404	2738	10442	13	81	4
25	-5109	-3329	7513	15289	9	80	-1
26	-6953	-8540	3624	13899	9	79	-1
27	-3406	-3803	5134	12635	1	78	-1
28	1535	503	5338	11487	1	77	-1
29	4133	4418	3630	10442	1	76	-1
30	3125	3232	1293	9493	8	75	-1
31	0	-3	3232	8630	9	74	-1
32	-2557	-2944	2554	7845	9	73	-1
33	-2755	-2053	189	7132	0	72	-1
34	-826	-1243	1227	6484	0	71	-1
35	1460	966	2703	5894	1	70	-1
36	2318	2974	1352	5358	1	69	-1
37	1277	1149	1697	4871	9	68	-1
38	-632	-511	1781	4428	9	67	-1
39	-1837	-2020	1326	4026	9	66	-1
40	-1480	-1563	540	3660	0	65	-1
41	0	-317	1563	3327	1	64	-1
42	0		0				-1

Tablo 2.7'de giriş sinyali ve bu giriş sinyaline karşı algoritmada kullanılan değişkenlerin değerleri verilmiştir. Bu tabloyu inceleyerek algoritmanın çalışması kolayca çözülebilmektedir. Şekil 2.11'de verilen sinyaller incelendiğinde başlangıçta 0,7msn'ye kadar çıkış sinyalinin giriş sinyalini izlemediği görülmektedir. Bunun nedeni başlangıçta giriş değerlerinin sıfır olmasından kaynaklanmaktadır. Bu nedenle algoritmanın uygun adım büyüklüğünü seçmesi giriş değerini takip edebilmesi için zaman geçmektedir. Bu süreden sonra ise nerdeyse giriş sinyali ile çıkış sinyali aynı değerdedir. Giriş sinyalinin çıkış sinyalini takip edemediği durumlar sinyaldeki değişimin en fazla olduğu durumlardır. Bu durumda da adım büyüklüğü değişkeni en büyük değerlerini aldığı anlardır. Adım büyüklüğü değeri arttıkça ses sinyalinin iki örneği arasındaki farkı ifade etme hassasiyeti de azalacağından giriş sinyalini takip etmesi de zorlaşacaktır.

Şekil 2.12'de ADPCM kodçözücünün, Şekil 2.13'te ADPCM kodlayıcının akış diyagramı verilmiştir.



Şekil 2.12. ADPCM kod çözücü algoritmasının akış diyagramı



Şekil 2.13. ADPCM kodlama algoritmasının akış diyagramı

2.3.5. Kalite değerlendirilmesi

Sentetik sesin kalitesi ses kodlayıcıların (speech coder) en önemli özelliklerinden biridir. Sinyal gürültü oranı (signal-to-noise ratio, SNR) PCM ve ADPCM gibi dalga biçimi (waveform) kodlayıcı algoritmaların kalitesini değerlendirmek için sık tercih edilen bir yöntemdir. PCM ve ADPCM gibi dalga biçimi kodlayıcıların performansı sinyal gürültü oranı (SNR) nin bazı formları kullanılarak kolayca ölçülebilir [94].

Orijinal ses sinyali x[n] ile kod çözücü çıkışındaki ses sinyali de y[n] ile verilmiş olsun. SNR aşağıdaki formülle tanımlanır. Burada n ölçüm aralığını kapsayan zaman indeksidir. Hesaplanan SNR değeri desibel cinsinden bir sonuç verir. Bir müzik kasetinin SNR oranı 60 db, müzik CD'sinin ise 100 db'dir.

$$SNR = 10log_{10}\left(\frac{\sum_{n=0}^{N} x[n]^2}{\sum_{n=0}^{N}(x[n]^2 - y[n]^2)}\right) \quad (2.1)$$

2.4. Sırörtme

Sırörtme nesne içerisine veri gizleme olarak da bilinir. Gizli bir mesajı hiç kimsenin haberi olmadan kuşku uyandırmayacak şekilde bir başkasına göndermek asıl amaçtır. Sırörtme, Yunanca kaplamak, örtmek anlamına gelen "stegos" ve yazı anlamına gelen "graphia" kelimelerinden türetilmiştir [96].

Özellikle Amerika Birleşik Devletlerinde yaşanan 11 Eylül saldırıları sonrasında sırörtme üzerine araştırmalar artmıştır [96]. Bu saldırı öncesinde, teröristlerin eBay, Usenet ve Amazon gibi popüler web sitelerinde eylem detaylarını, planlarını resimler içerisine gizleyerek paylaştıkları iddia edilmişti. Örneğin terörist saldırı ile ilgili bilgileri sırörtme yöntemlerini kullanarak bir resmin içerisine gömmüşler ve bunu Internet üzerinden yayınlamışlardır. Hangi siteden yayın yapıldığını bilen diğer gurup elemanı ilgili resmi siteden indirerek gömülmüş bilgiyi resim içerisinden çıkartarak asıl bilgiye ulaşmıştır. Bu yöntemle hiçbir şekilde dikkat çekmeksizin istedikleri gibi haberleşme imkânı bulmuşlardır.

Şifreleme haberleşme kanalı dinlendiğinde şifreli haberleşme yapıldığı kolaylıkla anlaşılabilir ve şifre çözme amaçlı saldırılara maruz kalınabilir. Stenografide ise gönderilmek istenen bilgi başkalarının eline geçmesinde sakınca olmayan bir bilgi üzerine saklandığından kolay bir şekilde tespiti yapılamaz. Bu yöntemde, şifrelemenin zayıflığı olan şifreli haberleşme bilgilerinin gözlenerek fark edilmesi ve düzenlenecek saldırıların engellenmesi amaçlanmıştır. Sırörtme yönteminde en kritik nokta, yapılan saldırılardan korunmak değil saldırıların yapılmasını önlemektir.

Bilgisayar ile sırörtme iki temel prensip üzerine kurulmuştur. Birincisi resim veya ses dosyalarının, sahip oldukları özellikleri yitirmeden değiştirilebilmeleri ilkesidir. İkincisi ise, insanın, renk veya ses kalitesinde meydana gelen küçük değişiklikleri ayırt edememesine dayanmaktadır. Renk tonlarında ve ses şiddetinde değişiklikler yapılarak orijinal resim ve ses dosyasında fark edilmeyecek değişiklikler yapılarak veri gömme işlemi gerçekleştirilir.

Sırörtme uygulamalarında kullanılan en düşük değerlikli bit (LSB- Least Significant Bit) gömme yöntemidir. İlk olarak kullanılan bu yöntem, aynı zamanda en basit yöntemdir [96,97]. Bu yöntem ile bir ses dosyasının içerisine veri gömmek istersek her bir ses örneğinin son bitini değiştirerek gömmek istediğimiz verileri bit bit bu ses örnekleri üzerine saklarız. İnsan kulağı tarafından algılanamayacak seviyede olan bu değişimler sayesinde, veri gizlenmiş bir sesin dinlenilerek tespiti mümkün değildir.

Sırörtmede kullanılan kavramlar aşağıda özetlenmiştir.

Örtü Dosyası (Cover–Image): İçerisine gizli bilgi bulunduran dosyanın gömüleceği dosyadır. Bu dosya resim, video veya ses dosyası olabilir.

Gömü Dosyası (Stego–Image): Gizli bilgiye sahip dosyadır.

Örtü Anahtarı (Stego–Key): Gizleme işlemi sırasında kullanılan güvenlik anahtarıdır.

Steganalysis: gizli bilginin bulunması ile ilgili uğraşan bilim dalıdır.

2.4.1. ADPCM stenografi

ADPCM kodlarının üzerine veri gömme işleminde kullanılması sırasında bütün kod değerlerinin kullanılması söz konusu değildir. Sıkıştırılmış kodlar ses sinyalindeki değişimleri ifade ettiği için bazı kodlar seste çok büyük değişimler meydana getirir bazı kodlar ise çok az değişim meydana getirirler. Veri gömme işlemini büyük değişimler getiren kodlar üzerinde yapmak ses sinyalinin bozulmasına neden olacaktır. Bu nedenle veri gömme işlemi için ses sinyalinde en az değişim meydana getiren kodlar seçilerek veri gömme işlemi bu kodlar üzerinde yapılmalıdır. ADPCM kodlayıcıda ses sinyalindeki değişimler iki parametre ile belirlenir. Bunlardan birincisi ADPCM kodu, ikincisi ise algoritma içinde kullanılan adım büyüklüğü değeridir. Sesteki değişimin en az olduğu durumda adım büyüklüğü 7 değerini, kod değişkeni ise 0 yada 8 değerini alır. Tabii bu durum gürültüsüz ortamlarda söz konusudur gürültülü ortamlarda ise gürültüden dolayı bu değerlerin oluşması söz konusu değildir. Gürültülü bir ortam için bu işlem yapılacaksa belli bir eşik değeri belirlenmeli ve bunun altındaki değerlere veri gömme işlemi uygulanmalıdır. Naofumi Aoki tarafından yapılan bir çalışmada LSB tekniği ile veri gömme işlemi yapmak için aşağıdaki kod kullanılmıştır [98].

```
if (     s == 7  &&     (c & 0x7) == 0 )
         {
                  if (b == 0) c &= 0x7;
                  if (b == 1) c |= 0x8;
         }
```

Yukarıdaki verilen veri gömme kodunda ‘s’ değişkeni adım büyüklüğünü ‘c’ değişkeni ADPCM kodunu göstermektedir. Burada istenen şart adım büyüklüğü yani ‘s’ değişkeninin minimum olması, kod değişkeninin de yani ‘c’ nin en az değişime neden kod olmasıdır. Buda kod değişkenini 0 ya da 8 değerini aldığı durumlardır. Daha sonra tespit edilen bu değerlerin işaret değerini gösteren en yüksek değerlikli

biti değiştirilerek veri gömme işlemi gerçekleştirilecektir. ‘b’ değişkeni yani gömülmek istenen veri bitleri eğer ‘0’ değerinde ise en yüksek değerlikli bit sıfıra kurulacak , ‘1’ değerinde ise en yüksek değerlikli bit bire kurulacaktır. Veri çıkartım işleminde ise tekrar kod değişkeninin 0 ya da 8 değerini aldığı durumlar tespit edilir ve en yüksek değerlikli biti olan işaret bitinin aldığı değerler toplanarak veri tekrar elde edilir.

Naofumi Aoki tarafından yapılan çalışma kayıpsız veri gömme işlemini yaparak sinyal üzerinde herhangi bir bozulma meydana getirmemektedir. Ama çalışmanın sonuçlarında gürültülü ortamlarda veri gömme işleminin gerçekleştirilemediği belirtilmiştir.

2.5. Kriptografi

Kriptografinin temel amacı gizli bilgilerin, istenilmeyen kişilerin eline geçmeden hedefe gönderilmesidir. Özellikle haberleşme alanında büyük önem kazanmaktadır. Birbirinden uzak mesafelerdeki iki kullanıcının paylaştığı bilgiler bir başkası tarafından erişilmeden hedefine ulaşabilmelidir. Gizli haberleşme yöntemleri eski çağlardan itibaren kullanılmıştır. Teknoloji değişip geliştikçe farklı yöntemler geliştirilmiştir.

Yunanca kryptós (gizli) ve lógos (kelime) kelimelerinden türetilen kriptoloji, genellikle gizli formda, güvenli haberleşme ile ilgilenen matematik biliminin bir dalıdır [96]. Kriptolojide orijinal düz metin (plaintext), rasgele olarak anlamsız şifreli metine (ciphertext) dönüştürülmektedir. Yani yazılı mesajların anlaşılmaz bilgilere dönüştürülmesi esasına dayanarak çalışır. Şifreli metin şifre çözme algoritması kullanılarak tekrar orijinal haline dönüştürülmektedir. Bu yöntemde yetkisiz kişiler haberleşme bilgilerine bakarak gizli bilginin varlığından haberdar olabilir fakat içeriği hakkında bilgi sahibi olamazlar.

Kriptolojide kullanılan temel kavramlar aşağıdaki gibi özetlenebilir

Kriptoloji (Cryptology): Bilginin şifrelenmesi ve tekar şifresinin çözülmesi ile ilgili çalışan ve araştırma yapan matematik biliminin bir dalıdır. Kriptografi ve kriptanaliz olarak iki daldan oluşur.

Kriptanaliz (Cryptanalysis): şifrelenmiş bilginin gerçek anlamda çözümlenmesi ya da zayıf taraflarının bulunması işlemidir. Kısaca kodların kırılması bilimidir.

Kriptanalist (Cryptanalyst): Kriptanaliz alanında çalışanlara verilen isimdir.

Kriptografik/Şifreli Sistem: Kriptografik yöntemler ile çalışan bir şifreleme sistemidir. Sadece yetkili kullanıcıların okuyabileceği mesajların şifrelenmesi ve tekrar orijinal metnin elde edilmesini sağlar.

Kriptografik Algoritma (Cryptographic Algorithm): Mesajların şifrelenmesi veya şifre çözümlerinde kullanılan hesaplama yöntemidir.

Anahtar (Key): Bir mesajın şifrelenmesi ve şifresinin çözülmesi için kullanılan semboller ve karakterler dizisidir. Şifreleme ve şifre çözme algoritmaları için giriş olarak kullanılır.

Şifreleme (Encryption): Düz metni şifreli metne dönüştürme işlemidir.

Şifre Çözme (Decryption): Şifreli metni tekrar orijinal metne dönüştürme işlemidir.

Düzmetin (orijinal metin): Şifrelenmek istenen ve herhangi bir işleme gerek olmaksızın okunabilen bilgidir.

Şifrelimetin: Şifrelenmiş metindir. Anlamsız sayılar ve semboller den oluşur.

Cipher: Şifreleme ve şifre çözme algoritmasıdır.

Doğrulama (Authentication): Mesajı gönderenin kimliğinin veya mesajın orijinalliğinin doğrulanmasıdır.

Bütünlük (Integrity): Mesajın bozulmadığının doğrulanmasıdır.

Kripto sistemlerde kullanılan şifreleme yöntemlerini temelde iki sınıfa ayırabiliriz;

1- Simetrik Şifreleme

2- Asimetrik Şifreleme

2.5.1. Simetrik şifreleme

Simetrik şifreleme ilk olarak kullanılan şifreleme tipiydi. 1970'lerin sonlarında public-key (açık anahtar) şifrelemenin geliştirilmesine kadar kullanılan tek şifreleme tipiydi. Simetrik şifreleme aynı zamanda konvansiyonel (geleneksel) şifreleme secret-key (özel-anahtar) veya single-key (tek-anahtar) olarak da adlandırılabilir. Üç tane önemli şifreleme algoritması vardır. Bunlar ; DES, 3-DES ve AES.

Simetrik şifrelemenin kullanımındaki ana problem anahtarın gizliliğinin korunmasıdır. Bilginin güvenliği algoritmanın değil anahtarın gizliliğine bağlıdır. Diğer bir deyişle anahtarın gizli tutulması yeterlidir. Algoritmanın gizli tutulması gerekli değildir. Simetrik şifreleme bu özelliği sayesinde yaygın olarak kullanılabilir. Algoritma gizlilik gerektirmeyecek şekilde üretilebildiğinden ucuz fiyatlı veri şifreleme algoritma entegreleri geliştirilebilir. Bu entegreler yaygın bir şekilde elde edilebilir ve içlerine başka ürünlerde eklenebilir.

Simetrik algoritmalar stream cipher (akan şifre) ve blok cipher olarak ikiye ayrılabilir. Stream cipher'lar belli bir anda bir bitlik düz-metni şifreleyebilirken, blok cipher'lar pek çok biti alıp (tipik olarak modern cipher'larda 64 bit) bunları tek bir birim olarak şifrelerler.

2.5.2. Asimetrik şifreleme

Asimetrik şifreleme (public key) yöntemi simetrik şifreleme yönteminin en büyük dezavantjı olan gizli anahtarı (secret key) karşı tarafa, kimsenin öğrenmeden gönderilmesindeki zorluğu ortandan kaldırma adına geliştirilmiştir.

Asimetrik şifrelemede iki adet anahtar oluşturulur. Bu anahtarlar genel anahtar (public key) ve özel anahtar (private key) olarak adlandırılır. Genel anahtar ile veri şifrelenir özel anahtar ile de sadece şifrelenmiş verinin şifresi çözülüp orjinal hale getirilir. Genel anahtar olarak belirtilen anahtar umuma açıktır ve herkes tarafından bilinmesinde herhangi bir sakınca yoktur. Çünkü bu anahtarla sadece veri şifrelenir ve bu anahtarla şifrelenmiş veriler ancak ve ancak genel anahtara karşılık oluşturulmuş özel anahtarla çözülebilir. Bu nedenle özel anahtarın kesinlikle gizli olarak kalması gerekir.

Belirtilen yöntem şu mantıkla çalışır. İki kişi arasında bir veri alışverişi yapıldığını varsayalım. Bu iki kişi hemen bütün şifreleme kitaplarında Alice ve Bob diye geçer. Alice, Bob'a şifreli bir mesaj göndermek istemektedir. Bob kendi bilgisayarı üzerinde bir adet genel anahtar (şifreleme için) ve bir adet de özel anahtar (şifre çözmek için) oluşturur. Bob oluşturmuş olduğu genel anahtarı Alice gönderir. Yolda genel anahtarın başkaları tarafından görülmesinde herhangi bir sakınca yoktur. Simetrik şifreleme algoritmalarından farkı buradadır. Çünkü genel anahtar sadece şifreleme yapar. Alice Bob'a ait genel anahtar ile mesajını şifreler ve şifreli bir şekilde Bob'a yollar. Bob'un elinde de kendisine ait olan genel anahtara karşılık gelen özel anahtar vardır ve bu özel anahtar, sadece oluşturulmuş olan genel anahtar ile şifrelenmiş mesajları çözer ve bu sebepten ötürü hep gizli kalmalıdır. Bob şifrelenmiş mesajı özel anahtarı ile çözerek okur. Yolda metin şifrelenmiş olarak gittiğinden ve özel anahtar her zaman Bob un elinde gizli bir şekilde tutulduğundan mesajın güvenliği sağlanmıştır.

Bu genel/özel anahtar çiftini üretmek için özel bir algoritma kullanılır. Bu algoritma ilk kez Amerikalı üç bilim adamı tarafından 1977 yılında geliştirilmiştir ve ismini bu

üç kişinin baş harflerinden almış olan RSA(Rivest, Shamir, Adleman) algoritmasıdır. En sık kullanılan genel anahtar şifreleme algoritmalarına örnek olarak; RSA, Diffie-Hellman, DSS, ECC verilebilir [96].

2.5.3. Düşük işlem gücüne sahip sistemler için şifreleme algoritmaları

Bu tarz şifreleme algoritmaları mikrodenetleyici tabanlı sistemler için geliştirilmiştir. Örnek uygulama alanı olarak RFID sistemlerini verebiliriz. Bu tarz sistemlerde kod ve veri belleğinin ve işlem gücünün sınırlı olmasından dolayı şifreleme algoritmasının bu sınırlamaları göz önünde bulundurarak tasarlanması gerekmektedir. Bu tarz şifreleme algoritmalarına örnek olarak TEA (Tiny Encryption Algorithm), XTEA (Extended TEA) ve SEA (Scalable Encryption Algorithm) verilebilir. TEA algoritması kırılabildiği için günümüzde geçerliliğini yitirmiştir.

Bu çalışmada şifreleme biriminde SEA ve XTEA algoritmalarının kullanılmasının en önemli nedenlerinden biri; algoritmanın güncel ve saldırılara karşı dayanıklı olmasının yanında bellek büyüklüğü ve işlem gücü gibi sınırlı kaynaklara sahip gömülü sistemlere yönelik olarak tasarlanmış olmasıdır. Oldukça esnek bir yapıya sahip olan SEA ve XTEA basit mantıksal ve aritmetik komutlar ile çalışan mikodenetleyicilerde rahatlıkla kullanılabilmektedir. Ayrıca yapılmış çalışmalar, güvenli ses iletişim uygulamalarında SEA ve XTEA şifreleme algoritmasının daha önce kullanılmadığını göstermektedir.

Assembly fonksiyon olarak tanımlanıp C derleyicisi ile kullanılan bu şifreleme algoritmaların assembly kodları Ek C, Ek D, Ek E, ve Ek F'de verilmiştir.

2.5.4. SEA (Scalable encryption algorithm)

SEA algoritması 2006 yılında François-Xavier Standaert ve arkadaşları tarafından geliştirilmiştir. Bu şifreleme algoritması bellek büyüklüğü ve işlem gücü gibi sınırlı kaynaklara sahip gömülü sistemlere yönelik olarak tasarlanmış bir şifreleme algoritmasıdır [99] . SEA algoritması simetrik blok şifreleme mantığı ile çalışır.

Gömülü sistemler için tasarlandığı için küçük kod ve bellek alanı kullanır ve sınırlı bir komut setine sahiptir. Bu sebeple sadece, özel veya, bit/kelime rotasyonları, mod 2b toplama ve s-box gibi bit operasyonlarını kullanır ve bu işlemleri gerçekleştirecek assembly komutlara göre tasarlanmıştır. Bu özelliğinden dolayı üst seviye programlama dilleri ile çalıştırılması yerine assembly komutlarla kullanımı daha hızlı sonuçlar vermektedir.

Şekil 2.14'te yapısı ve blok diyagramı verilen SEA oldukça esnek bir yapıya sahiptir. Farklı metin, anahtar/kelime uzunlukları üzerinde çalışabilen algoritma SEA(n,b) şeklinde ifade edilmektedir. Ayrıca, değişken tur sayılarıyla Feistel yapısına dayanan SEA, aşağıdaki parametreler ile tanımlanmaktadır [99]:

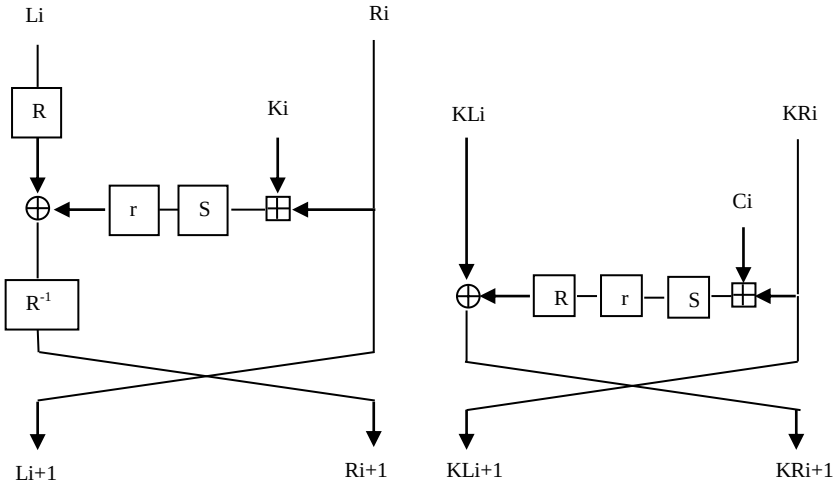
n: ham metin ve anahtar büyüklüğü
b:kelime büyüklüğü
$n_b = n/2b$: Feistel dallanması başına kelime sayısı
n_r: şifreleme tur sayısı

SEA algoritmasının gerçeklemesinde n ve b parametreleri hedef işlemcinin özelliğine uygun olarak seçilir. Bunun nedeni kullanılan mikro denetleyicinin assembly komut setine uyum sağlaması ve en hızlı şekilde çalışabilmesi içindir. Ancak anahtar ve ham metin büyüklüğü 48, 96, …, 192 bit gibi 6'nın katları olması gerekmektedir. Yeterli güvenlik düzeyinin sağlanabilmesi için kelime uzunluğunun $b \geq 8$ ve tur sayısının en az $n_r = 3n/4+2(n_b + [b/2])$ olması bir diğer önemli noktadır [99].

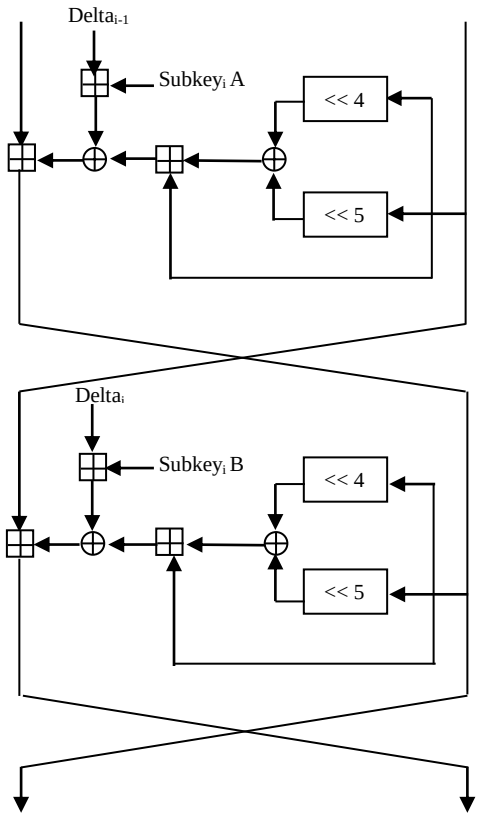
2.5.1. XTEA (Extended tiny encryption algorithm)

1995 yılında, Roger M. Needham ve David J. Wheeler tarafından geliştirilen TEA algoritması 1997 yılında üzerinde iyileştirmeler yapılıp XTEA olarak yayınlanmıştır. XTEA algoritması basit bit kaydırma ve toplama işlemlerinin tekrarlanmasından oluşan bir algoritmadır. Şekil 2.15'de algoritmanın birbirini tekrar eden iki bloğu görülmektedir. Bu blokların sayısı artıkça yapılan şifrelemenin çözülmesi gittikçe daha zorlaşmaktadır. XTEA'nın geliştiricileri sağlam bir şifreleme için bu işlemin en

az 64 kez yapılması gerektiğini belirtmekte fakat 32 tekrarın çoğu durumda yeterli olacağını belirtmişlerdir [100,101].



Şekil 2.14. Ölçeklenebilir şifreleme algoritması(SEA)

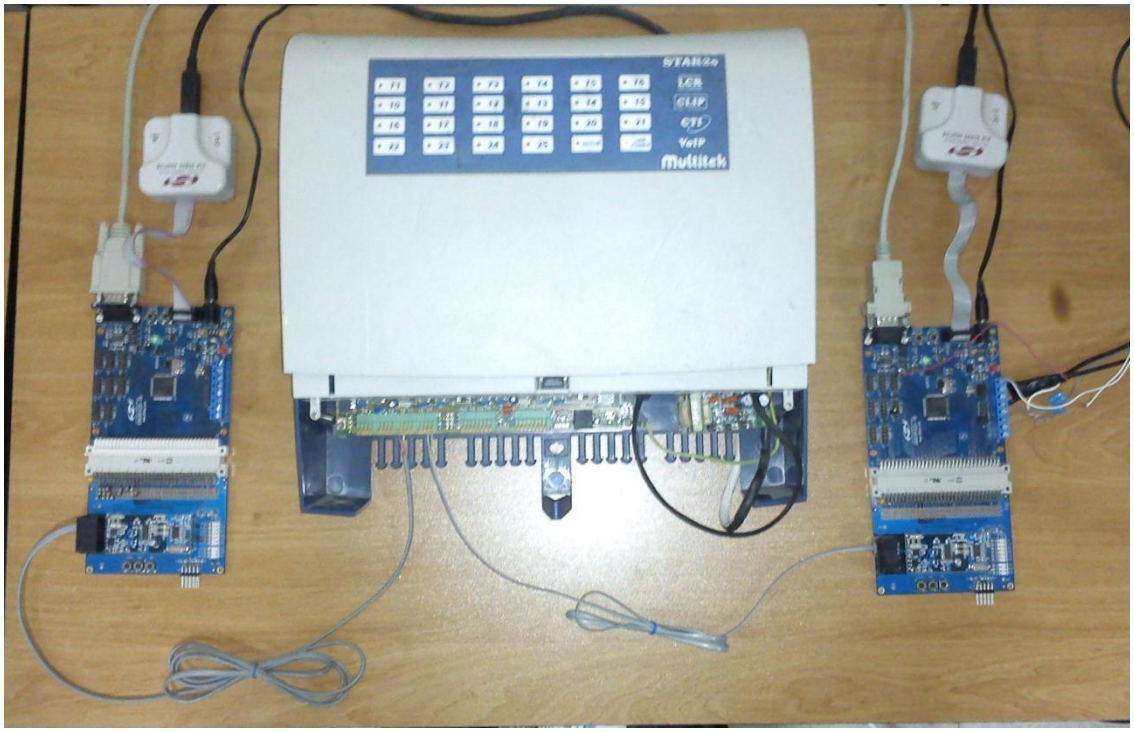


Şekil 2.15. XTEA şifreleme algoritması çevrimi

BÖLÜM 3. SİSTEMDE KULLANILAN DONANIM VE YAZILIMLARIN TANITILMASI

3.1. Kullanılan Donanımlar

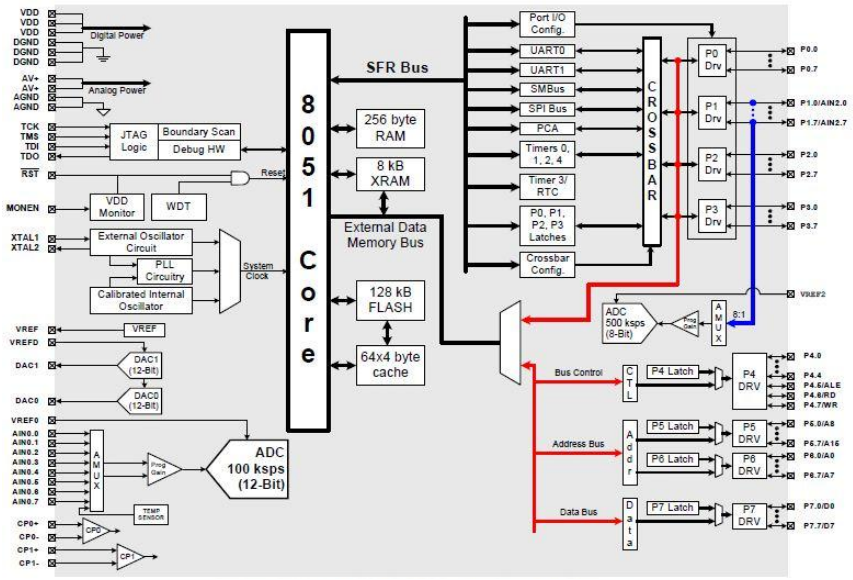
Sistemin gerçekleştirilmesi için Silicon Labrotories firmasının ürettiği C8051F120 mikrodenetleyicisi kullanılmıştır. Bu mikrodenetleyici için geliştirilmiş C8051F102DK geliştirme kartı kullanılarak sistem çalıştırılmıştır. Dial up modem olarakta yine aynı firmanın ürettiği SI2457 entegresi için geliştirilmiş MODEMDK geliştirme kartı kullanılmıştır. Sistemin çalıştırılması için telefon santrali ile iki modül birbirine PSTN üzerinden bağlanmıştır.



Şekil 3.1. Sistemin resmi

3.1.1. C8051F120 mikrodenetleyicisi

C8051F120 mikrodenetleyicisi 100 pinli 64 adet giriş çıkış ucu bulunan bir mikrodenetleyicidir. Sinyal işleme amaçlı geliştirilmiş olan bu mikrodetleyici, sistem üzerinde programlanabilme özelliği ile program geliştirme açısından büyük kolaylıklar sağlamaktadır. Bu mikrodenetleyicinin sahip olduğu temel özellikler aşağıda listelenmiştir.



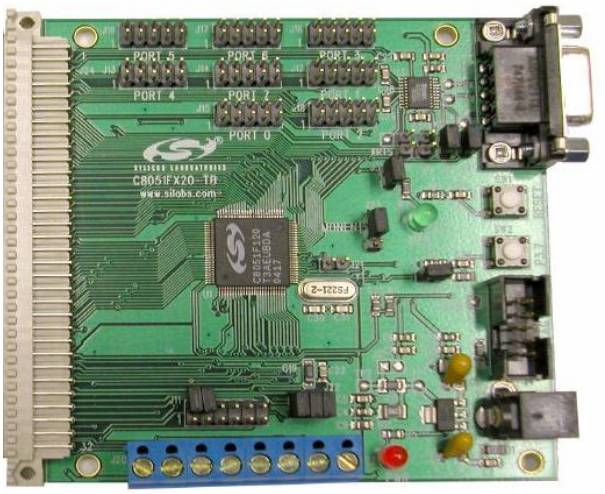
Şekil 3.2. C8051F120 mikrodenetleyicisi blok diyagramı

a. 100MIPS işlem gücüne sahip, 8051 uyumlu yüksek hızlı pipeline işlemcidir.
b. 12 veya 10 bit, 100 Ksps 8 kanal ADC
c. 8 bit, 500 Ksps 8 kanal ADC
d. 2 adet 12 bit DAC
e. 16 bit MAC devresi(Multiply and Accumulate Engine)
f. 128 KB sistem üzerinde programlanabilen flash memory
g. 8 KB ve 256 bayt RAM
h. 64 KB harici bellek adresleme kapasitesi
i. 2 adet UART seri iletişim birimi
j. 5 adet zamanlayıcı
k. 6 adet Programmable Counter/Timer Array(PCA)

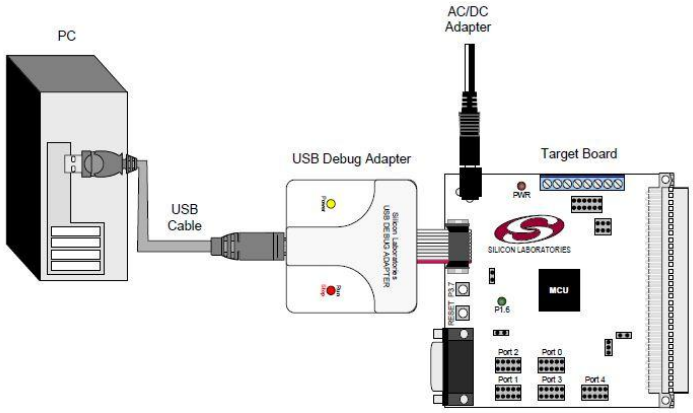
l. JTAG bağlantısı ile devre üzerinde debuging

3.1.2. C8051F120DK mikrodenetleyi kartı

C8051F120 mikrodenetleyicisi için geliştirilmiş bu kart ile hızlı ve kolay bir şekilde program geliştirmek mümkündür. Geliştirme kartı üzerindeki soketler sayesinde entegrenin bütün uç bağlantıları kullanılabilmektedir. Kullanılan ek bir USB Debug Adaptör ile kart üzerinde hiçbir değişiklik yapmadan direkt bilgisayar ortamında yazılan program karta yüklenip çalıştırılabilmekte ve istendiğinde durdurulup yeni program yüklenebilmektedir. Özellikle entegre üzerinde hata ayıklama (on chip debuging) özelliği sayesinde gerektiğinde gerçek zamanlı olarak adım adım çalıştırma yapılabilir ve entegre içindeki değişkenlerin durumları hata ayıklama için izlenebilir. Şekil 3.4'te geliştirme kartının çalıştırılması için gerekli bağlantılar gösterilmiştir.



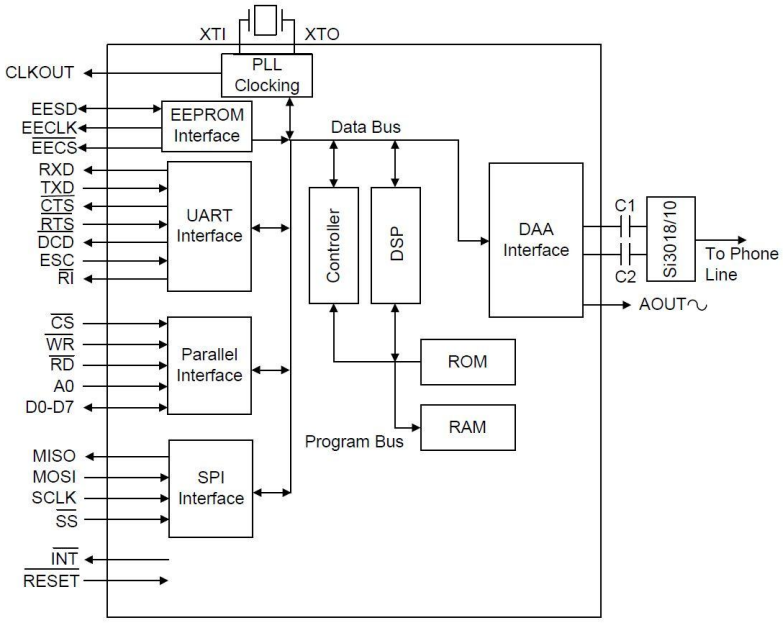
Şekil 3.3. C8051F120DK geliştirme kartı



Şekil 3.4. C8051F120DK geliştirme kartının bilgisayara bağlantısı

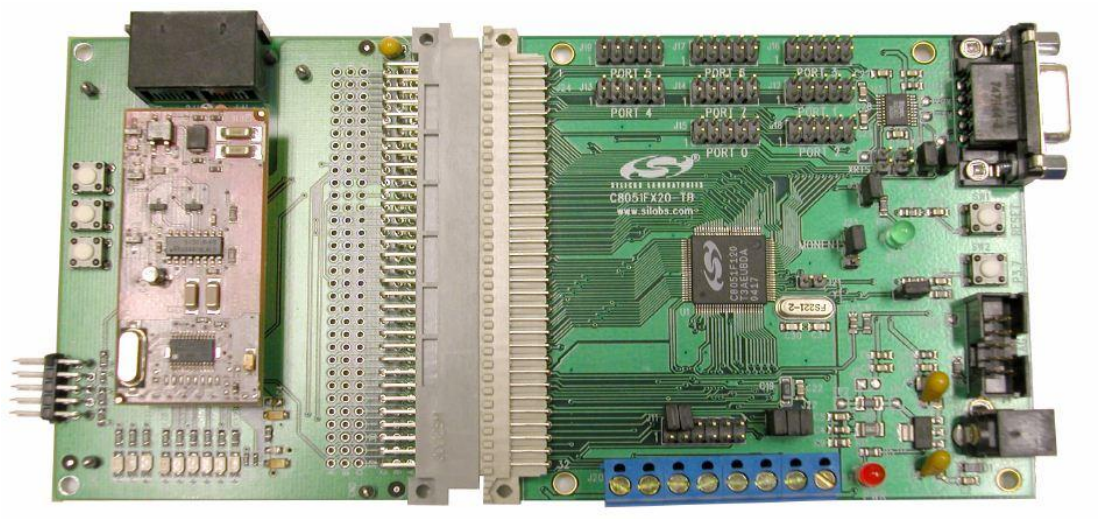
3.1.3. SI2457 dial up modem entegresi

Telefon uygulamaları için geliştirilmiş olan bu entegre telefon hattı üzerinden 2400-56000 bps bit akış hızlarında dial-up bağlantı sağlamak için kullanılmaktadır. UART yada paralel arabirim tarafından bu entegreyi programlayıp kullanmak mümkündür. PSTN üzerinden sayısal veri haberleşmesini sağlayan bu entegre AT komut setini desteklemektedir.



Şekil 3.5. SI2457 blok diyagramı

3.1.4. MODEMDK geliştirme kartı



Şekil 3.6. MODEMDK geliştirme kartı

Şekil 3.6'da görüldüğü gibi modem kartı C8051F120DK geliştirme kartına bir soket ile bağlanıp kullanılabilmektedir. Kart üzerindeki soketler ile telefon hattına bağlanabilmektedir. Ayrıca paralel telefonda bağlanabilmektedir.

3.1.5. Telefon santrali

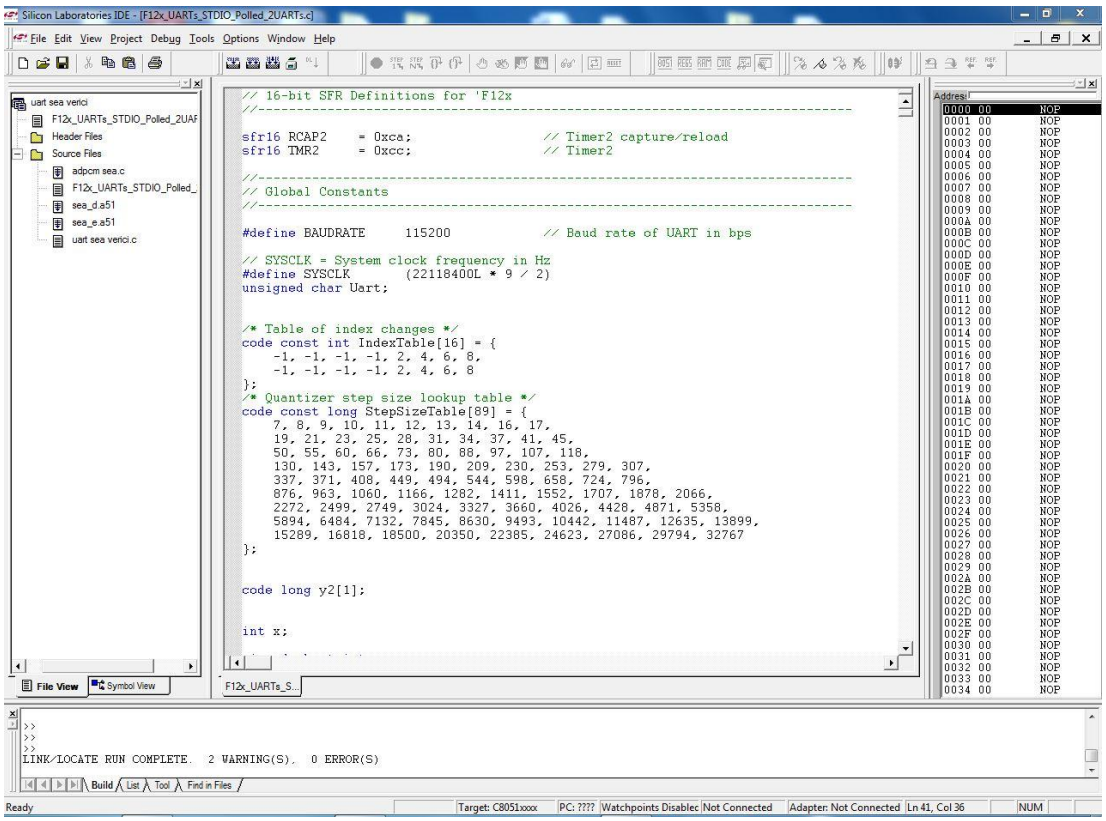
Şekil 3.7. Telefon santrali

Denemelerde PSTN hattını gerçekleştirmek için bir telefon santrali kullanılmış ve dahili arama özellikleri ile bu işlem gerçekleştirilmiştir.

3.2. Yazılım Geliştirme Araçları

Uygulamada seçilen entegreler Silicon Laboratories firmasınına ait olduğu için program geliştirmede de bu firmanın geliştirdiği yazılımlar kullanılmıştır. Sistemin test edilmesi sırasında ise Matlab ortamında geliştirilmiş arayüz kullanılmıştır.

3.2.1. Silicon laboratories IDE

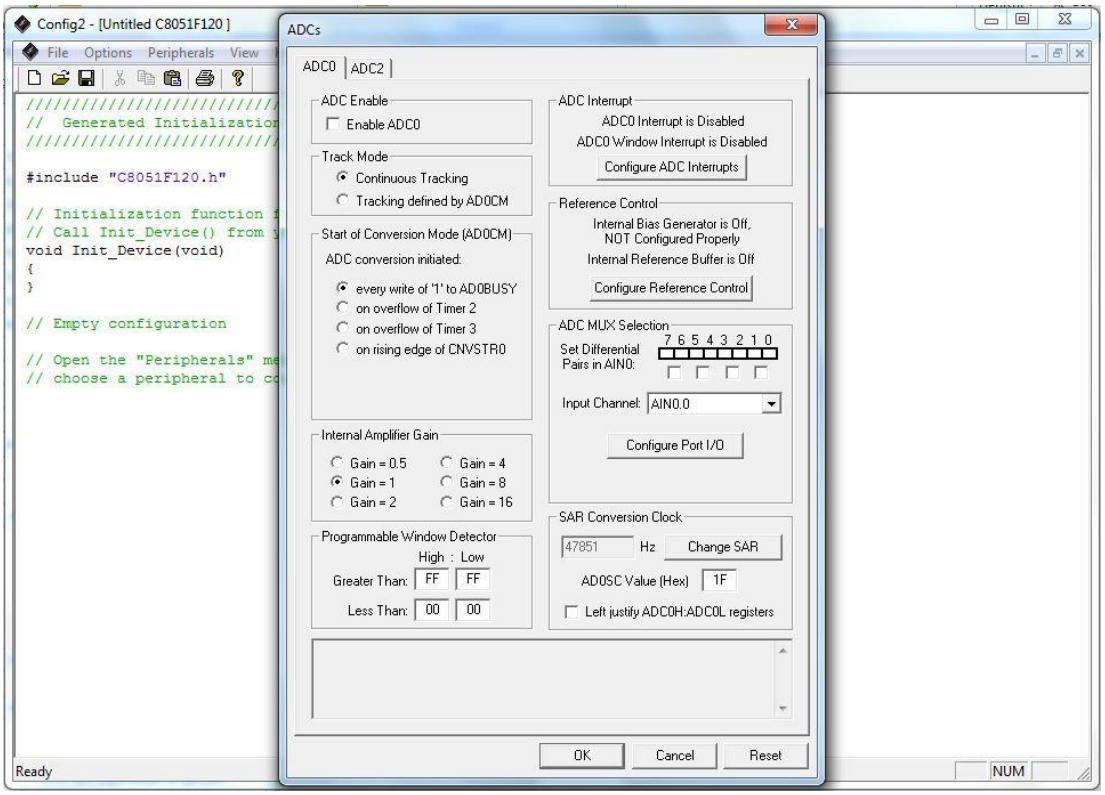


Şekil 3.8. Silicon Laboratories IDEyazılımı ekran görüntüsü

Silicon Laboratories IDE yazılımı Silicon Laboratories firmasının 8051 tabanlı mikrodenetleyicilerinin programlanması için geliştirmiş olduğu bir yazılımdır. Kod geliştirme, programlama ve hata ayıklama işlemlerinin hepsi bu programla yapılabildiği için büyük kolaylıklar sağlamaktadır. Ayrıca bu işlemler sırasında

donanıma müdahale etmeye de gerek kalmamaktadır. Bu yazılım sayesinde entegre üzerinde çalışan programı adım adım çalıştırırken aynı zamanda kaydedici ve hafıza alanı içerikleri de takip edilebilmektedir

3.2.2. Konfügürasyon sihirbazı

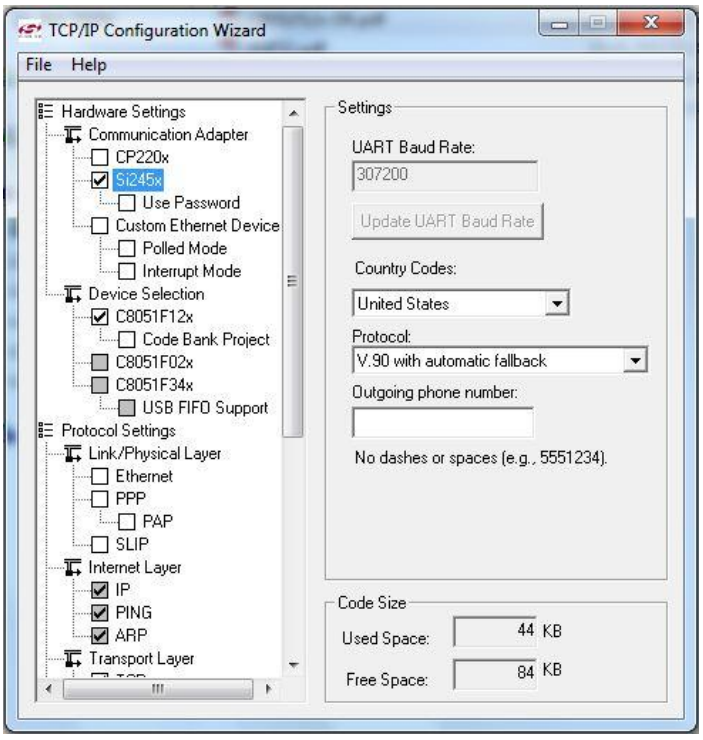


Şekil 3.9. Konfügürasyon sihirbazı yazılımı ekran görüntüsü

Seçilen entegreler içerisinde bulunan donanımları kontrol eden birçok kaydedici ve bayrak olmasından ve bunların kendi aralarında olan bağlantılarından dolayı, donanımsal ayarlamaları kolaylaştırmak amacıyla bu yazılım kullanılmaktadır. Configuration Wizard yazılımı sayesinde donanım ayarlamaları ilgili pencerelerden yapılır ve 8051 C kodları üretilir. Daha sonra ise üretilen bu kodlar yazılan programa kopyalanır.

3.2.3. TCP-IP configuration wizard yazılımı

Silicon Laboratories firmasının ağ entegreleri için geliştirdiği bu yazılım DHCP server, FTP server, HTTP server gibi ağ yönetim protokollerinin seçilen entegre ve özelliklerde 8051 kodlarının üretilmesini sağlar. Bu çalışmada da SI2457 entegresi ile kurulacak dial-up bağlantı işlemi bu programın oluşturduğu kodlar tarafından gerçekleştirilmiş ve ana program ile bu programın kontrolü sağlanmıştır.

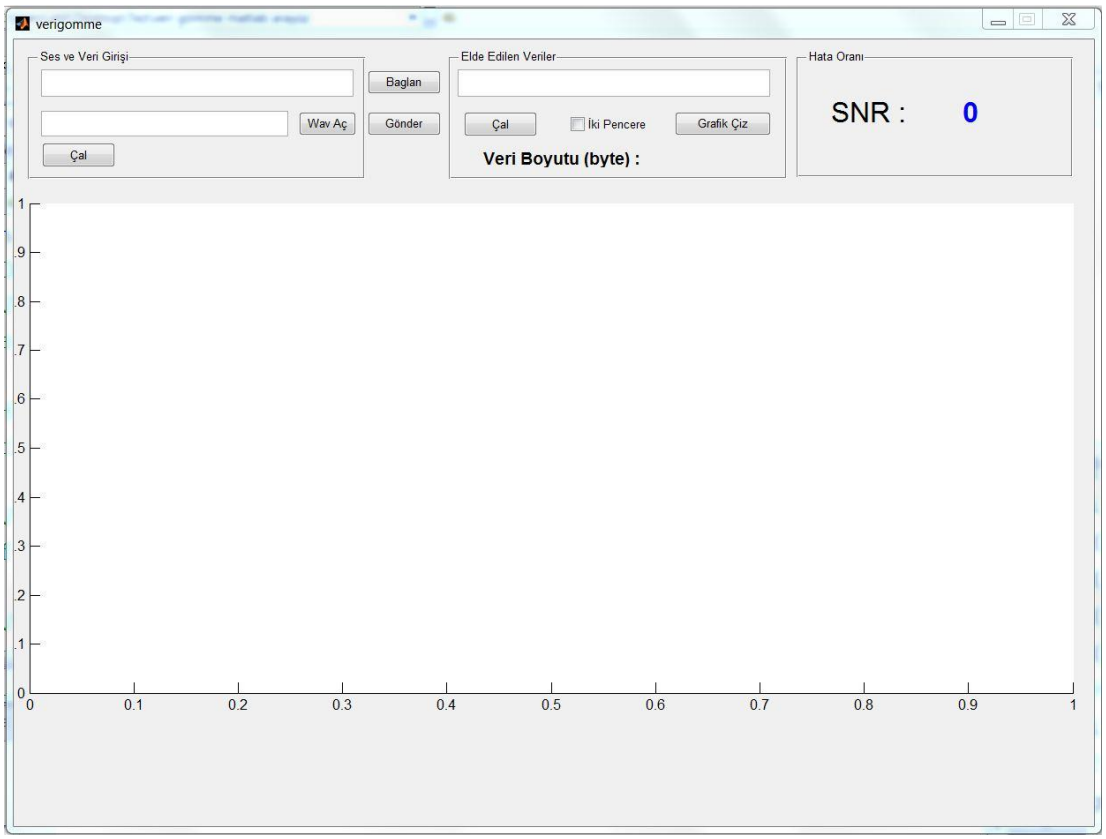


Şekil 3.10. TCP-IP Configuration Wizard yazılım ekran görüntüsü

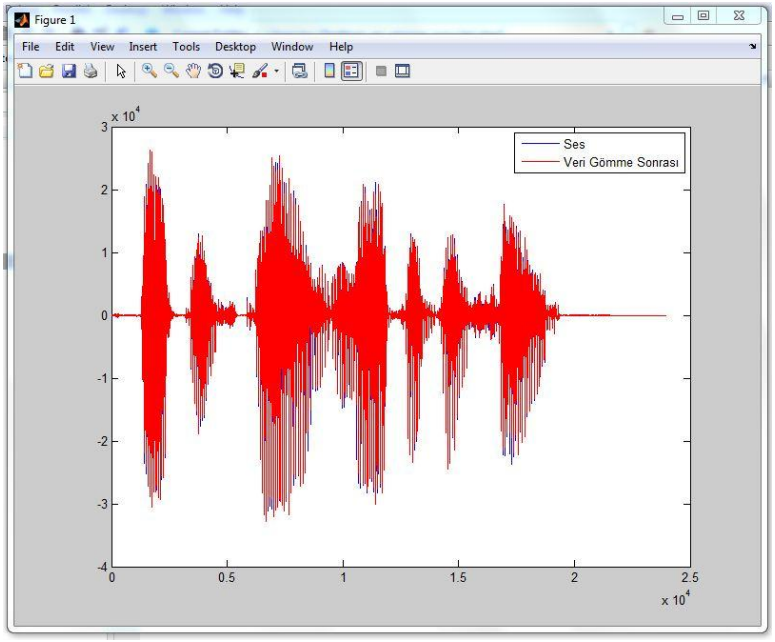
3.3. Sistemin Test Edilmesi İçin Geliştirilen Arayüz

Sistemin test edilmesi için geliştirilen ortam C8051F120 geliştirme kartı ve bilgisayardan oluşmaktadır. Matlab ortamında geliştirilen arayüz yardımı ile .vaw formatında seçilen bir ses dosyası ve bu ses dosyası üzerine gömülecek veri RS232 üzerinden geliştirme kartına gönderilmektedir. Geliştirme kartında ise alınan ses dosyası ADPCM kodlayıcı kullanılarak sıkıştırılmakta ve sonrasında sıkıştırılmış kodlar üzerinde veri gömme işlemi uygulanmaktadır. Yine geliştirme kartı üzerinde

sıkıştırılmış kodlardan veri çıkartımı yapılmakta ve sonrasında ses sinyalinin kodu çözülmektedir. Daha sonra RS232 üzerinden çıkartılan verinin uzunluğu ve kodu çözülmüş ses dosyası bilgisayara gönderilmektedir. Bilgisayar ortamında orijinal sinyal ile kodu çözülmüş sinyal karşılaştırılarak SNR değeri hesaplanmakta ve gösterilmektedir. SNR değeri sayesinde de ses kodlama ve veri gömme algoritmalarının sinyal üzerinde ne kadar bozulma meydana getirdiği incelenmektedir. Ayrıca her iki ses sinyalide arayüzde bulunan çal butonları sayesinde dinlenebilmektedir. Arayüzün altında bulunan alanda ise orijinal ses ile kodu çözülmüş ses sinyalinin zamana bağlı grafikleri çizilerek grafiksel olarak karşılaştırma yapılabilmektedir. Ayrıca arayüz sayesinde kaydedilen veriler Şekil 3.12 de gösteridiği gibi Matlab ortamında açılarak detaylı olarak incelenebilmektedir.



Şekil 3.11. Sistemin test edilmesi için geliştirilen arayüz



Şekil 3.12. Kaydedilmiş ses verilerinin Matlab ortamında incelenmesi

BÖLÜM 4. YAPILAN ÇALIŞMALAR VE SONUÇLAR

4.1. Matlab Arayüzü İle Veri Gömme Test İşlemi

Sistem gerçek zamanlı çalıştırılmadan önce, ses sinyali üzerinde veri gömme işleminden dolayı oluşacak bozulmaları ve uygun veri gömme miktarını belirlemek amacıyla bir test düzeneği kurulmuştur. Bu test düzeneği bir adet C8051F120DK geliştirme kartının RS232 portu aracılığıyla bilgisayara bağlanması ile oluşturulmuştur. Geliştirme kartı içerisinde ADPCM kodlayıcı ve kod çözücü, veri gömme ve veri çıkartma programları ile birlikte çalıştırılmaktadır. Matlab ortamında oluşturulmuş bir arayüz sayesinde önce veri gömme işleminde kullanılacak metin RS232 bağlantısı üzerinden geliştirme kartına aktarılır ve bir dizide tutulur. Ardından ses dosyasının ilk örneği olan 16 bitlik sayı geliştirme kartına aktarılır ve bu sayı ADPCM kodlayıcı tarafından kodlanır. Elde edilen ADPCM kodunun veri gömme işlemi için uygun olup olmadığı araştırılır ve eğer uygun ise 1 bitlik veri gömme işlemi gerçekleştirilir. Sonrasında da işlemler tersine yapılarak önce veri çıkartma işlemi yapılır ve çıkartılan veriler bir dizide biriktirilir. Son olarak da ADPCM kod çözücüye uygulanan bu kod tekrar ses sinyaline çevrilerek RS232 üzerinden bilgisayara gönderilir.

Ses dosyasının tamamı geliştirme kartında işlendikten sonra dizide saklanan ADPCM kodu içerisinden çıkartılan veriler ve gömülen veri miktarı yine bilgisayara aktarılır. Bilgisayardaki arayüzde giriş ses sinyali ve veri gömüldükten sonra elde edilen ses sinyallerinin grafikleri çizilir ve SNR oranı hesaplanarak hata oranı alanına yazılır. Yine çıkartılan veriler ve veri miktarı ilgili alanda gösterilir. Ayrıca her iki ses sinyali çal butonları yardımı ile dinlenerek bozulma miktarı kulaklada tespit edilebilir.

Veri gömme sonuçlarının değerlendirilmesi amacıyla kullanılan ses dosyaları Jerry D. Gibson ve arkadaşları tarafından yapılan PCM, DPCM ve ADPCM ses kodlayıcılarının performansının karşılaştırılması amacıyla yapılan çalışmada kullanılan ses dosyalarıdır [76]. Bu dosyalar .vaw formatında kaydedilmiştir. 16 bit çözünürlüğe sahip olan bu dosyaların örnekleme frekansı 8 KHz'dir. 24000 örnekten oluşan bu dosyalar 3 sn uzunluğundadır.

Veri gömme işlemi sırasında yapılan denemelerde mikrofondan alınan sinyaldeki gürültülerden dolayı veri gömme işlemi adım büyüklüğünün küçük değerlerinde mümkün olmamaktadır. Bu nedenle sistemin test edilmesi sırasında ses kayıt kalitesinden ve hepsinin sabit uzunlukta olmasından dolayı bu dosyalar kullanılmıştır. Bu dosyalar da aşağıdaki cümleler söylenmektedir.

s1.vaw dosyası: "The pipe began to rust while new." (kadın sesi)

s2.vaw dosyası: "Thieves who rob friends deserve jail." (erkek sesi)

s4.vaw dosyası: "Open the crate but don't break the glass." (erkek sesi)

s5.vaw dosyası: "Oak is strong and also gives shade." (erkek sesi)

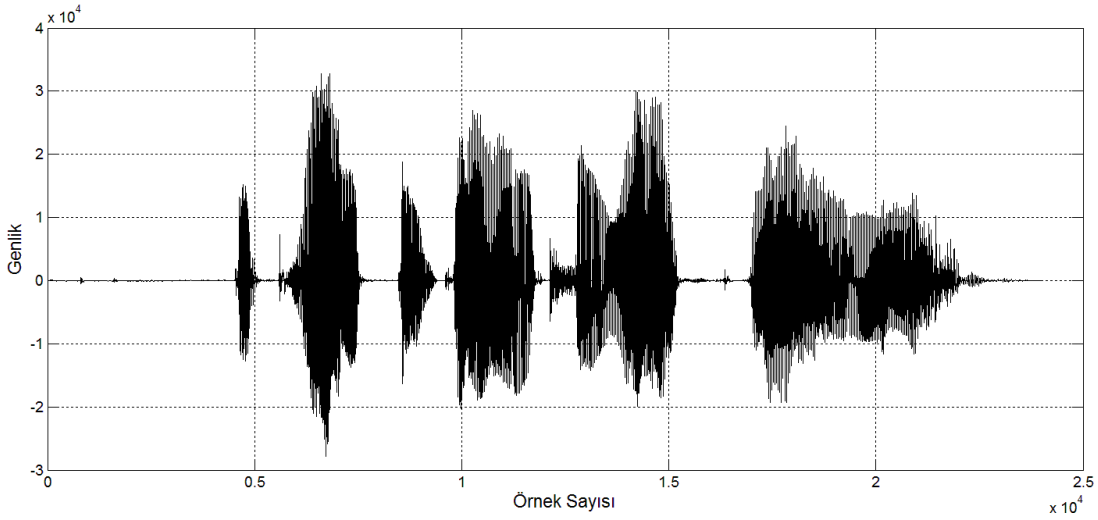
s6.vaw dosyası: "Cats and dogs each hate the other" (erkek sesi)

Bundan sonraki kısımda yukarıdaki ses dosyaları için veri gömme işlemi sonrasında elde edilen sonuçlar verilmiştir.

4.1.1. s1.vaw dosyası üzerinde yapılan incelemeler

Şekil 4.1'de s1.vaw dosyasına kaydedilmiş ses sinyali verilmiştir. Yatay eksen sinyalin örnek sayısını göstermektedir. 8 Khz ile örneklenen sinyal için örnek aralıkları 125μs'dir. 3 sn uzunluğundaki sinyal 24000 örnekten oluşmaktadır. Dikey

eksen ise her bir ses örneğinin genliğini göstermektedir. 16 bit ile örneklenen bu sinyalde genlik değeri -32768 ile +32767 aralığındadır.



Şekil 4.1. s1.vaw ses dosyası

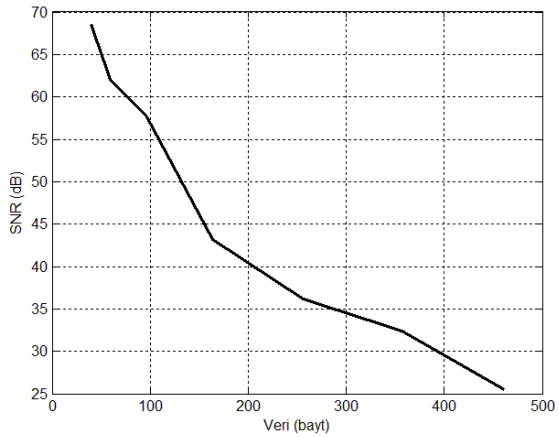
Tablo 4.1 veri gömme işlemi için kullanılan adım büyüklüğünü, veri gömme miktarını ve veri gömme işlemi sonrası sinyalde meydana gelen bozulmayı göstermektedir. Adım büyüklüğü ‘8’den küçük olarak seçildiğinde kayıpsız veri gömme yapılmakta ve sinyal üzerinde herhangi bir bozulma meydana gelmemektedir.

Tablo 4.1. s1.vaw dosyası için veri gömme işlemi sonrasında elde edilen sonuçlar

Adım büyüklüğü	Gömülen veri miktarı (byte)	SNR (dB)
8	34	Bozulma yok
10	40	68,5434
13	59	62,0479
16	97	57,6683
19	164	43,1478
25	256	36,2034
34	357	32,3691
50	461	25,4719

Bu tablodaki veriler kullanılarak Şekil 4.2’deki grafik elde edilir Tablodaki değerler ve elde edilen grafik incelendiğinde gömülen veri miktarı arttığında SNR oranı

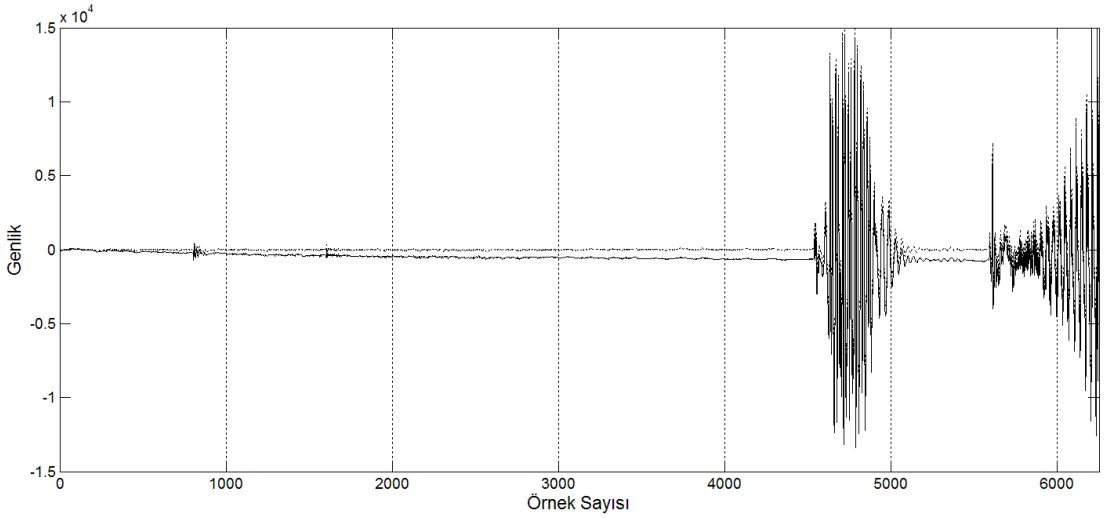
belirgin bir şekilde düşmektedir. Ama 25 db de bile ses sinyalindeki bozulma kulakla anlaşılmayacak seviyededir.



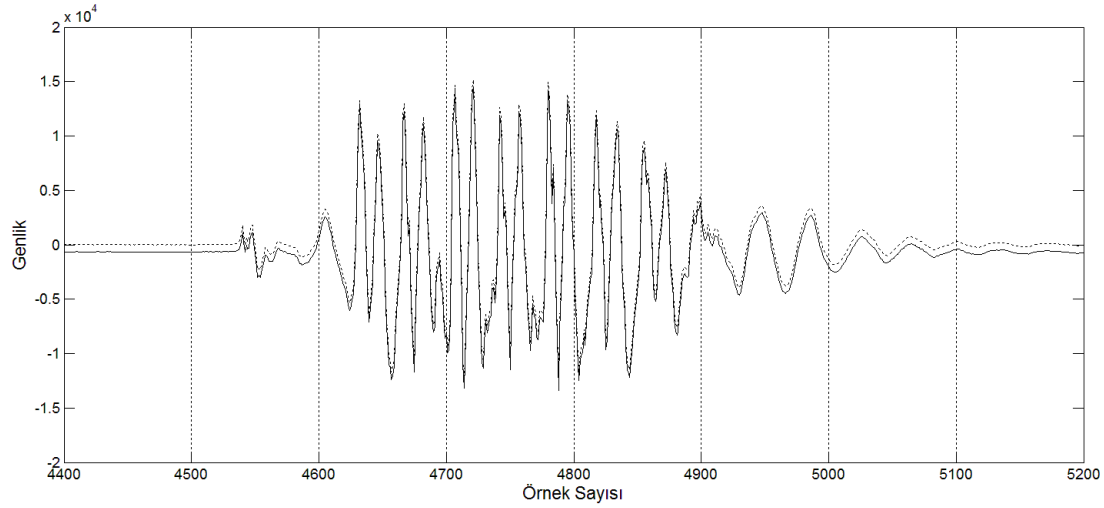
Şekil 4.2. s1.vaw dosyası için gömülen veri miktarına bağlı olarak sinyaldeki bozulma

Şekil 4.3 SNR oranının en düşük olduğu deneme için sinyal üzerinde meydana gelen bozulmanın en belirgin olduğu 0 ile 6250 aralığındaki örnekleri göstermektedir. Burada kesikli çizgi giriş sinyalini, düz çizgi ise veri gömüldükten sonraki sinyali göstermektedir. Bu aralık sinyaldeki değişimin en az olduğu dolayısıyla da veri gömme işleminin en yoğun olduğu aralıktır. Şekil incelendiğinde sinyalde meydana gelen bozulma şekil bozulması değil sinyalin üzerine bir ofset geriliminin eklenmesidir. Bu durum SNR oranını belirgin bir şekilde düşürse de insan kulağının bu tür değişimlere karşı hassas olmamasından dolayı sinyaldeki bozulma dinleyerek fark edilememektedir.

Şekil 4.4 incelendiğinde veri gömme sonucu sinyal üzerinde meydana gelen bozulma etkileri daha açık bir şekilde görülmektedir. Ses sinyalinin 4400 ile 5200 örnek aralığına bakıldığında sinyalde ki değişimlerin fazla ve az olduğu alanlar birlikte görülmektedir. Değişimin az olduğu anlarda veri gömme işleminden dolayı sinyalin genliğinde bir kayma gözlenmesine rağmen sinyal şekli ve frekansta herhangi bir bozulma olmamaktadır. Değişimin çok olduğu alanlarda ise zaten veri gömme işlemi uygulanmadığı için giriş sinyali aynen takip edilmektedir.



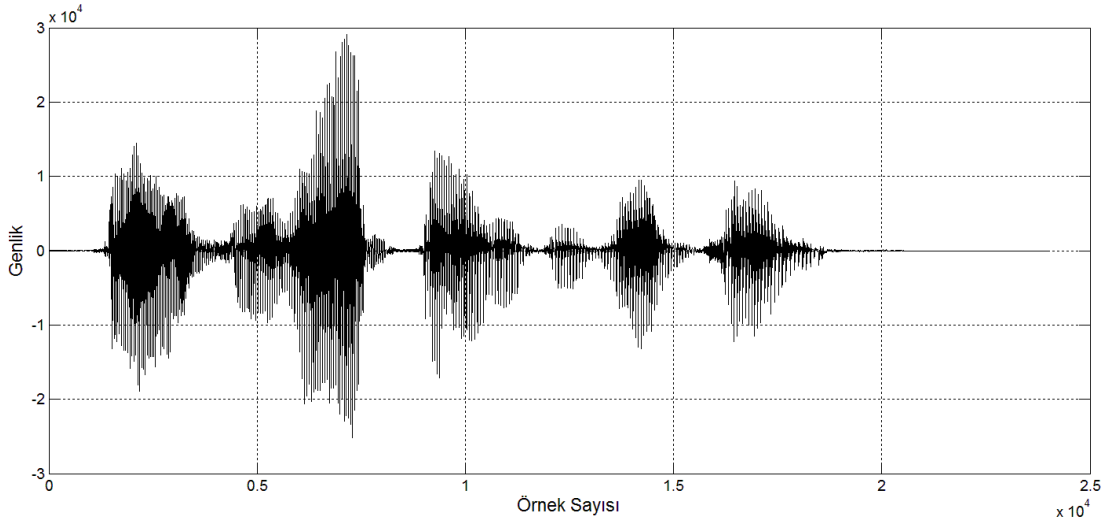
Şekil 4.3. s1.vaw dosyası üzerinde meydana gelen bozulma



Şekil 4.4. s1.vaw dosyası üzerinde meydana gelen bozulma

4.1.2. s2.vaw dosyası üzerinde yapılan incelemeler

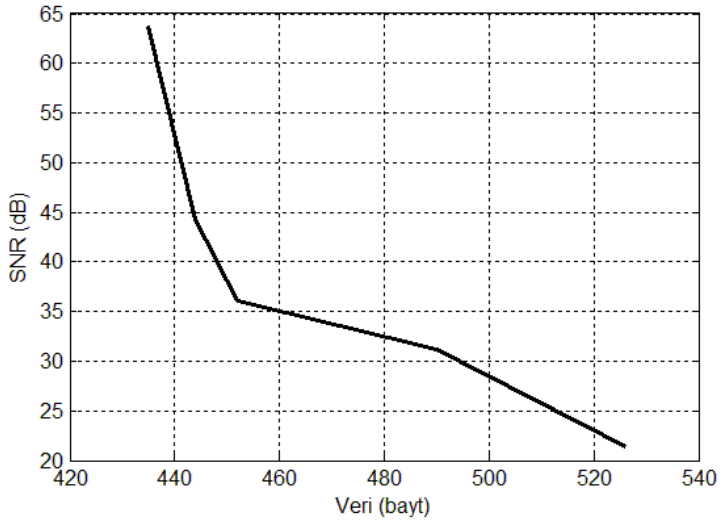
Şekil 4.5'te s2.vaw ses dosyasına ait ses sinyali verilmiştir. Tablo 4.2'de ki değerler s1.vaw dosyası için verilen değerler ile karşılaştırıldığında s2.vaw dosyasında düşük adım büyüklüğü değerlerinde daha çok miktarda veri gömülebildiği görülmüştür.



Şekil 4.5. s2.vaw ses dosyası

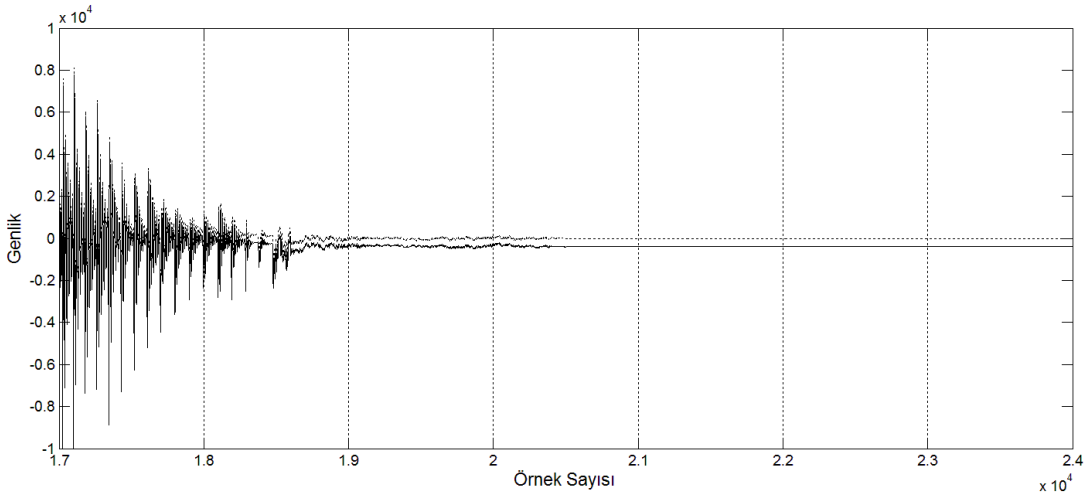
Tablo 4.2. s2.vaw dosyası için veri gömme işlemi sonrasında elde edilen sonuçlar

Adım büyüklüğü	Gömülen veri miktarı (byte)	SNR (dB)
8	435	Bozulma yok
9	435	63,7445
19	444	44,3220
25	452	36,0562
50	490	31,2285
100	526	21,4461

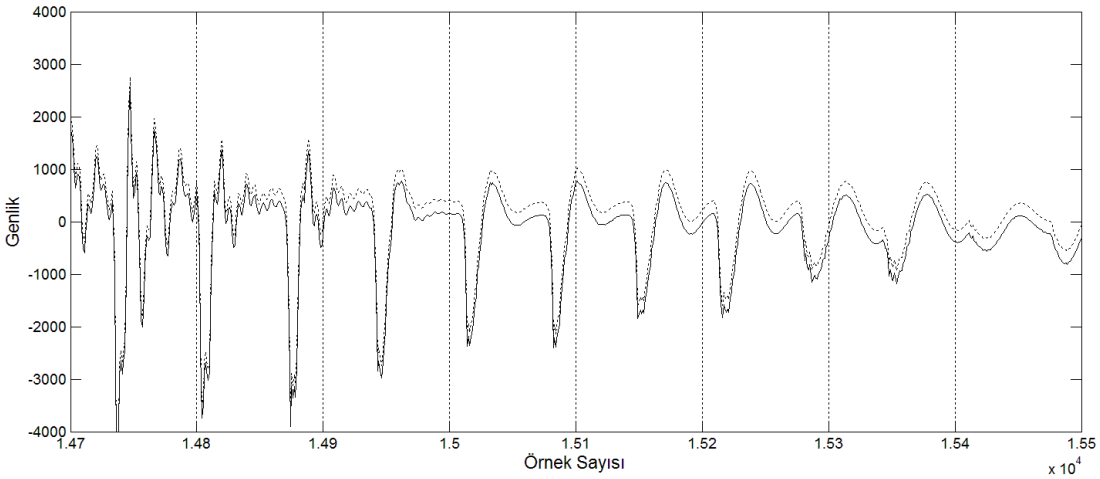


Şekil 4.6. s2.vaw dosyası için gömülen veri miktarına bağlı olarak sinyaldeki bozulma

Seçilen adım büyüklüğü değerleri kodlama işlemi sırasında oluşmamışsa veri gömme miktarı da doğal olarak azalacaktır. Bu adım büyüklüğü değerinin oluşması konuşan kişinin ses tonlarına şiddetine hatta cinsiyetine göre değişmektedir.



Şekil 4.7. s2.vaw dosyası üzerinde meydana gelen bozulma



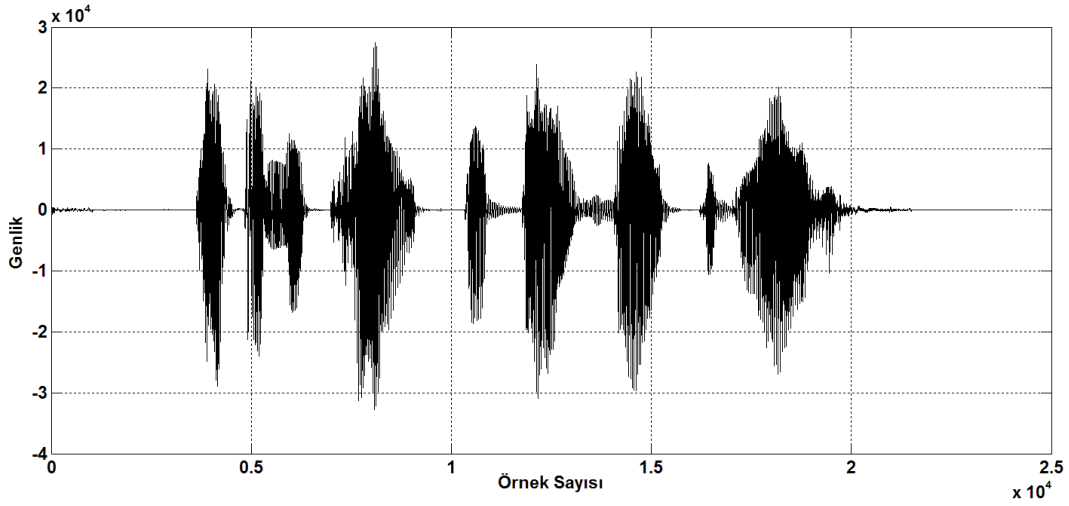
Şekil 4.8. s2.vaw dosyası üzerinde meydana gelen bozulma

Şekil 4.6'da görülen grafik, s2.vaw dosyası için Tablo 4.2'deki veriler kullanılarak çizilmiştir. Bu grafikte incelendiğinde gömülen veri miktarı arttığında ses sinyalindeki bozulmanın arttığı görülmektedir.Şekil 4.7 s2.vaw dosyası içerisindeki 17000 ile 24000 ve Şekil 4.8 s2.vaw dosyası içerisindeki 14700 ile 15500 arasındaki

örnekleri içermektedir. Burada kesikli çizgiler ile gösterilmiş giriş sinyali ile düz çizgilerle gösterilmiş veri gömüldükten sonraki sinyal arasındaki farklar görülmektedir.

4.1.3. s4.vaw dosyası üzerinde yapılan incelemeler

Şekil 4.9'da görülen s4.vaw dosyası görülmektedir. Bu ses sinyali içinde veri gömme işlemi uygulanıp benzer incelemeler yapılmıştır ve elde edilen sonuçlar Tablo 4.3'te verilmiştir.



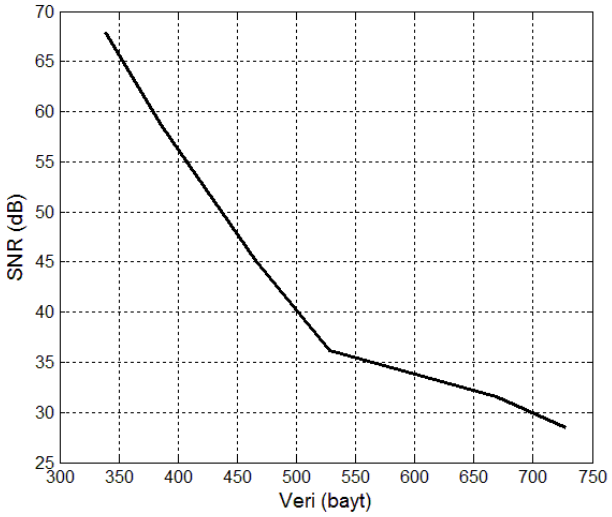
Şekil 4.9. s4.vaw ses dosyası

Tablo 4.3'teki veriler incelendiğinde 728 bayt veri gömülebildiği görülmektedir. Bu değer önceki ses sinyallerine gömülen veri miktarından fazla olmasına rağmen sinyaldeki 28,4576 dB bozulma diğer ses sinyalleri ile neredeyse aynıdır. Bunun nedeni de sinyalin başında ve sonunda bulunan uzun sessizlik anlarından kaynaklanmaktadır. Sessizlik arttıkça sinyaldeki değişim azalacak ve buna bağlı olarak veri gömme miktarı da fazla olacaktır.

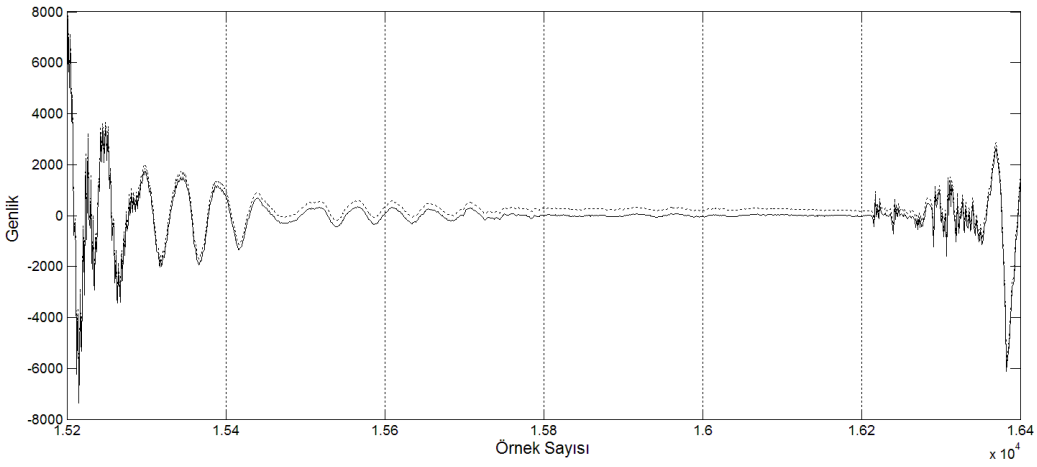
Şekil 4.10'da ki grafik Tablo 4.3'teki veriler kullanılarak çizilmiştir. Önceki incelemelerde de görüldüğü gibi bu ses sinyali içinde gömülen veri miktarının artması ile sinyaldeki bozulma artmış ve buna bağlı olarak SNR değeri azalmıştır.

Tablo 4.3. s4.vaw dosyası için veri gömme işlemi sonrasında elde edilen sonuçlar

Adım büyüklüğü	Gömülen veri miktarı (byte)	SNR (dB)
8	333	Bozulma yok
9	339	67,9560
14	387	58,5538
19	466	45,1739
25	528	36,1630
50	669	31,5704
100	728	28,4576

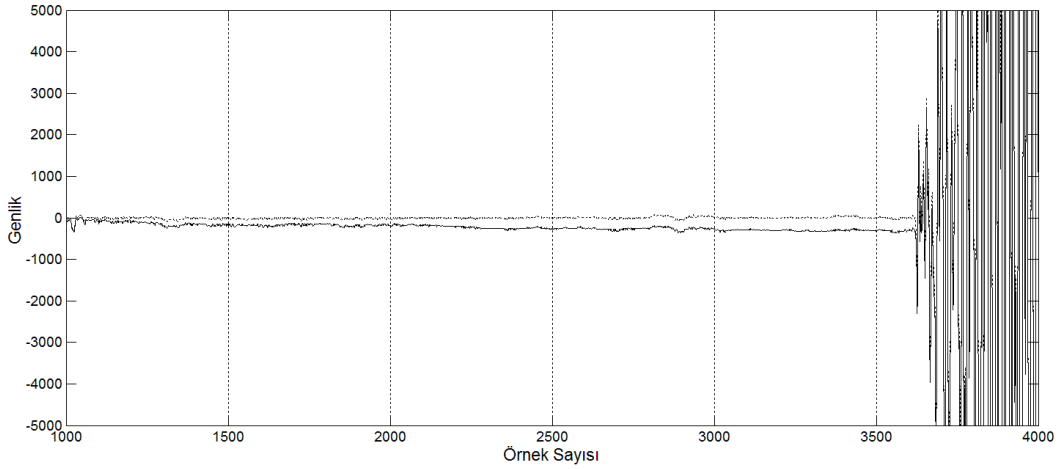


Şekil 4.10. s4.vaw dosyası için gömülen veri miktarına bağlı olarak sinyaldeki bozulma

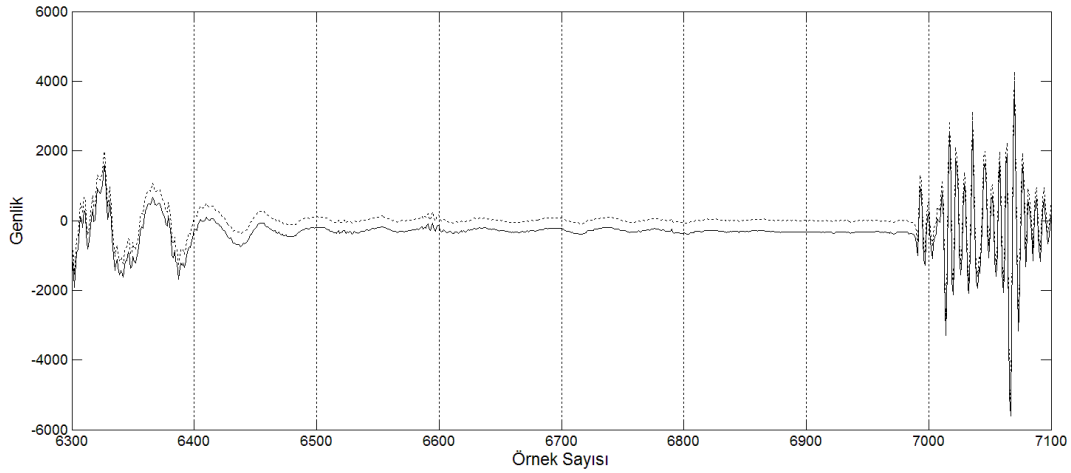


Şekil 4.11. s4.vaw dosyası üzerinde meydana gelen bozulma

Şekil 4.11, Şekil 4.12 ve Şekil 4.13 te s4.vaw dosyasındaki ses sinyali üzerinde veri gömme işlemi sonrasında en çok bozulma olan alanlar verilmiştir. Önceki ses sinyallerinde de olduğu gibi sinyalin ofset değerinde bir kayma görülmektedir. Fakat sinyalin genlik ve frekansı korunmaktadır.



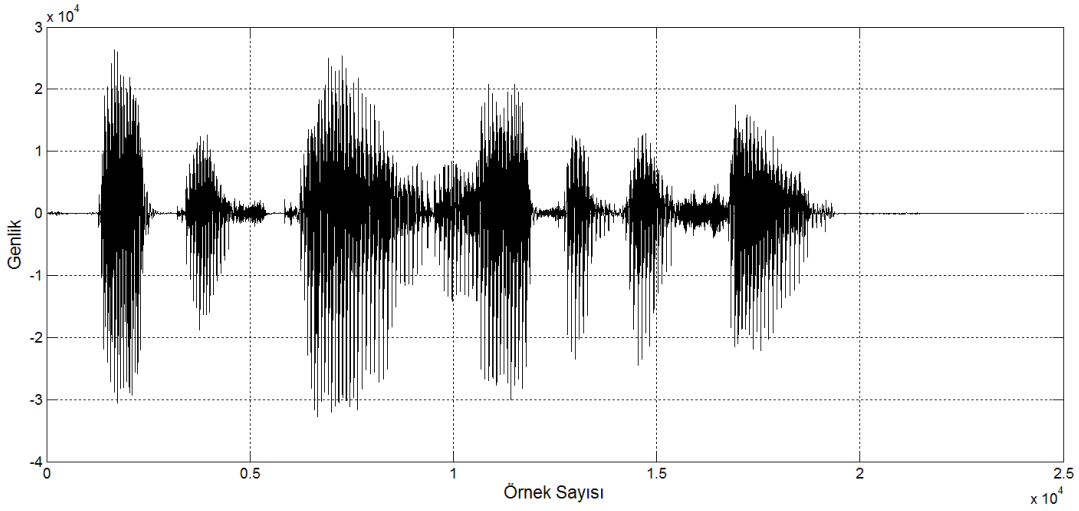
Şekil 4.12. s4.vaw dosyası üzerinde meydana gelen bozulma



Şekil 4.13. s4.vaw dosyası üzerinde meydana gelen bozulma

4.1.4. s5.vaw dosyası üzerinde yapılan incelemeler

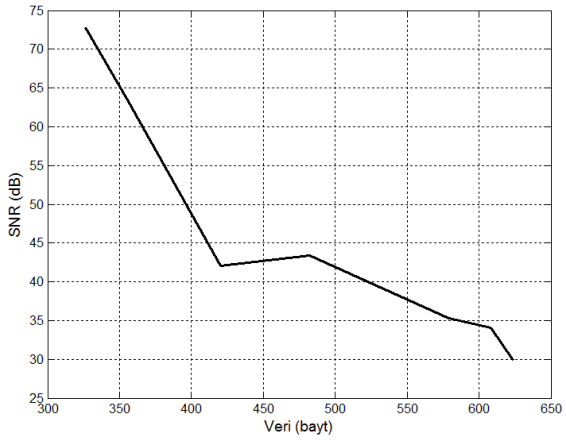
Şekil 4.14'te s5.vaw dosyası gösterilmiştir. Bu ses sinyalinin sonundaki uzun sessizlik alanı dikkat çekmektedir. Bu gömülen veri miktarının artmasına neden olacaktır.



Şekil 4.14. s5.vaw ses dosyası

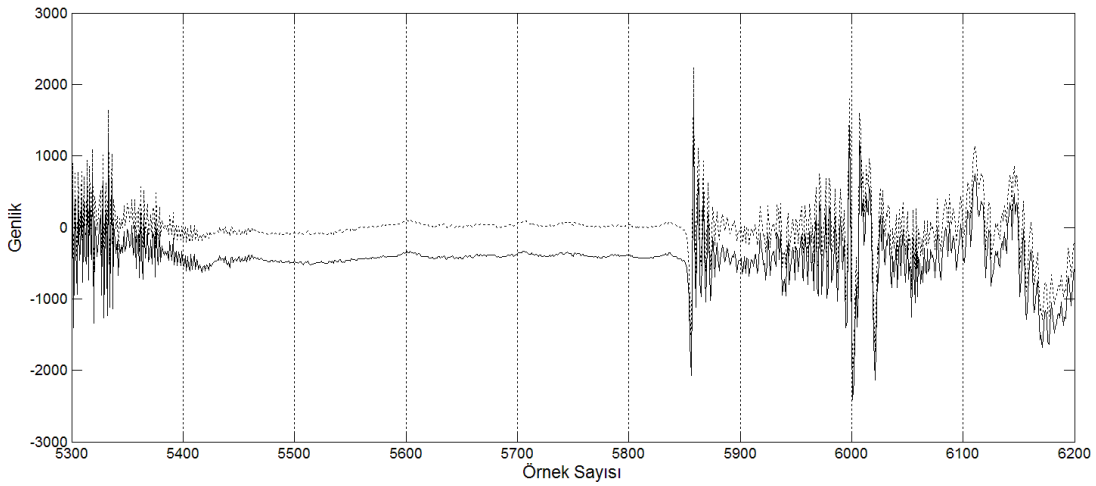
Tablo 4.4. s5.vaw dosyası için veri gömme işlemi sonrasında elde edilen sonuçlar

Adım büyüklüğü	Gömülen veri miktarı (byte)	SNR (dB)
8	324	Bozulma yok
9	326	72,7887
14	356	63,2879
19	420	42,0700
25	482	43,4146
50	579	35,3235
100	608	34,0199
200	624	29,8807

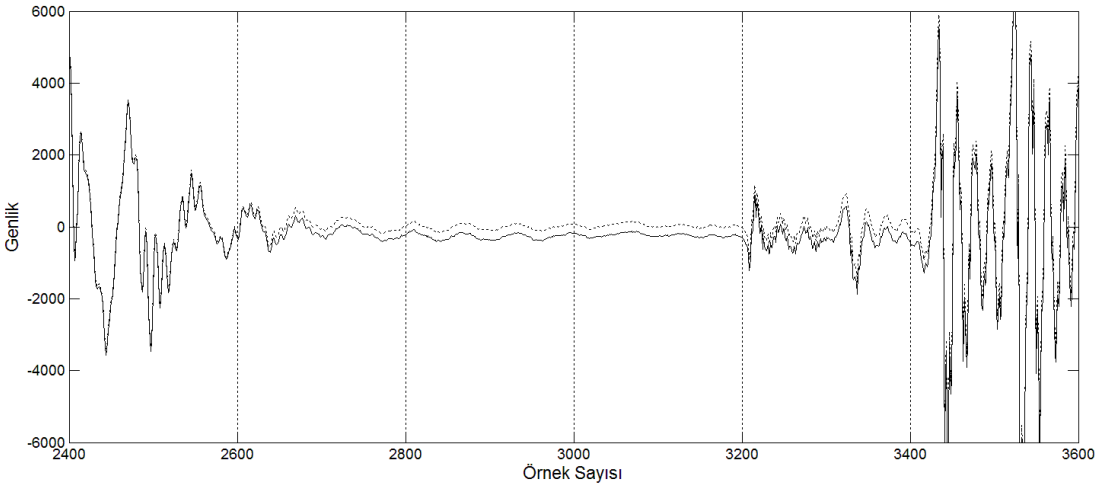


Şekil 4.15. s5.vaw dosyası için gömülen veri miktarına bağlı olarak sinyaldeki bozulma

Tablo 4.4'te elde edilen değerler kullanılarak Şekil 4.15'teki grafik elde edilmiştir. Şekil 4.16 ve Şekil 4.17'de s5.vaw dosyası üzerinde veri gömme işleminden sonra meydana gelen bozulmaların en fazla olduğu aralıklar verilmiştir.



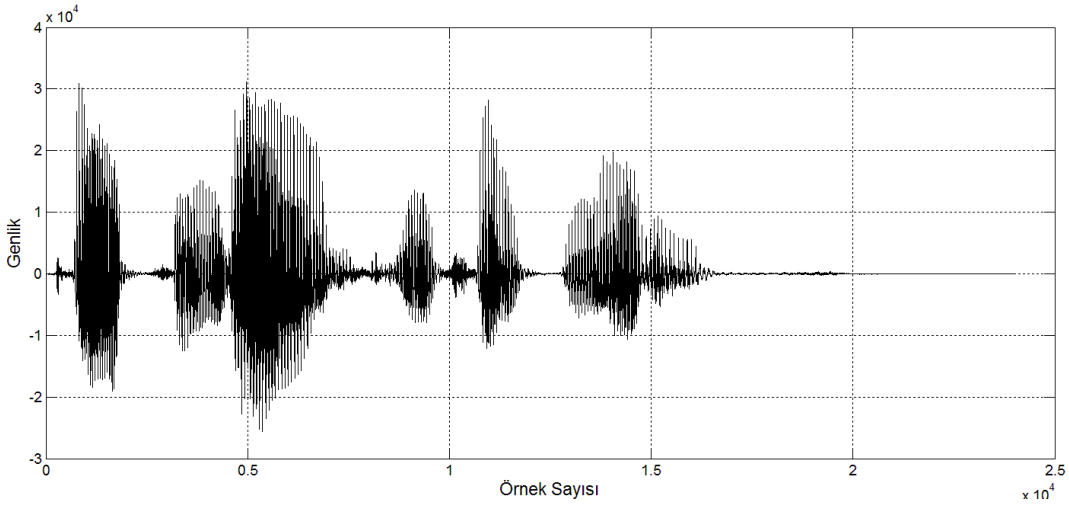
Şekil 4.16. s5.vaw dosyası üzerinde meydana gelen bozulma



Şekil 4.17. s5.vaw dosyası üzerinde meydana gelen bozulma

4.1.5. s6.vaw dosyası üzerinde yapılan incelemeler

Şekil 4.18'de verilen s6.vaw dosyası üzerinde de aynı işlemler tekrarlanmıştır. Bu dosyanın sonunda da uzun sessizlik anları bulunmaktadır.



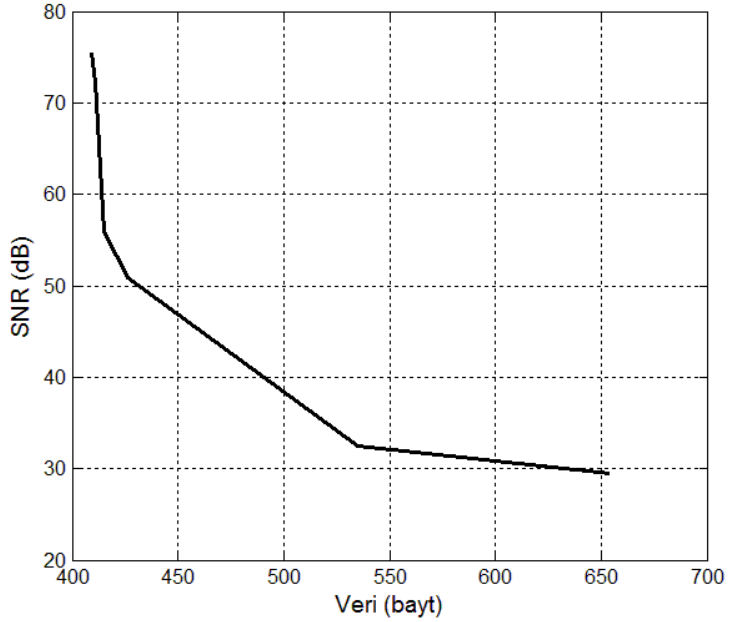
Şekil 4.18. s6.vaw ses dosyası

Tablo 4.5. s6.vaw dosyası için veri gömme işlemi sonrasında elde edilen sonuçlar

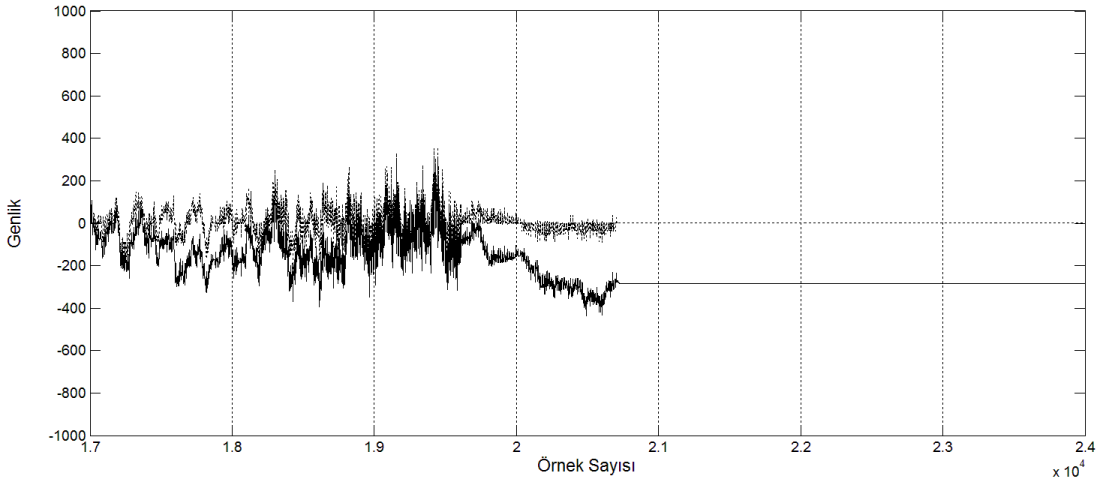
Adım büyüklüğü	Gömülen veri miktarı (byte)	SNR (dB)
8	409	Bozulma yok
9	409	Bozulma yok
10	410	75,3572
14	411	72,2631
19	415	55,9733
25	426	50,8282
50	535	32,4118
100	654	29,4710

Tablo 4.5 elde edilen değerler incelendiğinde ilk iki adımda da bozulma olmadığı görülmektedir. Bu durum adım büyüklüğü 9'dan küçük olarak seçildiğinde de aynı alanlara veri gömülmesinden kaynaklanmaktadır. Yani adım büyüklüğü hiçbir zaman 8 değerini almamıştır. Bu tabloda elde edilen değerler kullanılarak gömülen veri miktarına bağlı olarak sinyaldeki bozulmayı gösteren Şekil 4.19'daki grafik çizilmiştir.

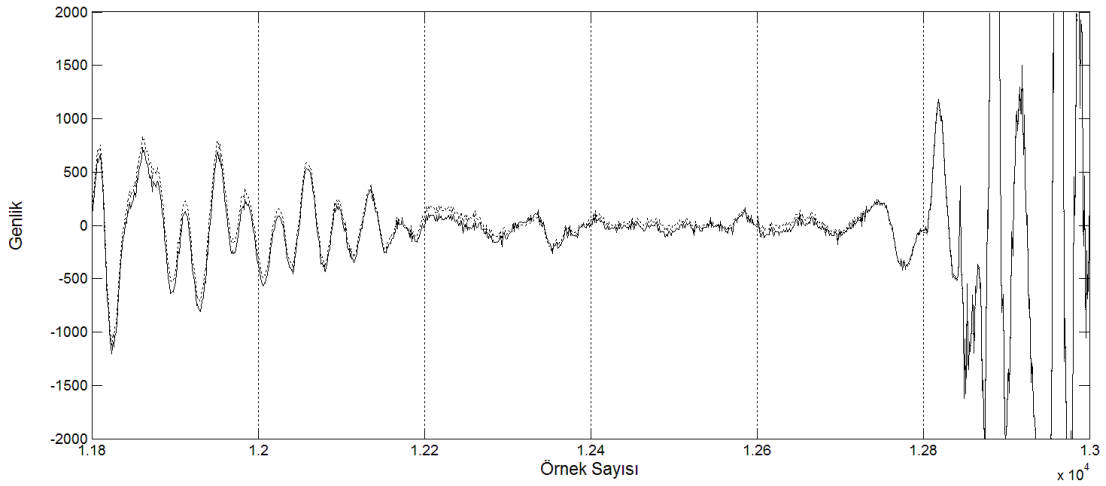
Şekil 4.20 ve Şekil 4.21'de s5.vaw dosyası üzerinde veri gömme işleminden sonra meydana gelen bozulmaların en fazla olduğu aralıklar verilmiştir.



Şekil 4.19. s6.vaw dosyası için gömülen veri miktarına bağlı olarak sinyaldeki bozulma



Şekil 4.20. s6.vaw dosyası üzerinde meydana gelen bozulma



Şekil 4.21. s6.vaw dosyası üzerinde meydana gelen bozulma

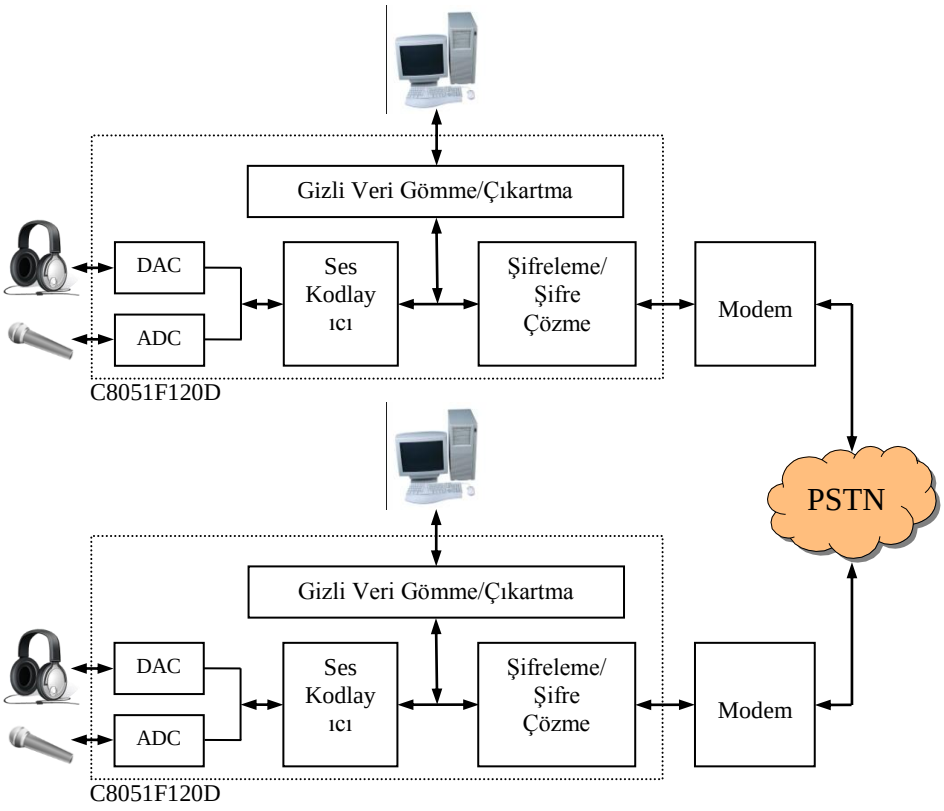
4.1.6. Veri gömme sonuçlarının değerlendirilmesi

Naofumi Aoki tarafından yapılan çalışmada ADPCM kodlarının üzerine kayıpsız veri gömme işlemi yapılmıştır ve buna bağlı olarak gömülen veri miktarı çok azdır. Hatta gürültülü ortamlarda yaptıkları denemelerde veri gömme işlemi gerçekleştirilememiştir. Tez çalışmasında ise kurulan sistemde kayıtlı ses dosyaları kullanılmadığından oluşan gürültülerden dolayı kayıpsız veri gömme işlemi uygulamak mümkün olmamıştır. Bu nedenle değişik parametreler ile yapılan veri gömme işlemleri ile orijinal sinyal üzerindeki bozulmalar incelenerek gerçeklenecek güvenli konuşma sistemi için uygun veri gömme parametrelerinin belirlenmesi sağlanmıştır.

4.2. Güvenli Konuşma Sistemi Üzerinden Gizli Metin Gönderilmesi

Gerçekleştirilen güvenli haberleşme sisteminin blok diyagramı Şekil 4.22'de verilmiştir. Burada iki güvenli haberleşme birimi PSTN ağının özelliklerini taşıyan bir telefon santrali üzerinden bir birine bağlanmıştır. Her iki güvenli haberleşme birimi RS232 bağlantısı üzerinden bilgisayara bağlanmıştır. Bilgisayar ortamında yazılan metinler Hyper Terminal programı ile güvenli haberleşme birimlerine gönderilebilmekte ve güvenli haberleşme birimlerinden gelen metinler de yine Hyper Terminal programı ile görülebilmektedir. Gizli metin gönderilmediği zamanlarda ses

sinyali önce kodlanarak ADPCM kodu elde edilir, daha sonra ise SEA şifreleme algoritması ile şifrelenerek PSTN üzerinden modem vasıtasıyla diğer güvenli haberleşme birimine gönderilir. Şifreleme algoritması 12 baytlık bloklar halinde veri şifrelediği için ADPCM kodları bir dizi içerisinde biriktirilerek toplu halde şifreleme ve gönderme işlemleri yapılır.

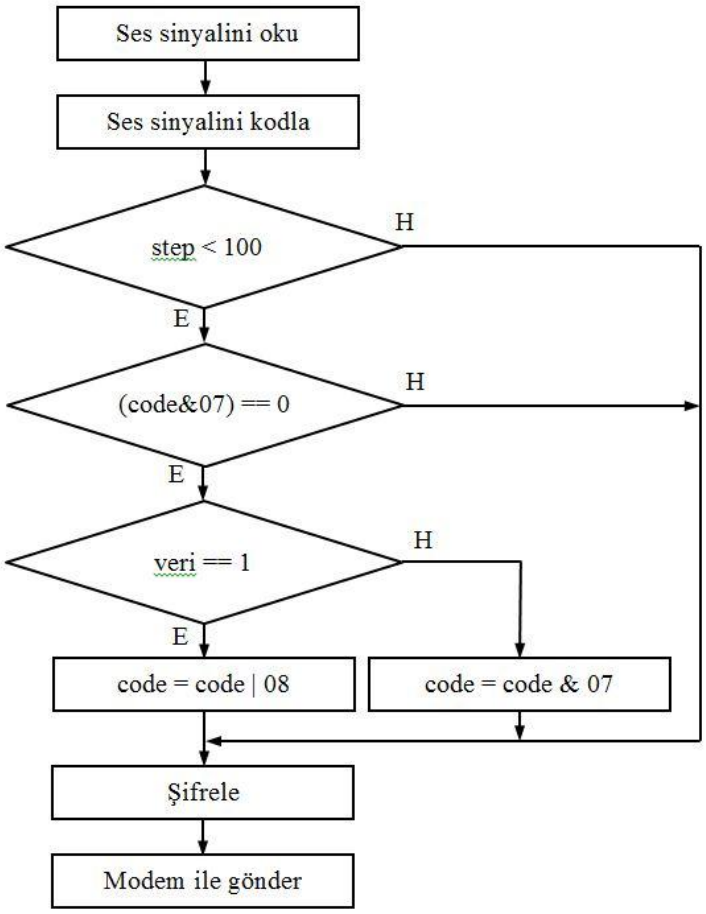


Şekil 4.22. Gizli metin gönderen güvenli konuşma sistemi

Gönderilen şifrelenmiş kodlar alındığında ise önce şifresi çözüldükten sonra kodu çözülerek ses sinyali tekrar elde edilir ve bir DAC yardımı ile hoparlöre aktarılır.

Eğer gönderilecek bir metin var ise ADPCM kodları üzerinde veri gömmeye uygun alanlar araştırılarak kodlar değiştirildikten sonra şifrelenerek gönderilir. Veri gömme işlemi Şekil 4.23'teki akış diyagramı ile gösterilmiştir. Kodlanmış sinyal üzerinde veri gömme için step değerinin 100'den küçük olduğu anlardaki kodlar seçilmiştir.

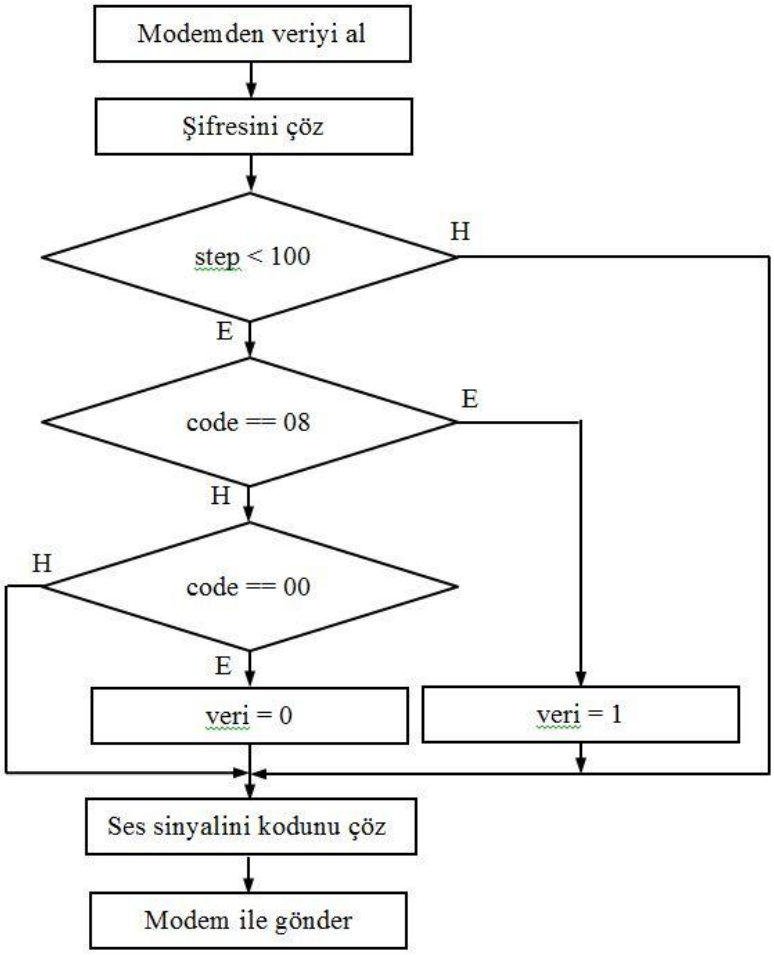
Step değeri 100'den küçük olduğu durumlarda kod '00' veya '08' ise veri gömme işlemi gerçekleştirilecektir. Bunun tespiti içinde kod değişkeni '07' ile 've' işlemine tabi tutulur ve eğer sonuç sıfır ise veri gömme işlemi uygulanır. Veri gömme işlemi sırasında tespit edilen kod değerinin üçüncü biti değiştirilerek veri gömme işlemi yapılacaktır. Eğer gömülmek istenen bit '1' ise kod değeri 08 sayısı ile 'veya' işlemine tabii tutularak üçüncü bitinin '1' olması sağlanacaktır. Eğer gömülmek istenen bit '0' ise kod değeri 07 sayısı ile 've' işlemine tabii tutularak üçüncü bitinin '0' olması sağlanacaktır. Daha sonrada elde edilen değiştirilmiş yeni kod değeri şifrelenerek gönderilecektir.



Şekil 4.23. Veri gömme algoritması

Veri çıkartım işleminde ise Şekil 4.24'de akış diyagramı verilen algoritma kullanılmıştır. Bu algoritmada veri gömülen kodları tespit etmek için yine step

değerinin 100'den küçük olduğu durumlar araştırılır. Step değeri 100'den küçük ise ve kod '00' veya '08' değerini almışsa veri gömüldüğü anlamına gelecektir. Bu durumda kod değişkenin üçüncü biti kontrol edilerek kod '00' ise gömülen bit '0', kod '08' ise gömülen bit '1' olarak kabul edilecektir. Sonrasında elde edilen karakterler RS232 üzerinde Hyper Terminal programına gönderilerek bilgisayar ekranında gösterilecektir.

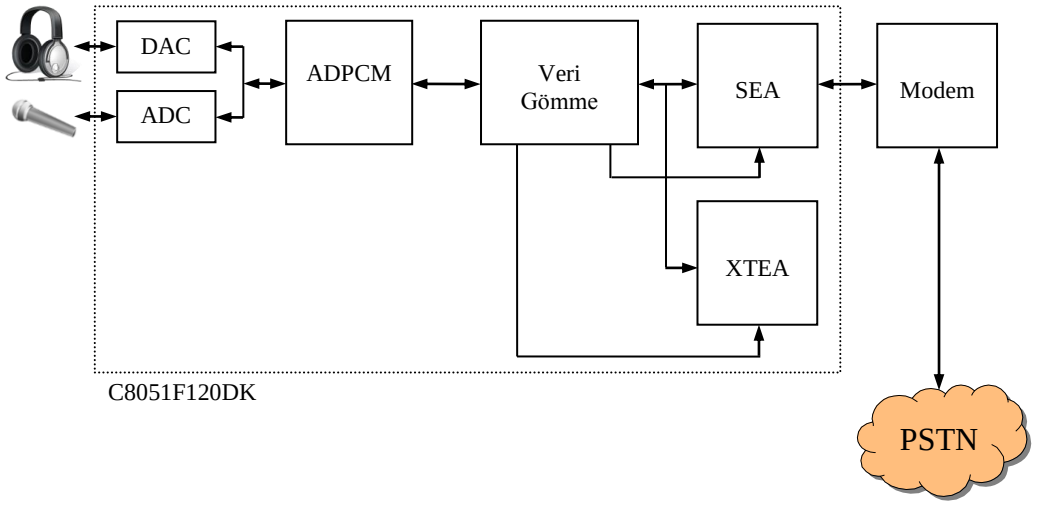


Şekil 4.24. Veri çıkartım algoritması

Yapılan bu uygulama sayesinde PSTN üzerinden güvenli haberleşme sağlanmıştır. Hattın dinlenmesi ve şifrenin çözülmesi ihtimaline karşın ADPCM kodları üzerinden gizli metin gönderilerek güvenlik arttırılmıştır.

4.3. Gönderilen Gizli Metin ile Şifreleme Algoritmalarının Değiştirilmesi

Yapılan ikinci uygulamanın blok diyagramı Şekil 4.25'te verilmiştir. Bu uygulamada amaçlanan yine güvenliğin arttırılması amacıyla, kodlanmış ses üzerinden gönderilen gizli veriyle belli aralıklarla şifreleme algoritmasının ve kullanılan anahtarın değiştirilmesidir. Sabit bir şifreleme algoritması ile çalışılırken anahtarın tespiti ile bundan sonraki bütün konuşma dinlenebilir. Tekrar güvenliğin sağlanması için gizli haberleşmeyi yapan iki kişinin bir başkasının haberi olmadan ortak bir anahtar belirlemesi gerekmektedir.



Şekil 4.25. Gizli Metin ile Şifreleme Algoritmalarının Değiştirilmesi

Bu uygulamada ise SEA ve XTA şifreleme algoritmaları kısa zaman aralıkları ile değiştirilmiştir. Bu algoritmaların değiştirilmesi için gerekli bilgide yine konuşma sinyali üzerinden gizli olarak gönderilmiştir. Her iki güvenli haberleşme biriminde bulunan bir tablo yardımı ile de gönderilen gizli bilgiye bağlı olarak her iki birimde aynı algoritma ve aynı anahtar kullanılması sağlanmıştır. Bu durumda hat dinlenildiğinde bütün anahtarlar elde edilmiş bile olsa gönderilen gizli bilgiye ulaşılmadan hangi aralıkta hangi şifreleme algoritması ve hangi anahtar kullanıldığının tespiti yapılamayacaktır. Güvenliğin daha da artması için kullanılan şifreleme algoritmalarının sayısı da arttırılabilir.

Burada dikkat edilmesi gereken nokta öncelikle yeni şifreleme algoritması ve anahtar değerini içeren bilgi gömüldükten sonra şifreleme algoritmasının değiştirilmesidir. Karşı tarafta da veri çıkartıldıktan sonra şifreleme algoritması ve anahtar değiştirilecektir.

Kullanılan yöntem sayesinde AES gibi çok güçlü şifreleme algoritmaları kullanılmadan, SEA ve XTEA gibi düşük işlem gücüne sahip mikroişlemciler için geliştirilmiş algoritmalar kullanılarak daha yüksek güvenliğe sahip sistemler geliştirilebilir.

BÖLÜM 5. GENEL DEĞERLENDİRME

Tez çalışmasında kablolu iletişim hatlarında kullanılmak için güvenli haberleşme amaçlı bir sistem tasarlanmış ve gerçeklenmiştir. Sistemin gerçeklenmesi amacıyla düşük işlem gücüne sahip bir mikrodenetleyici seçilmiştir. Bu seçimlere bağlı olarak ta düşük işlem gücüne sahip ve düşük bit akış hızına sahip olan ADPCM ses kodlama algoritması kullanılarak ses sinyalinin sıkıştırılması ve sonrasında sıkıştırılmış kod çözülerek konuşma sinyalinin tekrar elde edilmesi sağlanmıştır. Yine bu seçimler nedeniyle, şifreleme algoritması olarak gömülü sistemler için geliştirilmiş, düşük işlem gücüne sahip SEA ve XTEA şifreleme algoritmaları kullanılmıştır.

Yapılan çalışmada öncelikle ADPCM algoritması kullanılarak ses kodlama ve kod çözme işlemi gerçekleştirilmiştir. Daha sonra ADPCM kodları üzerine veri gömme işlemi gerçekleştirilmiş ve veri gömme işleminin sinyal üzerinde meydana getirdiği bozulmalar incelenmiştir. Bu incelemeler sonucu elde edilen değerler, sistemin çalıştırılması sırasında veri gömme işleminde kullanılacak parametreleri belirlemek için kullanılmıştır.

Gerçeklenen bu sistem sayesinde, dijital hale getirilmiş konuşma bilgisi şifrelenerek PSTN hattı üzerinden iletilebilmekte ve böylece görüşmenin başkaları tarafından dinlemmesi önlenerek güvenli bir görüşme yapılabilmektedir. Şifreli olarak gönderilen bu konuşma sinyali üzerine gizli bir metin gömülmesi ile de güvenlik seviyesi arttırılmıştır. Görüşme sırasında gönderilen sinyalin şifresi saldırganlar tarafından çözülmüş olsa da ses sinyalleri üzerindeki gizli metin ile güvenli haberleşmeye devam edilebilir. Ses sinyalleri üzerine gizli bir metnin gömüldüğü

fark edilse bile bu metnin elde edilebilmesi için veri gömme parametrelerinin ve veri gömme algoritmasının da çözülmesi gerekmektedir.

Burada, konuşma bilgisi aldatıcı bilgi olarak kullanılarak asıl gönderilmek istenen bilgi gizli metin ile gönderilebilir. Bu şekilde de güvenlik daha da arttırılmış olmaktadır.

Yapılan ikinci bir uygulama ile de konuşma sinyali üzerine gömülen gizli metin, kullanılan şifreleme algoritmasının şifreleme-şifre çözme işlemleri sırasında kullandığı anahtarı ve kullanılan şifreleme algoritmasını değiştirmek amacıyla kullanılmıştır. Bu sayede kullanılan düşük güçlü şifreleme algoritmalarının kırılma ihtimali daha da azaltılarak, yüksek hızlı bir mikro işlemciye gerek kalmadan yüksek güvenlik elde edilmiş olmaktadır. Bu durumda saldırganın şifreleme işlemini çözebilmesi için sadece şifreleme algoritmasını ve şifreleme işlemi sırasında kullanılan anahtarı elde etmesi yetmez. Hangi anahtarın hangi anda kullanıldığını ve kullanılan anahtarın hangi şifreleme algoritmasına ait olduğunu da sürekli takip etmesi gerekecektir. Bu da şifreleme işleminin çözülme olasılığını azaltarak sistemin güvenliğini arttıracaktır.

Ayrıca sabit bir anahtarla çalışan sistemlerde anahtarın elde edilmesi ile konuşmanın dinlenmesi söz konusu olacaktır. Bu durumda tekrar gizli görüşme yapılabilmesi için her iki tarafın yeniden bir anahtar belirleyerek, bu anahtarı başka birinin eline geçmeden birbirlerine ulaştırabilmesi gerekmektedir. Konuşma sinyali üzerine gömülen gizli metin bu amaçla da kullanılarak, şifrelemede kullanılacak yeni yöntemin ve anahtarlarında saldırganlar tarafından fark edilmeden hızlı bir şekilde paylaşılmasına olanak sağlayacaktır.

Sonuç olarak, yapılan tez çalışmasında kullanılan güvenli haberleşme uygulamaları ele alınarak bu uygulamalardaki güvenliğin arttırılması sağlanmıştır. Geliştirilen sistem üzerinde bazı iyileştirmeler de yapmak mümkündür. Tasarlanan sistem belirli bir gürültünün olduğu ortamda test edilmiştir. Ortamdaki gürültü miktarı arttıkça

sinyal üzerindeki değişimlerin hızı arttığından veri gömülebilecek alanlar da azalmaktadır ve belki de aşırı gürültülü ortamlarda veri gömme işlemini yapabilmek için kullanılan veri gömme parametrelerinin arttırılması gerekecektir. Bu parametreler sabit olarak kullanıldığında da düşük gürültülü ortamlarda sinyaldeki bozulmalar artacak ve sinyalin değiştirildiğinin fark edilmesi kolaylaşacaktır. Bunu önlemek için sistemin o anki gürültü durumuna bağlı bir adaptif yapı kullanılarak uygun veri gömme parametrelerini belirleyen ve sürekli güncelleyen bir program sisteme ilave edilebilir.

Tasarlanan sistemde, dijital bilginin iletilmesi sırasındaki veri kayıpları düşünülmemiştir. Buradaki veri kayıpları dialup modemin kullandığı veri iletim protokolleri tarafından önlenmektedir. Ama bağlantı kopması gibi durumlar göz önünde bulundurularak ilave senkronizasyon programları yazılıp sisteme dahil edilebilir.

Yapılan ikinci uygulamada güvenliği arttırma amaçlı iki adet şifreleme algoritması kullanılarak bu algoritmaların gizli bilgi ile değiştirilmesi sağlanmıştır. Güvenliği arttırmak için şifreleme algoritmalarının sayısı arttırılabilir. Böylece şifreleme işleminin çözülebilmesi ihtimali daha da azaltılmış olacaktır.

KAYNAKLAR

[1] M. K. SUNDARESHAN, R. RAMASWAMY, "Desıgn And Deployment Of An Integrated Data Cıpherıng Unıt Insıde A Low Bıt Rate Voıce Transcoder For Secure Voıce Communıcatıons Over Telephone Networks," IEEE International Conference on "World Prosperity Through Communications", ICC '89, BOSTONICC/89., vol. 3, pp. 1149–1153, 1989.

[2] K. G. GOPALAN, D. S. BENINCASA, S. J. WENNDT, "Data embedding in audio signals," 2001 IEEE Aerospace Conference Proceedings (Cat. No.01TH8542), vol. 6, pp. 2713–2720.

[3] H. MALIK, A. KHOKHAR, R. ANSARI, "ROBUST DATA-HIDING IN AUDIO," 2004 IEEE International Conference on Multimedia and Expo (ICME), pp. 959–962, 2004.

[4] C.-C. CHANG, R. C.-T. LEE, G.-X. XIAO, T.-S. CHEN, "A new Speech Hiding Scheme based upon sub-band coding," Proceedings of the 2003 Joint Conference of the Fourth International Conference on Information, Communications and Signal Processing, 2003 and Fourth Pacific Rim Conference on Multimedia., vol. 2, pp. 980–984, 2003.

[5] Y. CHEN, T. LI, D. GAO, X. HU, X. ZHANG, J. LIU, "A secure mobile communication approach based on information hiding," IEE Mobility Conference 2005. The Second International Conference on Mobile Technology, Applications and Systems, vol. 2005, no. 1, pp. 129–129, 2005.

[6] J. D. GIBSON, M. G. KOKES, "DATA EMBEDDING FOR SECURE COMMUNICATIONS," MILCOM 2002. Proceedings, vol. 1, pp. 406–410.

[7] N. LAZIC, P. AARABI, "Communication Over an Acoustic Channel Using Data Hiding Techniques," IEEE Transactıons On Multımedıa, vol. 8, no. 5, pp. 918–924, 2006.

[8] S. CHEN, H. LEUNG, H. DING, "Telephony Speech Enhancement by Data Hiding," IEEE Transactıons On Instrumentatıon And Measurement, vol. 56, no. 1, pp. 63–74, 2007.

[9] M. W. FAKHR, “A Novel Data Hiding Technique for Speech Signals with High Robustness,” 2007 IEEE International Symposium on Signal Processing and Information Technology, pp. 379–384, 2007.

[10] G. TROULLINOS, “A SOFTWARE BASED APPROACH TO SECURE VOICE APPLICATIONS,” Proceedings of the Third IEEE International Conference on Electronics, Circuits, and Systems, ICECS ’96.,, vol. 1, pp. 176–182.

[11] FEIZI-KHANKANDISOHEIL, F. MARVASTI, M. A. AKHAEE, “Two Techniques for Audio Watermarking Based on a Novel Transformation,” 2007 IEEE International Conference on Signal Processing and Communications (ICSPC 2007),, no. November, pp. 1139–1142, 2007.

[12] E. JAHANGIRI, S. GHAEMMAGHAMI, “Hıgh Rate Data Hıdıng In Speech Usıng Voıcıng Dıversıty In An Adaptıve Mbe Scheme,” TENCON 2008 - 2008 IEEE Region 10 Conference, pp. 1–6.

[13] F. A. P. PETITCOLAS, R. J. ANDERSON, M. G. KUHN, “Information Hiding — A Survey,” Proceedings of the IEEE, vol. 87, no. 7, pp. 1062–1078, 1999.

[14] D. E. SKOPIN, I. M. M. EL-EMARY, R. J. RASRAS, R. S. DIAB, “Advanced algorithms in audio steganography for hiding human speech signal,” 2010 2nd International Conference on Advanced Computer Control, pp. 29–32, 2010.

[15] J. W. SEOK, J. W. HONG, “Audio watermarking for copyright protection of digital audio data,” Electronics Letters, vol. 37, no. 1, p. 60, 2001.

[16] B. GEISER, V. PETER, “Hıgh Rate Data Hıdıng In Acelp Speech Codecs,” IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2008., pp. 4005–4008, 2008.

[17] T. RABIE, D. GUERCHI, “Magnıtude Spectrum Speech Hıdıng,” 2007 IEEE International Conference on Signal Processing and Communications (ICSPC 2007),, pp. 1147–1150, 2007.

[18] H. M. DIPU KABIR, S. B. ALAM, “Hardware based realtime, fast and highly secured speech communication using FPGA,” 2010 IEEE International Conference on Information Theory and Information Security, pp. 452–457, Dec. 2010.

[19] P. PRANDONI, M. VETTERLI, “Perceptually hidden data transmission over audio signals,” Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, 1998., vol. 6, pp. 3665–3668, 1998.

[20] J. S. COLLURA, D. J. RAHIKKA, “Interoperable Secure Voıce Communıcatıons In Tactıcal Systems,” Speech Coding for Algorithms for Radio Channels (Ref. No. 2000/012), IEE Seminar, pp. 7/1–7/13, 2000.

[21] J. D. GIBSON, A. SERVETTI, H. DONG, A. GERSHO, J. C. D. MARTIN, “SELECTIVE ENCRYPTION AND SCALABLE SPEECH CODING FOR VOICE COMMUNICATIONS OVER MULTI-HOP WIRELESS LINKS,” 2004 IEEE Military Communications Conference MILCOM 2004, pp. 792–798, 2004.

[22] M. ASHTIANI, S. ASADI, P. H. GOUDARZI, “A New Method in Transmitting Encrypted Data by FCM Algorithm,” 2006 2nd International Conference on Information & Communication Technologies, vol. 1, pp. 1046–1051, 2006.

[23] D. LIXIN, “A new approach of data hiding within speech based on Hash and Hilbert Transform,” ICSNC ’06. International Conference on Systems and Networks Communications, 2006., pp. 6–9.

[24] T. CHMAYSSANI, G. BAUDOIN, G. HENDRYCKX, “SECURE COMMUNICATIONS THROUGH SPEECH DEDICATED CHANNELS USING DIGITAL MODULATIONS,” 42nd Annual IEEE International Carnahan Conference on Security Technology, 2008. ICCST 2008., pp. 312–317, 2008.

[25] K. GOPALAN, Q. SHI, “Audio Steganography Using Bit Modification - A Tradeoff on Perceptibility and Data Robustness for Large Payload Audio Embedding,” 2010 Proceedings of 19th International Conference on Computer Communications and Networks, pp. 1–6, Aug. 2010.

[26] S. L. TIMOTHY, “Implementation of a Real-Time HY-2 Channel Vocoder Algorithm,” MILCOM 97 Proceedings, vol. 1, pp. 525–529, 1997.

[27] M. A. OZKAN, B. ORS, G. SALDAMLI, “Secure Voice Communication via GSM Network,” in 7th International Conference onElectrical and Electronics Engineering (ELECO), 2011, 2011, p. II–288 – II–292.

[28] M. WASIF, C. R. SANGHAVI, M. ELAHI, “Secure Mobile Communication Using Low Bit-rate Coding Method,” in International Conference on Wireless Communications, Networking and Mobile Computing, 2007. WiCom 2007., 2007, pp. 1410–1413.

[29] S. ISLAM, F. AJMAL, S. ALI, J. ZAHID, A. RASHDI, “Secure end-to-end communication over GSM and PSTN networks,” 2009 IEEE International Conference on Electro/Information Technology, pp. 323–326, Jun. 2009.

[30] N. N. KATUGAMPALA, K. T. AL-NAIMI, S. VILLETTE, A. M. KONDOZ, “REAL TIME DATA TRANSMISSION OVER GSM VOICE CHANNEL

FOR SECURE VOICE & DATA APPLICATIONS," Secure Mobile Communications Forum: Exploring the Technical Challenges in Secure GSM and WLAN, 2004. The 2nd IEE, vol. 7/1–7/4, 2004.

[31] K. J. CHRISTABEL, S. EMMANUEL, M. S. KANKANHALLI, "Quality-Aware GSM Speech Watermarking," in IEEE International Symposium on Circuits and Systems, ISCAS 2008., 2008, pp. 2965–2968.

[32] J. HOLUB, M. D. STREET, "IMPACT OF END TO END ENCRYPTION ON GSM SPEECH TRANSMISSION QUALITY - A CASE STUDY," in Secure Mobile Communications Forum: Exploring the Technical Challenges in Secure GSM and WLAN, 2004. IEEE, 2004, pp. 6/1–6/4.

[33] S. ISLAM, F. AJMAL, "Developing and implementing encryption algorithm for addressing GSM security issues," 2009 International Conference on Emerging Technologies, pp. 358–361, Oct. 2009.

[34] M. RASHIDI, A. SAYADIYAN, P. MOWLAEE, "Data Mapping onto Speech-like Signal to Transmission over the GSM Voice Channel," in 40th Southeastern Symposium on System Theory, 2008. SSST 2008., 2008, pp. 54–58.

[35] L. CHEN, Q. GUO, "An OFDM-based secure data communicating scheme in GSM voice channel," 2011 International Conference on Electronics, Communications and Control (ICECC), pp. 723–726, Sep. 2011.

[36] Y. YANG, S. FENG, W. YE, X. JI, "A Transmission Scheme for Encrypted Speech over GSM Network," 2008 International Symposium on Computer Science and Computational Technology, pp. 805–808, 2008.

[37] M. RASHIDI, A. SAYADIYAN, P. MOWLAEE, "A Harmonic Approach to Data Transmission over GSM Voice Channel," 2008 3rd International Conference on Information and Communication Technologies: From Theory to Applications, pp. 1–4, Apr. 2008.

[38] W. B. DIAB, S. TOHME, C. BASSIL, "VPN Analysis and New Perspective for Securing Voice over VPN Networks," Fourth International Conference on Networking and Services (icns 2008), pp. 73–78, Mar. 2008.

[39] R. V. LAKSHMI, D. KRISHNAN, P. S., V. K., J. POROOR, A. DHAR, A. V. VIDYAPEETHAM, "Usable and Secure Registration of Guest-Phones into Enterprise VoIP Network," 2010 International Conference on Advances in Computer Engineering, pp. 115–119, Jun. 2010.

[40] G. EPIPHANIOU, C. MAPLE, P. SANT, P. NORRINGTON, "The Effects of Encryption on VoIP Streams under the Code-Excited Linear Prediction Coder G.729," in 2010 International Conference for Internet Technology and Secured Transactions (ICITST 2010),, 2010, pp. 1–6.

[41] W. CHOU, “Strategies to Keep Your VoIP Network Secure,” IT Professional IEEE, no. October, pp. 42–46, 2007.

[42] P. GUPTA, V. SHMATIKOV, “Security Analysis of Voice-over-IP Protocols,” 20th IEEE Computer Security Foundations Symposium (CSF’07), pp. 49–63, Jul. 2007.

[43] G. H. KHAKSARI, A. L. WIJESINHA, R. K. KARNE, “Secure VoIP Using a Bare PC,” 2009 3rd International Conference on New Technologies, Mobility and Security, pp. 1–5, Dec. 2009.

[44] B. TRIKI, S. REKHIS, N. BOUDRIGA, “Secure and QoS-aware SIP handover for VoIP communication in vehicular adhoc networks,” 2011 7th International Wireless Communications and Mobile Computing Conference, pp. 695–700, Jul. 2011.

[45] R. DANSEREAU, S. JIN, R. GOUBRAN, “Reducing Packet Loss in CBC Secured VoIP using Interleaved Encryption,” 2006 Canadian Conference on Electrical and Computer Engineering, no. 2, pp. 1320–1324, 2006.

[46] H. XIAO, P. ZARRELLA, “QUALITY EFFECTS OF WIRELESS VOIP USING SECURITY SOLUTIONS,” in MILCOM 2004 - 2004 IEEE Military Communications Conference, 2004, pp. 1352–1357.

[47] M. LEGGIERI, E. GAMBI, S. SPINSANTE, “Quality assessment of secure VoIP communications,” in 16th International Conference onSoftware, Telecommunications and Computer Networks, SoftCOM 2008., 2008, pp. 135–139.

[48] A. D. ELBAYOUMY, S. J. SHEPHERD, “QoS Control Using an End-Point CPU Capability Detector in a Secure VoIP System,” 10th IEEE Symposium on Computers and Communications (ISCC’05), no. Iscc, pp. 792–797, 2005.

[49] C. Y. YEUN, S. M. AL-MARZOUQI, “Practical Implementations for Securing VoIP Enabled Mobile Devices,” in 2009 Third International Conference on Network and System Security, 2009, pp. 409–414.

[50] S. YOON, J. JEONG, H. JEONG, Y. WON, “Lawful Interception Scheme for Secure VoIP Communications Using TTP,” International Symposium on Computer Science and its Applications, pp. 149–152, Oct. 2008.

[51] J. KIM, S. YOON, H. JEONG, Y. WON, “Implementation and Evaluation of SIP-Based Secure VoIP Communication System,” 2008 IEEE/IFIP International Conference on Embedded and Ubiquitous Computing, pp. 356–360, Dec. 2008.

[52] C. KRATZER, J. DITTMANN, T. VOGEL, R. HILLERT, “Design and evaluation of steganography for voice-over-IP,” 2006 IEEE International Symposium on Circuits and Systems, p. 4, 2006.

[53] A. NASCIMENTO, A. PASSITO, E. MOTA, E. NASCIMENTO, L. CARVALHO, “Can I Add a Secure VoIP Call?,” 2006 International Symposium on a World of Wireless, Mobile and Multimedia Networks(WoWMoM’06), pp. 435–437, 2006.

[54] D. ZISIADIS, S. KOPSIDAS, L. TASSIULAS, “An Architecture for Secure VoIP and Collaboration Applications,” Third International Workshop on Security, Privacy and Trust in Pervasive and Ubiquitous Computing (SecPerU 2007), no. SecPerU, pp. 79–84, Jul. 2007.

[55] F. SATTAR, M. HUSSAIN, K. NISAR, “A secure architecture for open source VoIP solutions,” 2011 International Conference on Information and Communication Technologies, pp. 1–6, Jul. 2011.

[56] R. A. MALANEY, “A Secure and Energy Efficient Scheme for Wireless VoIP Emergency Service,” in Global Telecommunications Conference, 2006. GLOBECOM ’06. IEEE, 2006, pp. 1–6.

[57] M. YAO-HUA, W. BING, “A methodology for the improvement of Voice-over-IP,” 2010 2nd International Conference on Computer Engineering and Technology, pp. V6–401–V6–403, 2010.

[58] C.-H. WANG, M.-W. LI, W. LIAO, “A DISTRIBUTED KEY-CHANGING MECHANISM FOR SECURE VOICE,” in IEEE International Conference on Multimedia and Expo, 2007, 2007, pp. 895–898.

[59] H. TIAN, K. ZHOU, Y. HUANG, D. FENG, J. LIU, “A Covert Communication Model Based on Least Significant Bits Steganography in Voice over IP,” 2008 The 9th International Conference for Young Computer Scientists, pp. 647–652, Nov. 2008.

[60] W. MAZURCZYK, “VoIP Steganography and Its Detection – A Survey,” eprint arXiv:1203.4374, pp. 1–19.

[61] B. GEISER, F. MERTZ, P. VARY, “Steganographic Packet Loss Concealment for Wireless VoIP,” 2008 ITG Conference on Voice Communication (SprachKommunikation),, pp. 1–4, 2008.

[62] J. P. CAMPBELL, R. A. DEAN, “The history of speech coding,” http://www.nsa.gov/about/_files/cryptologic_heritage/publications/wwii/sigsaly_history.pdf. .

[63] B. H. JUANG, T. CHEN, “The past present and future of speech Processing,” IEEE SIGNAL PROCESSING MAGAZINE 1998, vol. 15, no. 3, pp. 24–48, 1998.

[64] N. M. ANAS, Z. RAHMAN, A. SHAFII, M. NAJIB, A. RAHMAN, “Secure Speech Communication over Public Switched Telephone Network SBim,” in Asia-Pacific Conference on Applied Electromagnetics Proceedings, APACE, 2005, pp. 336–339.

[65] D. MANUAL, “TMS320C54CST Client Side Telephony DSP Data Manual,” Texas Instruments, 2001.

[66] J. CALPE, J. R. MAGDALENA, J. F. GUERRERO, J. V. FRANECS, “TOLL-QUALITY DIGITAL SECRAPHONE,” in Electrotechnical Conference, MELECON 96, 1996, no. 2, pp. 1714–1717.

[67] L. DIEZ-DEL-RIO, S. MORENO-PEREZ, R. SARMIENTO, J. PARERA, M. VEIGA-PEREZ, R. GARCIA-GOMEZ, “Secure speech and data communication over the public switching telephone network.,” in International Conference on Acoustics, Speech, and Signal Processing (ICASSP-94), 1994, pp. 425–428.

[68] P. AFFECTED, “DSP32C Digital Signal Processor Data Sheet,” Lucent Technologies, no. 5, 1996.

[69] W. ZHI-JUN, N. XIN-XIN, Y. YI-XIAN, “DESIGN OF SPEECH INFORMATION HIDING TELEPHONE,” in TENCON 02. Proceedings. IEEE Conference on Computers, Communications, Control and Power Engineering, 2002, pp. 113–116.

[70] E. INSAM, TCP/IP embedded internet applications. 2003.

[71] “Si2457/34/15/04 datasheet,” in Silicon Laboratories Si2457/34/15/04, www.silabs.com.

[72] D. BUS, T. PHONE, P. BUS, “Si2493/57/34/15/04 (Revision D) and Si2494/39 Modem Designer’s Guide,” Silicon Laboratories www.silabs.com.

[73] “EMBEDDED MODEM DEVELOPMENT KIT USER’S GUIDE,” in Silicon Laboratories www.silabs.com.

[74] P. MERMELSTEIN, D. J. MILLAR, “ADAPTIVE PREDICTIVE CODING OF SPEECH AND VOICEBAND DATA SIGNALS,” in Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP ’82, 1982, pp. 972–975.

[75] G. BONNEROT, J. RAULIN, M. BELLANGER, “Performance of a 32 kbit/s ADPCM coder for digital long-haul telephone transmission,” in ICASSP ’81.

IEEE International Conference on Acoustics, Speech, and Signal Processing, 1981, vol. 6, pp. 848–851.

[76] J. D. GIBSON, S. K. JONES, J. L. MELSA, "Sequentially Adaptive Prediction and Coding of Speech Signals," IEEE TRANSACTIONS ON COMMUNICATION, vol. COM-22, pp. 1789–1797, 1974.

[77] R. LEFEBVRE, P. GOURNAY, "Speech Coders," in Handbook of Signal Processing in Acoustics Springer, 2009, pp. 587–620.

[78] R. GOLDBERG, L. RIEK, "Speech Coders," in CRC Press, 2000.

[79] D. Y. PAN, "Digital Audio Compression," Digital Technical Journal, Digital Equipment Corp., vol. 5, no. 2, pp. 28–40, 1993.

[80] J. D. GIBSON, "Methods , Standards , and Applications," Cicuits and Systems Magazine, IEEE, pp. 30–49, 2005.

[81] E. SHLOMOT, V. CUPERMAN, A. GERSHO, "Hybrid coding of speech at 4 kbps," in Speech Coding For Telecommunications Proceeding IEEE, 1997, pp. 37–38.

[82] A. MURRAY, S. DANAHER, "LOW BIT RATE SPEECH CODING FOR SINGLE SPEAKERS," in Techniques for Speech Processing and their Application, IEE Colloquium on, 1994.

[83] M. YONG, "Study of voice packet reconstruction methods applied to CELP speech coding," IEEE International Conference on Acoustics, Speech, and Signal Processing ICASSP-92, pp. 125–128 vol.2, 1992.

[84] D. P. KEMP, R. A. SUEDA, T. E. TREMAIN, "An evaluation of 4800bps voice coders," in International Conference on Acoustics, Speech, and Signal Processing. ICASSP-89., 1989, pp. 200–203.

[85] M. BUDAGAVI, J. D. GIBSON, "Speech coding in mobile radio communications," in Proceedings of the IEEE, 1998, vol. 86, no. 7, pp. 1402–1412.

[86] W. JIA, W. CHAN, "Personal speech coding," in Acoustics, Speech and Signal Processing, Proceedings of the 1998 IEEE International Conference, 1998, pp. 65–68.

[87] A. S. SPANIAS, "Speech Coding : A Tutorial Review," in Proceedings of the IEEE, 1994, vol. 82, no. 10, pp. 1541–1582.

[88] L. R. LITWIN, "Spech Coding with Wavelets," IEEE Potentials, pp. 38–41, 1998.

[89] J. V. MACRES, “Real-Time Implementations and Applications of the US Federal Standard CELP Voice Coding Algorithm,” in Military Communications Conference, MILCOM ’92, IEEE, 1991, pp. 373–377.

[90] E. PRYADI, K. GANDI, H. Y. KANALEBE, “SPEECH COMPRESSION USING CELP SPEECH CODING TECHNIQUE IN GSM AMR,” in 2008 5th IFIP International Conference on Wireless and Optical Communications Networks WOCN 08, 2008, no. 021, pp. 1–4.

[91] T. E. TREMAIN, J. S. COLLURA, “A Comparison of Five 16Kbps Voice Coding Algorithms,” in IEEE International Conference on Acoustics, Speech, and Signal Processing, ICASSP-88, 1988, pp. 695–698.

[92] S. R. QUACKENBUSH, “Coding of Natural Audio in MPEG-4,” in Proceedings of the 1998 IEEE International Conference on Acoustics Speech and Signal Processing ICASSP 98, 1998, pp. 3797–3800.

[93] S. KWONG, K. F. MAN, “A speech coding algorithm based on predictive coding,” in Proceedings DCC ’95 Data Compression Conference, 1995, p. 455.

[94] W. C. CHU, “ALGORITHMS SPEECH CODING Foundation and Evolution,” in JOHN WILEY & SONS, INC., 2003.

[95] R. RICHEY, “Adaptive Differential Pulse Code Modulation Using PIC Microcontrocontroller AN 63,” www.microchip.com. .

[96] Ö. ÇETIN, “HAREKETLİ GÖRÜNTÜ ÜZERİNDE VERİ GİZLEME VE ŞİFRELEME YÖNELİMLİ YENİ BİR ALGORİTMA TASARIMI,” Sakarya Üniversitesi, Fen Bilimlei Enstitüsü, 2008.

[97] Y. YALMAN, “SAYISAL SES İÇERİSİNDE GİZLİ VERİ TRANSFERİNİN KABLOSUZ ORTAMDA GERÇEKLEŞTİRİLMESİ,” Kocaeli Üniversitesi, Fen Bilimleri, Enstitüsü, 2007.

[98] N. AOKI, “Lossless Steganography Techniques for IP Telephony Speech Taking Account of the Redundancy of Folded Binary Code,” in 2009 Fifth International Joint Conference on INC, IMS and IDC, 2009, pp. 1689–1692.

[99] F.-X. STANDAERT, G. PIRET, N. GERSHENFELD, J.-J. QUISQUATER, “a Scalable Encryption Algorithm for Small Embedded Applications,” Smart Card Research and Advanced Applications Lecture Notes in Computer Science , Springer, vol. 3928, pp. 222–236.

[100] V. R. ANDEM, “A CRYPTANALYSIS OF THE TINY ENCRYPTION ALGORITHM,” The University of Alabama, 2003.

[101] “Tiny Encryption Algorithm, Extended Tiny Encryption Algorithm,” http://www.cix.co.uk/~klockstone/. .

EKLER

EK A. ADPCMEncoder() Fonksiyonu

```
const int IndexTable[16] = {
                                    -1 ,-1, -1, -1, 2, 4, 6, 8,
                                    -1 ,-1, -1, -1, 2, 4, 6, 8
                                  };

const long StepSizeTable[89] = {
                                      7, 8, 9, 10, 11, 12, 13, 14, 16, 17,
                                      19, 21, 23, 25, 28, 31, 34, 37, 41, 45,
                                      50, 55, 60, 66, 73, 80, 88, 97, 107, 118,
                                      130, 143, 157, 173, 190, 209, 230, 253, 279,
                                    307,
                                      337, 371, 408, 449, 494, 544, 598, 658, 724,
                                    796,
                                      876, 963, 1060, 1166, 1282, 1411, 1552,
                                    1707, 1878, 2066,
                                      2272, 2499, 2749, 3024, 3327, 3660, 4026,
                                    4428, 4871, 5358,
                                      5894, 6484, 7132, 7845, 8630, 9493, 10442,
                                    11487, 12635, 13899,
                                      15289, 16818, 18500, 20350, 22385, 24623,
                                    27086, 29794, 32767
                                  };

signed long diff;
long step;
signed long predsample;
signed long diffq;
int index;

char ADPCMEncoder( signed long sample )
{
        int code;
        int tempstep;

        predsample = prevsample;
```

```
index = previndex;
step = StepSizeTable[index];
diff = sample - predsample;

if(diff >= 0)
        code = 0;
else
        {
                code = 8;
                diff = -diff;
        }

tempstep = step;

if( diff >= tempstep )
        {
                code |= 4;
                diff -= tempstep;
        }
tempstep >>= 1;

if( diff >= tempstep )
        {
                code |= 2;
                diff -= tempstep;
        }

tempstep >>= 1;

if( diff >= tempstep )
        code |= 1;

diffq = step >> 3;

if( code & 4 )
        diffq += step;
if( code & 2 )
        diffq += step >> 1;
if( code & 1 )
        diffq += step >> 2;

if( code & 8 )
        predsample -= diffq;
else
        predsample += diffq;

if( predsample > 32767 )
        predsample = 32767;
```

```
        else if( predsample < -32768 )
                predsample = -32768;

        index += IndexTable[code];

        if( index < 0 )
                index = 0;
        if( index > 88 )
                index = 88;

        prevsample = predsample;
        previndex = index;

        return ( code & 0x0f );
}
```

EK B. ADPCMDecoder() Fonksiyonu

```
const int IndexTable[16] = {
                                -1, -1, -1, -1, 2, 4, 6, 8,
                                -1, -1, -1, -1, 2, 4, 6, 8
                            };

const long StepSizeTable[89] = {
                                    7, 8, 9, 10, 11, 12, 13, 14, 16, 17,
                                    19, 21, 23, 25, 28, 31, 34, 37, 41, 45,
                                    50, 55, 60, 66, 73, 80, 88, 97, 107, 118,
                                    130, 143, 157, 173, 190, 209, 230, 253, 279,
                                307,
                                    337, 371, 408, 449, 494, 544, 598, 658, 724,
                                796,
                                    876, 963, 1060, 1166, 1282, 1411, 1552,
                                1707, 1878, 2066,
                                    2272, 2499, 2749, 3024, 3327, 3660, 4026,
                                4428, 4871, 5358,
                                    5894, 6484, 7132, 7845, 8630, 9493, 10442,
                                11487, 12635, 13899,
                                    15289, 16818, 18500, 20350, 22385, 24623,
                                27086, 29794, 32767
                            };

long step;
signed long predsample;
signed long diffq;
int index;



signed long ADPCMDecoder(char code )
{

        predsample = prevsample;
        index = previndex;

        step = StepSizeTable[index];

        diffq = step >> 3;

        if( code & 4 )
                diffq += step;
        if( code & 2 )
                diffq += step >> 1;
```

```
        if( code & 1 )
                diffq += step >> 2;

        if( code & 8 )
                predsample -= diffq;
        else
                predsample += diffq;


        if( predsample > 32767 )
                predsample = 32767;
        else if( predsample < -32768 )
                predsample = -32768;

        index += IndexTable[code];

        if( index < 0 )
                index = 0;
        if( index > 88 )
                index = 88;

        prevsample = predsample;
        previndex = index;

        return( predsample );
}
```

EK C. SEA Şifreleme Fonksiyonu

```
NAMESea_e

?PR?sea_e?Sea_e                 SEGMENT CODE

        PUBLIC      sea_e


        RSEG  ?PR?sea_e?Sea_e
sea_e:

Cnt    EQU  R7

#define         RVec   0x0e
#define         LVec   0x08
#define         RKey   0x28
#define         LKey   0x1C



;=======================          forward          SEA          (encrypt)
==============================

Sea:        ;b=8, nb=6, 93 rounds
                                mov             a,#0E7h
                                anl             0D0h,a
    mov   Cnt,#1 ;23    ;i<[93+1]/2... -> 46

SeaX11:
    ;--- round 1
    mov   a,LVec+0    ;pre-word-rotation of LVec
    xch   a,LVec+1
    xch   a,LVec+2
    xch   a,LVec+3
    xch   a,LVec+4
    xch   a,LVec+5
    mov   LVec+0,a

    mov   a,RVec+0    ;--- round 1
    add   a,RKey+0
    mov   R2,a       ;+0
    mov   a,RVec+1
    add   a,RKey+1
```

```
    mov   R3,a       ;+1
    mov   a,RVec+2
    add   a,RKey+2
    mov   R4,a       ;+2
    anl   a,R3
    xrl   a,R2
    mov   R2,a       ;+0 stored for the rest of substitution
    rr    a          ;bitwise rotation
    xrl   LVec+0,a
    mov   a,R4
    anl   a,R2
    xrl   a,R3
    xrl   LVec+1,a   ;=+1
    orl   a,R2
    xrl   a,R4
    rl    a
    xrl   LVec+2,a   ;=+2

    mov   a,RVec+3   ;----- repeat for following 3 bytes (4,5,6)
    add   a,RKey+3
    mov   R2,a
    mov   a,RVec+4
    add   a,RKey+4
    mov   R3,a
    mov   a,RVec+5
    add   a,RKey+5
    mov   R4,a
    anl   a,R3
    xrl   a,R2
    mov   R2,a
    rr    a
    xrl   LVec+3,a
    mov   a,R4
    anl   a,R2
    xrl   a,R3
    xrl   LVec+4,a
    orl   a,R2
    xrl   a,R4
    rl    a
    xrl   LVec+5,a

    cjne  Cnt,#47,SeaX12    ;when halfway through, swap keys, other key schedule
and other key used
    ljmp  SeaX21
SeaX12:

    mov   a,RKey+0   ;--- round 1 Key schedule
    add   a,Cnt          ;assuming that all_rounds < 512
```

```
        mov   R2,a       ;+0
        mov   a,RKey+2
        anl   a,RKey+1
        xrl   a,R2
        mov   R2,a       ;+0 stored for the rest of substitution
        rr    a          ;bitwise rotation
        xrl   LKey+1,a   ;this includes the bytewise rotation (+0->+1); this needs no
compensation
        mov   a,RKey+2
        anl   a,R2
        xrl   a,RKey+1
        xrl   LKey+2,a
        orl   a,R2
        xrl   a,RKey+2
        rl    a
        xrl   LKey+3,a

        mov   a,RKey+5   ;--- the same for following 3 bytes of key
        anl   a,RKey+4
        xrl   a,RKey+3
        mov   R2,a
        rr    a
        xrl   LKey+4,a
        mov   a,RKey+5
        anl   a,R2
        xrl   a,RKey+4
        xrl   LKey+5,a
        orl   a,R2
        xrl   a,RKey+5
        rl    a
        xrl   LKey+0,a

        inc   Cnt

        ;------- now round 2; rename R<->L both Vec and Key;
        mov   a,RVec+0   ;pre-word-rotation of RVec
        xch   a,RVec+1
        xch   a,RVec+2
        xch   a,RVec+3
        xch   a,RVec+4
        xch   a,RVec+5
        mov   RVec+0,a

        mov   a,LVec+0   ;--- round 2
        add   a,LKey+0
        mov   R2,a       ;+0
        mov   a,LVec+1
        add   a,LKey+1
```

```
mov   R3,a       ;+1
mov   a,LVec+2
add   a,LKey+2
mov   R4,a       ;+2
anl   a,R3
xrl   a,R2
mov   R2,a       ;+0 stored for the rest of substitution
rr    a          ;bitwise rotation
xrl   RVec+0,a
mov   a,R4
anl   a,R2
xrl   a,R3
xrl   RVec+1,a   ;=+1
orl   a,R2
xrl   a,R4
rl    a
xrl   RVec+2,a   ;=+2

mov   a,LVec+3   ;----- repeat for following 3 bytes (4,5,6)
add   a,LKey+3
mov   R2,a
mov   a,LVec+4
add   a,LKey+4
mov   R3,a
mov   a,LVec+5
add   a,LKey+5
mov   R4,a
anl   a,R3
xrl   a,R2
mov   R2,a
rr    a
xrl   RVec+3,a
mov   a,R4
anl   a,R2
xrl   a,R3
xrl   RVec+4,a
orl   a,R2
xrl   a,R4
rl    a
xrl   RVec+5,a

mov   a,LKey+0   ;--- round 2 Key schedule
add   a,Cnt          ;assuming that all_rounds < 512
mov   R2,a       ;+0
mov   a,LKey+2
anl   a,LKey+1
xrl   a,R2
mov   R2,a       ;+0 stored for the rest of substitution
```

```
    rr   a          ;bitwise rotation
    xrl   RKey+1,a     ;this includes the bytewise rotation (+0->+1); this needs no
compensation
    mov   a,LKey+2
    anl   a,R2
    xrl   a,LKey+1
    xrl   RKey+2,a
    orl   a,R2
    xrl   a,LKey+2
    rl    a
    xrl   RKey+3,a

    mov   a,LKey+5    ;--- the same for following 3 bytes of key
    anl   a,LKey+4
    xrl   a,LKey+3
    mov   R2,a
    rr    a
    xrl   RKey+4,a
    mov   a,LKey+5
    anl   a,R2
    xrl   a,LKey+4
    xrl   RKey+5,a
    orl   a,R2
    xrl   a,LKey+5
    rl    a
    xrl   RKey+0,a

    inc   Cnt
    ljmp  SeaX11


    ;--- halfway through. Continue with swap keys, other key schedule (n-cnt), other
key used
    ;--- key swap by renaming RKey<->LKey
    ;don't forget that Vec needs to be renamed too as we left the 1st part after odd nr
of round
    ;

SeaX21:
    dec   Cnt        ;--- this is instead of nround-i
    mov   a,LKey+0    ;--- odd round key schedule, but key is swapped -> renamed
    add   a,Cnt           ;assuming that all_rounds < 512
    mov   R2,a       ;+0
    mov   a,LKey+2
    anl   a,LKey+1
    xrl   a,R2
    mov   R2,a       ;+0 stored for the rest of substitution
    rr    a          ;bitwise rotation
```

```
    xrl   RKey+1,a     ;this includes the bytewise rotation (+0->+1); this needs no
compensation
    mov   a,LKey+2
    anl   a,R2
    xrl   a,LKey+1
    xrl   RKey+2,a
    orl   a,R2
    xrl   a,LKey+2
    rl    a
    xrl   RKey+3,a

    mov   a,LKey+5    ;--- the same for following 3 bytes of key
    anl   a,LKey+4
    xrl   a,LKey+3
    mov   R2,a
    rr    a
    xrl   RKey+4,a
    mov   a,LKey+5
    anl   a,R2
    xrl   a,LKey+4
    xrl   RKey+5,a
    orl   a,R2
    xrl   a,LKey+5
    rl    a
    xrl   RKey+0,a

    ;------- now even round; rename R<->L both Vec and Key (but Key is already
renamed due to swap; but the other part of key is used in part2...);
    mov   a,RVec+0     ;pre-word-rotation of RVec
    xch   a,RVec+1
    xch   a,RVec+2
    xch   a,RVec+3
    xch   a,RVec+4
    xch   a,RVec+5
    mov   RVec+0,a

    mov   a,LVec+0    ;--- even round
    add   a,LKey+0
    mov   R2,a        ;+0
    mov   a,LVec+1
    add   a,LKey+1
    mov   R3,a        ;+1
    mov   a,LVec+2
    add   a,LKey+2
    mov   R4,a        ;+2
    anl   a,R3
    xrl   a,R2
    mov   R2,a        ;+0 stored for the rest of substitution
```

```
    rr   a          ;bitwise rotation
    xrl  RVec+0,a
    mov  a,R4
    anl  a,R2
    xrl  a,R3
    xrl  RVec+1,a   ;=+1
    orl  a,R2
    xrl  a,R4
    rl   a
    xrl  RVec+2,a   ;=+2

    mov  a,LVec+3   ;----- repeat for following 3 bytes (4,5,6)
    add  a,LKey+3
    mov  R2,a
    mov  a,LVec+4
    add  a,LKey+4
    mov  R3,a
    mov  a,LVec+5
    add  a,LKey+5
    mov  R4,a
    anl  a,R3
    xrl  a,R2
    mov  R2,a
    rr   a
    xrl  RVec+3,a
    mov  a,R4
    anl  a,R2
    xrl  a,R3
    xrl  RVec+4,a
    orl  a,R2
    xrl  a,R4
    rl   a
    xrl  RVec+5,a

    dec  Cnt
    mov  a,RKey+0   ;--- even round Key schedule - due to swap renamed LKey<-
>RKey
    add  a,Cnt          ;assuming that all_rounds < 512
    mov  R2,a       ;+0
    mov  a,RKey+2
    anl  a,RKey+1
    xrl  a,R2
    mov  R2,a       ;+0 stored for the rest of substitution
    rr   a          ;bitwise rotation
    xrl  LKey+1,a   ;this includes the bytewise rotation (+0->+1); this needs no
compensation
    mov  a,RKey+2
    anl  a,R2
```

```
xrl   a,RKey+1
xrl   LKey+2,a
orl   a,R2
xrl   a,RKey+2
rl    a
xrl   LKey+3,a

mov   a,RKey+5    ;--- the same for following 3 bytes of key
anl   a,RKey+4
xrl   a,RKey+3
mov   R2,a
rr    a
xrl   LKey+4,a
mov   a,RKey+5
anl   a,R2
xrl   a,RKey+4
xrl   LKey+5,a
orl   a,R2
xrl   a,RKey+5
rl    a
xrl   LKey+0,a

;--- odd round
mov   a,LVec+0     ;pre-word-rotation of LVec
xch   a,LVec+1
xch   a,LVec+2
xch   a,LVec+3
xch   a,LVec+4
xch   a,LVec+5
mov   LVec+0,a

mov   a,RVec+0    ;--- odd round
add   a,RKey+0
mov   R2,a        ;+0
mov   a,RVec+1
add   a,RKey+1
mov   R3,a        ;+1
mov   a,RVec+2
add   a,RKey+2
mov   R4,a        ;+2
anl   a,R3
xrl   a,R2
mov   R2,a        ;+0 stored for the rest of substitution
rr    a           ;bitwise rotation
xrl   LVec+0,a
mov   a,R4
anl   a,R2
xrl   a,R3
```

```
        xrl   LVec+1,a    ;=+1
        orl   a,R2
        xrl   a,R4
        rl    a
        xrl   LVec+2,a    ;=+2

        mov   a,RVec+3    ;----- repeat for following 3 bytes (4,5,6)
        add   a,RKey+3
        mov   R2,a
        mov   a,RVec+4
        add   a,RKey+4
        mov   R3,a
        mov   a,RVec+5
        add   a,RKey+5
        mov   R4,a
        anl   a,R3
        xrl   a,R2
        mov   R2,a
        rr    a
        xrl   LVec+3,a
        mov   a,R4
        anl   a,R2
        xrl   a,R3
        xrl   LVec+4,a
        orl   a,R2
        xrl   a,R4
        rl    a
        xrl   LVec+5,a

        cjne  Cnt,#1,SeaX22
        sjmp  SeaX23
SeaX22: ljmp  SeaX21
SeaX23:
        ;----- finished. LKey<->RKey swap not needed due to previous rename.
        ;                LVec<->RVec swap not needed as the last step did not swap (as
none does).

        ret

END
```

EK D. SEA Şifre Çözme Fonksiyonu

```
NAMESea_d

?PR?sea_d?Sea_d                 SEGMENT CODE

        PUBLIC      sea_d


        RSEG  ?PR?sea_d?Sea_d
sea_d:

Cnt    EQU  R7

#define         RVec  0x0e
#define         LVec  0x08
#define         RKey  0x22
#define         LKey  0x16

;=======================          inverse          SEA          (decrypt)
==============================

;--- differs from forward SEA only in placement of word rotation - it is
;    pre-word-rotation in forward and post-inverse-word-rotation in backward



ISea:          ;b=8, nb=6, 93 rounds
                                mov             a,#0E7h
                                anl             0D0h,a
      mov   Cnt,#1 ;23     ;i<[93+1]/2... -> 46

ISeaX11:
      ;--- round 1
      mov   a,RVec+0    ;--- round 1
      add   a,RKey+0
      mov   R2,a       ;+0
      mov   a,RVec+1
      add   a,RKey+1
      mov   R3,a       ;+1
      mov   a,RVec+2
      add   a,RKey+2
      mov   R4,a       ;+2
      anl   a,R3
      xrl   a,R2
```

```
    mov   R2,a       ;+0 stored for the rest of substitution
    rr    a          ;bitwise rotation
    xrl   LVec+0,a
    mov   a,R4
    anl   a,R2
    xrl   a,R3
    xrl   LVec+1,a   ;=+1
    orl   a,R2
    xrl   a,R4
    rl    a
    xrl   LVec+2,a   ;=+2

    mov   a,RVec+3   ;----- repeat for following 3 bytes (4,5,6)
    add   a,RKey+3
    mov   R2,a
    mov   a,RVec+4
    add   a,RKey+4
    mov   R3,a
    mov   a,RVec+5
    add   a,RKey+5
    mov   R4,a
    anl   a,R3
    xrl   a,R2
    mov   R2,a
    rr    a
    xrl   LVec+3,a
    mov   a,R4
    anl   a,R2
    xrl   a,R3
    xrl   LVec+4,a
    orl   a,R2
    xrl   a,R4
    rl    a
    xrl   LVec+5,a

    mov   a,LVec+0    ;post-inverse-word-rotation of LVec
    xch   a,LVec+5
    xch   a,LVec+4
    xch   a,LVec+3
    xch   a,LVec+2
    xch   a,LVec+1
    mov   LVec+0,a



    cjne  Cnt,#47,ISeaX12    ;when halfway through, swap keys, other key schedule
and other key used
    ljmp  ISeaX21
```

```
ISeaX12:

        mov   a,RKey+0    ;--- round 1 Key schedule
        add   a,Cnt           ;assuming that all_rounds < 512
        mov   R2,a        ;+0
        mov   a,RKey+2
        anl   a,RKey+1
        xrl   a,R2
        mov   R2,a        ;+0 stored for the rest of substitution
        rr    a           ;bitwise rotation
        xrl   LKey+1,a    ;this includes the bytewise rotation (+0->+1); this needs no
compensation
        mov   a,RKey+2
        anl   a,R2
        xrl   a,RKey+1
        xrl   LKey+2,a
        orl   a,R2
        xrl   a,RKey+2
        rl    a
        xrl   LKey+3,a

        mov   a,RKey+5    ;--- the same for following 3 bytes of key
        anl   a,RKey+4
        xrl   a,RKey+3
        mov   R2,a
        rr    a
        xrl   LKey+4,a
        mov   a,RKey+5
        anl   a,R2
        xrl   a,RKey+4
        xrl   LKey+5,a
        orl   a,R2
        xrl   a,RKey+5
        rl    a
        xrl   LKey+0,a

        inc   Cnt

        ;------- now round 2; rename R<->L both Vec and Key;
        mov   a,LVec+0    ;--- round 2
        add   a,LKey+0
        mov   R2,a        ;+0
        mov   a,LVec+1
        add   a,LKey+1
        mov   R3,a        ;+1
        mov   a,LVec+2
        add   a,LKey+2
        mov   R4,a        ;+2
```

```
anl   a,R3
xrl   a,R2
mov   R2,a       ;+0 stored for the rest of substitution
rr    a          ;bitwise rotation
xrl   RVec+0,a
mov   a,R4
anl   a,R2
xrl   a,R3
xrl   RVec+1,a   ;=+1
orl   a,R2
xrl   a,R4
rl    a
xrl   RVec+2,a   ;=+2

mov   a,LVec+3   ;----- repeat for following 3 bytes (4,5,6)
add   a,LKey+3
mov   R2,a
mov   a,LVec+4
add   a,LKey+4
mov   R3,a
mov   a,LVec+5
add   a,LKey+5
mov   R4,a
anl   a,R3
xrl   a,R2
mov   R2,a
rr    a
xrl   RVec+3,a
mov   a,R4
anl   a,R2
xrl   a,R3
xrl   RVec+4,a
orl   a,R2
xrl   a,R4
rl    a
xrl   RVec+5,a

mov   a,RVec+0    ;post-inverse-word-rotation of RVec
xch   a,RVec+5
xch   a,RVec+4
xch   a,RVec+3
xch   a,RVec+2
xch   a,RVec+1
mov   RVec+0,a


mov   a,LKey+0   ;--- round 2 Key schedule
add   a,Cnt          ;assuming that all_rounds < 512
```

```
        mov   R2,a       ;+0
        mov   a,LKey+2
        anl   a,LKey+1
        xrl   a,R2
        mov   R2,a       ;+0 stored for the rest of substitution
        rr    a          ;bitwise rotation
        xrl   RKey+1,a   ;this includes the bytewise rotation (+0->+1); this needs no
compensation
        mov   a,LKey+2
        anl   a,R2
        xrl   a,LKey+1
        xrl   RKey+2,a
        orl   a,R2
        xrl   a,LKey+2
        rl    a
        xrl   RKey+3,a

        mov   a,LKey+5   ;--- the same for following 3 bytes of key
        anl   a,LKey+4
        xrl   a,LKey+3
        mov   R2,a
        rr    a
        xrl   RKey+4,a
        mov   a,LKey+5
        anl   a,R2
        xrl   a,LKey+4
        xrl   RKey+5,a
        orl   a,R2
        xrl   a,LKey+5
        rl    a
        xrl   RKey+0,a

        inc   Cnt
        ljmp  ISeaX11


        ;--- halfway through. Continue with swap keys, other key schedule (n-cnt), other
key used
        ;--- key swap by renaming RKey<->LKey
        ;don't forget that Vec needs to be renamed too as we left the 1st part after odd nr
of round
        ;

ISeaX21:
        dec   Cnt        ;--- this is instead of nround-i
        mov   a,LKey+0   ;--- odd round key schedule, but key is swapped -> renamed
        add   a,Cnt          ;assuming that all_rounds < 512
        mov   R2,a       ;+0
```

```
    mov   a,LKey+2
    anl   a,LKey+1
    xrl   a,R2
    mov   R2,a       ;+0 stored for the rest of substitution
    rr    a          ;bitwise rotation
    xrl   RKey+1,a   ;this includes the bytewise rotation (+0->+1); this needs no compensation
    mov   a,LKey+2
    anl   a,R2
    xrl   a,LKey+1
    xrl   RKey+2,a
    orl   a,R2
    xrl   a,LKey+2
    rl    a
    xrl   RKey+3,a

    mov   a,LKey+5   ;--- the same for following 3 bytes of key
    anl   a,LKey+4
    xrl   a,LKey+3
    mov   R2,a
    rr    a
    xrl   RKey+4,a
    mov   a,LKey+5
    anl   a,R2
    xrl   a,LKey+4
    xrl   RKey+5,a
    orl   a,R2
    xrl   a,LKey+5
    rl    a
    xrl   RKey+0,a

    ;------- now even round; rename R<->L both Vec and Key (but Key is already renamed due to swap; but the other part of key is used in part2...);

    mov   a,LVec+0   ;--- even round
    add   a,LKey+0
    mov   R2,a       ;+0
    mov   a,LVec+1
    add   a,LKey+1
    mov   R3,a       ;+1
    mov   a,LVec+2
    add   a,LKey+2
    mov   R4,a       ;+2
    anl   a,R3
    xrl   a,R2
    mov   R2,a       ;+0 stored for the rest of substitution
    rr    a          ;bitwise rotation
    xrl   RVec+0,a
```

```
      mov   a,R4
      anl   a,R2
      xrl   a,R3
      xrl   RVec+1,a    ;=+1
      orl   a,R2
      xrl   a,R4
      rl    a
      xrl   RVec+2,a    ;=+2

      mov   a,LVec+3    ;----- repeat for following 3 bytes (4,5,6)
      add   a,LKey+3
      mov   R2,a
      mov   a,LVec+4
      add   a,LKey+4
      mov   R3,a
      mov   a,LVec+5
      add   a,LKey+5
      mov   R4,a
      anl   a,R3
      xrl   a,R2
      mov   R2,a
      rr    a
      xrl   RVec+3,a
      mov   a,R4
      anl   a,R2
      xrl   a,R3
      xrl   RVec+4,a
      orl   a,R2
      xrl   a,R4
      rl    a
      xrl   RVec+5,a

      mov   a,RVec+0     ;post-inverse-word-rotation of RVec
      xch   a,RVec+5
      xch   a,RVec+4
      xch   a,RVec+3
      xch   a,RVec+2
      xch   a,RVec+1
      mov   RVec+0,a



      dec   Cnt
      mov   a,RKey+0    ;--- even round Key schedule - due to swap renamed LKey<-
>RKey
      add   a,Cnt           ;assuming that all_rounds < 512
      mov   R2,a       ;+0
      mov   a,RKey+2
```

```
anl   a,RKey+1
xrl   a,R2
mov   R2,a       ;+0 stored for the rest of substitution
rr    a          ;bitwise rotation
xrl   LKey+1,a   ;this includes the bytewise rotation (+0->+1); this needs no compensation
mov   a,RKey+2
anl   a,R2
xrl   a,RKey+1
xrl   LKey+2,a
orl   a,R2
xrl   a,RKey+2
rl    a
xrl   LKey+3,a

mov   a,RKey+5   ;--- the same for following 3 bytes of key
anl   a,RKey+4
xrl   a,RKey+3
mov   R2,a
rr    a
xrl   LKey+4,a
mov   a,RKey+5
anl   a,R2
xrl   a,RKey+4
xrl   LKey+5,a
orl   a,R2
xrl   a,RKey+5
rl    a
xrl   LKey+0,a

;--- odd round
mov   a,RVec+0   ;--- odd round
add   a,RKey+0
mov   R2,a       ;+0
mov   a,RVec+1
add   a,RKey+1
mov   R3,a       ;+1
mov   a,RVec+2
add   a,RKey+2
mov   R4,a       ;+2
anl   a,R3
xrl   a,R2
mov   R2,a       ;+0 stored for the rest of substitution
rr    a          ;bitwise rotation
xrl   LVec+0,a
mov   a,R4
anl   a,R2
xrl   a,R3
```

```
        xrl   LVec+1,a    ;=+1
        orl   a,R2
        xrl   a,R4
        rl    a
        xrl   LVec+2,a    ;=+2

        mov   a,RVec+3    ;----- repeat for following 3 bytes (4,5,6)
        add   a,RKey+3
        mov   R2,a
        mov   a,RVec+4
        add   a,RKey+4
        mov   R3,a
        mov   a,RVec+5
        add   a,RKey+5
        mov   R4,a
        anl   a,R3
        xrl   a,R2
        mov   R2,a
        rr    a
        xrl   LVec+3,a
        mov   a,R4
        anl   a,R2
        xrl   a,R3
        xrl   LVec+4,a
        orl   a,R2
        xrl   a,R4
        rl    a
        xrl   LVec+5,a

        mov   a,LVec+0    ;post-inverse-word-rotation of LVec
        xch   a,LVec+5
        xch   a,LVec+4
        xch   a,LVec+3
        xch   a,LVec+2
        xch   a,LVec+1
        mov   LVec+0,a


        cjne  Cnt,#1,ISeaX22
        sjmp  ISeaX23
ISeaX22:
        ljmp  ISeaX21
ISeaX23:
        ;----- finished. LKey<->RKey swap not needed due to previous rename.
        ;                LVec<->RVec swap not needed as the last step did not swap (as
none does).

        ret
```

END

EK E. XTEA Şifre Çözme Fonksiyonu

```
;(C)2005 wek  http://www.efton.sk
;Free for personal use.
;For commercial use contact wek@efton.sk

;Implements inverse of the modified TEA algorithm by Needham&Wheeler in any
'51 compatible mcu

;uses r0,r4,r5,r6,r7,dptr
;uses r2 as round counter
;
;7051 cycles including ret
;0E0h=224 bytes including 16 bytes of key



;round:
;       z -= (((y << 4) ^ (y >> 5) +  y) ^ (sum + k[sum>>11 & 3])
;       sum -= delta;
;       y -= (((z << 4) ^ (z >> 5) +  z) ^ (sum + k[sum&3])
;
;
;
;Key can be also in IRAM or XRAM; with some modification.
;See xtea.a51 for comment on this.
;


   DSEG  AT  30h
y0:   DS    1
y1:   DS    1
y2:   DS    1
y3:   DS    1
z0:   DS    1
z1:   DS    1
z2:   DS    1
z3:   DS    1
tmp0: DS    1
tmp1: DS    1
tmp2: DS    1
tmp3: DS    1
sum0: DS    1
sum1: DS    1
sum2: DS    1
sum3: DS    1
```

```
   CSEG
XTea:
    mov   r2,#32*2   ;nr of rounds *2 (because of trick with twice the main code, one for y and one for z; and another inside...)
    mov   sum3,#0C6h
    mov   sum2,#0EFh
    mov   sum1,#037h
    mov   sum0,#020h

    mov   dptr,#key  ;dptr will not change
TeaRound:

    mov   r4,y0
    mov   r5,y1
    mov   r6,y2
    mov   r7,y3

TeaSubRound:
    mov   r0,#tmp3    ;tmp = y << 4
    mov   a,r7
    swap  a
    mov   @r0,a          ;@r0=tmp3
    mov   a,r6
    swap  a
    xchd  a,@r0          ;@r0=tmp3
    dec   r0
    mov   @r0,a          ;@r0=tmp2
    mov   a,r5
    swap  a
    xchd  a,@r0          ;@r0=tmp2
    dec   r0
    mov   @r0,a          ;@r0=tmp1
    mov   a,r4
    swap  a
    xchd  a,@r0          ;@r0=tmp1
    mov   tmp0,a
    anl   tmp0,#0F0h

    rrc   a            ;tmp ^=  y >> 5
    anl   a,#07h
    xrl   a,tmp3
    xch   a,tmp3
    rrc   a
    xrl   a,tmp2
    xch   a,tmp2
    rrc   a
    xrl   a,@r0  ;tmp1
```

```
    xch   a,@r0  ;tmp1
    rrc   a
    xrl   a,tmp0

    add   a,r4        ;y = y+tmp
    mov   r4,a
    mov   a,r5
    addc  a,tmp1
    mov   r5,a
    mov   a,r6
    addc  a,tmp2
    mov   r6,a
    mov   a,r7
    addc  a,tmp3
    mov   r7,a

    mov   a,r2
    jnb   acc.0,TeaX1
    mov   a,sum0        ;r0 = [sum&3]
    rl    a
    rl    a
    sjmp  TeaX2
TeaX1:
    mov   a,sum1        ;r0 = [sum>>11&3]
    rr    a
TeaX2:
    anl   a,#0Ch
    mov   r0,a

    movc  a,@a+dptr     ;result ^= sum + k[pointer]
    inc   r0
    add   a,sum0
    xrl   a,r4
    mov   r4,a
    mov   a,r0
    movc  a,@a+dptr
    inc   r0
    addc  a,sum1
    xrl   a,r5
    mov   r5,a
    mov   a,r0
    movc  a,@a+dptr
    inc   r0
    addc  a,sum2
    xrl   a,r6
    mov   r6,a
    mov   a,r0
    movc  a,@a+dptr
```

```
    addc  a,sum3
    xrl   a,r7
    mov   r7,a

    dec   r2
    mov   a,r2
    jb    acc.0,TeaSubRound2

    clr   c
    mov   a,y0
    subb  a,r4
    mov   y0,a
    mov   a,y1
    subb  a,r5
    mov   y1,a
    mov   a,y2
    subb  a,r6
    mov   y2,a
    mov   a,y3
    subb  a,r7
    mov   y3,a

    cjne  r2,#0,TeaRoundA
    ret
TeaRoundA:
    jmp   TeaRound

TeaSubRound2:
    clr   c
    mov   a,z0
    subb  a,r4
    mov   z0,a
    mov   r4,a
    mov   a,z1
    subb  a,r5
    mov   z1,a
    mov   r5,a
    mov   a,z2
    subb  a,r6
    mov   z2,a
    mov   r6,a
    mov   a,z3
    subb  a,r7
    mov   z3,a
    mov   r7,a

    clr   c
    mov   a,sum0   ;sum += delta
```

```
    subb  a,#0B9h    ;delta[0]
    mov   sum0,a
    mov   a,sum1
    subb  a,#079h    ;delta[1]
    mov   sum1,a
    mov   a,sum2
    subb  a,#037h    ;delta[2]
    mov   sum2,a
    mov   a,sum3
    subb  a,#09Eh    ;delta[3]
    mov   sum3,a

    jmp   TeaSubRound


Key:
    db    000h,000h,000h,000h
    db    000h,000h,000h,000h
    db    000h,000h,000h,000h
    db    000h,000h,000h,000h

    end
```

EK F. XTEA Şifreleme Fonksiyonu

```
;(C)2005 wek  http://www.efton.sk
;Free for personal use.
;For commercial use contact wek@efton.sk

;Implements the modified TEA algorithm by Needham&Wheeler in any '51 compatible mcu

;uses r0,r4,r5,r6,r7,dptr
;uses r2 as round counter
;
;6952 cycles including ret
;0DAh=218 bytes including 16 bytes of key

;round:
;       y += (((z << 4) ^ (z >> 5) +  z) ^ (sum + k[sum&3])
;       sum += delta;
;       z += (((y << 4) ^ (y >> 5) +  y) ^ (sum + k[sum>>11 & 3])
;
;
;
;Key can be also in IRAM or XRAM; needs some modification for this.
;IRAM is easy; simply add Key to r0 after calculating offset and replace mov a,r0; movc a,@a+dptr by mov a,@r0.
;XRAM is slightly more complicated, the complete pointer should be calculated to dptr
; (or the 8-bit movc a,@r0 shall be used, with p2 properly set; but only if the whole key does not cross the 256-byte boundary...)
;
;


   DSEG  AT  30h
y0:   DS    1
y1:   DS    1
y2:   DS    1
y3:   DS    1
z0:   DS    1
z1:   DS    1
z2:   DS    1
z3:   DS    1
tmp0: DS    1
tmp1: DS    1
tmp2: DS    1
tmp3: DS    1
```

```
sum0: DS    1
sum1: DS    1
sum2: DS    1
sum3: DS    1

   CSEG
XTea:
     clr   a
     mov   sum0,a   ;sum = 0
     mov   sum1,a
     mov   sum2,a
     mov   sum3,a
     mov   r2,#32*2   ;nr of rounds *2 (because of trick with twice the main code, one
for y and one for z; and another inside...)

     mov   dptr,#key  ;dptr will not change
TeaRound:

     mov   r4,z0
     mov   r5,z1
     mov   r6,z2
     mov   r7,z3

TeaSubRound:
     mov   r0,#tmp3    ;tmp = z << 4
     mov   a,r7
     swap  a
     mov   @r0,a          ;@r0=tmp3
     mov   a,r6
     swap  a
     xchd  a,@r0          ;@r0=tmp3
     dec   r0
     mov   @r0,a          ;@r0=tmp2
     mov   a,r5
     swap  a
     xchd  a,@r0          ;@r0=tmp2
     dec   r0
     mov   @r0,a          ;@r0=tmp1
     mov   a,r4
     swap  a
     xchd  a,@r0          ;@r0=tmp1
     mov   tmp0,a
     anl   tmp0,#0F0h

     rrc   a            ;tmp ^=  z >> 5
     anl   a,#07h
     xrl   a,tmp3
     xch   a,tmp3
```

```
    rrc   a
    xrl   a,tmp2
    xch   a,tmp2
    rrc   a
    xrl   a,@r0  ;tmp1
    xch   a,@r0  ;tmp1
    rrc   a
    xrl   a,tmp0


    add   a,r4        ;z = z+tmp
    mov   r4,a
    mov   a,r5
    addc  a,tmp1
    mov   r5,a
    mov   a,r6
    addc  a,tmp2
    mov   r6,a
    mov   a,r7
    addc  a,tmp3
    mov   r7,a

    mov   a,r2
    jb    acc.0,TeaX1
    mov   a,sum0        ;r0 = [sum&3]
    rl    a
    rl    a
    sjmp  TeaX2
TeaX1:
    mov   a,sum1        ;r0 = [sum>>11&3]
    rr    a
TeaX2:
    anl   a,#0Ch
    mov   r0,a

    movc  a,@a+dptr     ;result ^= sum + k[pointer]
    inc   r0
    add   a,sum0
    xrl   a,r4
    mov   r4,a
    mov   a,r0
    movc  a,@a+dptr
    inc   r0
    addc  a,sum1
    xrl   a,r5
    mov   r5,a
    mov   a,r0
    movc  a,@a+dptr
```

```
    inc   r0
    addc  a,sum2
    xrl   a,r6
    mov   r6,a
    mov   a,r0
    movc  a,@a+dptr
    addc  a,sum3
    xrl   a,r7
    mov   r7,a

    dec   r2
    mov   a,r2
    jb    acc.0,TeaSubRound2

    mov   a,r4
    add   a,z0
    mov   z0,a
    mov   a,r5
    addc  a,z1
    mov   z1,a
    mov   a,r6
    addc  a,z2
    mov   z2,a
    mov   a,r7
    addc  a,z3
    mov   z3,a

    cjne  r2,#0,TeaRoundA
    ret
TeaRoundA:
    jmp   TeaRound

TeaSubRound2:
    mov   a,r4
    add   a,y0
    mov   y0,a
    mov   r4,a
    mov   a,r5
    addc  a,y1
    mov   y1,a
    mov   r5,a
    mov   a,r6
    addc  a,y2
    mov   y2,a
    mov   r6,a
    mov   a,r7
    addc  a,y3
    mov   y3,a
```

```
    mov   r7,a

    mov   a,sum0   ;sum += delta
    add   a,#0B9h    ;delta[0]
    mov   sum0,a
    mov   a,sum1
    addc  a,#079h    ;delta[1]
    mov   sum1,a
    mov   a,sum2
    addc  a,#037h    ;delta[2]
    mov   sum2,a
    mov   a,sum3
    addc  a,#09Eh    ;delta[3]
    mov   sum3,a

    jmp   TeaSubRound


Key:
    db    000h,000h,000h,000h
    db    000h,000h,000h,000h
    db    000h,000h,000h,000h
    db    000h,000h,000h,000h

    end
```

ÖZGEÇMİŞ

Ahmet KARACA, 1977 yılında Balıkesir'de doğmuştur. İlk ve orta öğrenimini Balıkesir'de, lise öğrenimini Balıkesir 100. Yıl Endüstri Meslek Lisesi Elektronik bölümünde 1994 yılında tamamlamıştır. 1996 yılında Sakarya Üniversitesi Teknik Eğitim Fakültesi Elektronik ve Bilgisayar Eğitimi Bölümü Elektronik Öğretmenliği programına başlayarak lisans öğreniminden 2000 yılında mezun olmuştur. 2000-2003 yılları arasında Sakarya Üniversitesi Fen Bilimleri Enstitüsü Elektronik ve Bilgisayar Eğitimi ana bilim dalında yüksek lisans eğitimini tamamlamıştır. 2005 yılında Sakarya Üniversitesi Fen Bilimleri Enstitüsü Elektronik ve Bilgisayar Eğitimi ana bilim dalında doktora eğitimine başlamıştır. Aralık 2000'de Sakarya Üniversitesi Teknik Eğitim Fakültesi Elektronik ve Bilgisayar Eğitimi Bölümü'nde araştırma görevlisi olarak başladığı görevine 2003 yılından bu yana uzman olarak devam etmektedir.