

T.C.
TRAKYA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
BİLGİSAYAR MÜHENDİSLİĞİ BÖLÜMÜ



HATA DÜZELTME TEKNİKLERİ

BİLM 637
MOBİL VE KABLOSUZ BİLİŞİM DERSİ
PROJE ÖDEVİ
Doç.Dr. Hasan Hüseyin BALIK



(Edirne-2009)

İçindekiler

1. Giriş.....	- 2 -
BÖLÜM I.....	- 3 -
1. Sayısal İletişimin Temelleri.....	- 3 -
1.1. Veri iletimi karakteristikleri:.....	- 4 -
1.2. Cps (cycles per second):.....	- 5 -
1.3. Analog iletim:.....	- 6 -
1.4. Band genişliği:.....	- 6 -
1.5. Periyot ve Dalga Boyu:.....	- 7 -
1.6. DC işaretler:.....	- 7 -
1.7. İletim kapasitesi, hız ve gecikme:.....	- 8 -
2. Tanımlar.....	- 9 -
2.1. Kodlama:.....	- 9 -
2.2. Protokol:.....	- 9 -
2.3. Paralel iletim:.....	- 10 -
2.4. Seri iletim :.....	- 10 -
2.5. Asenkron seri iletim :.....	- 10 -
2.6. Senkron seri iletim :.....	- 11 -
2.7. Isokron Seri İletim :.....	- 11 -
2.8. Gerçek Zamanlı ve Gerçek Zamanlı Olmayan İletişim:.....	- 11 -
BÖLÜM II.....	- 12 -
1. Hata Sezme ve Düzeltme Teknikleri (Error Detection and Correction).....	- 12 -
1.1. Ağ katmanına uygun tanımlanmış servis arabirimi sağlamak.....	- 12 -
1.2. İletim hatalarını gidermek (Hata Denetimi).....	- 12 -
1.3. Yavaş alıcılardaki veri taşmasını önlemek için veri akışını ayarlamak (Akış denetimi).....	- 13 -
2. Protokoller.....	- 14 -
2.1. Dur ve bekle Protokolü:.....	- 14 -
2.2. Sürekli Tekrar İstemi (Continuous RQ).....	- 16 -
2.3. Seçmeli Tekrar (Selective repeat).....	- 16 -
2.4. Çerçeve Gerile (Go-Back-N).....	- 17 -
2.5. Kayan Pencere Protokolleri.....	- 18 -
3. Hata Düzeltme Teknikleri (Error Correction).....	- 19 -
3.1. Sembol yerine koyma :.....	- 19 -
3.2. Tekrar iletim :.....	- 19 -
3.3. İleriye dönük hata düzeltme (FEC) :.....	- 20 -
4. Hata Bulma Teknikleri (Error Dedection).....	- 1 -
4.1. Artıklık:.....	- 1 -
4.2. Tam sayım kodlaması:.....	- 1 -
4.3. Yankılama: (Echoplex):.....	- 1 -
4.4. Eşlik Denetimi (Parity Check).....	- 1 -
4.5. İki boyutlu Eşlik Denetimi (2-Dimensional Parity).....	- 1 -
4.6. LRC- Boyuna Fazlalık Sınaması:(Longitudinal Redundancy Check)-Toplama Denetimi.....	- 3 -
4.7. CRC (Cyclic Redundancy Check) Çevrimli Fazlalık Sınaması:.....	- 3 -
5. CRC katarı hesaplaması :.....	- 4 -
6. ARQ (Automatic Repeat Request-Otomatik Tekrar İsteği).....	- 5 -
7. Hamming Kodlama Sistemi.....	- 7 -
7.1. Hamming Kodu.....	- 7 -
7.2. (11,7) Hamming Code Kullanımı Orneği.....	- 8 -
7.3. (7,4) Hamming Kodlama.....	- 9 -
7.4. Hamming Matrisleri.....	- 10 -
7.5. Kanal kodlama.....	- 10 -
7.6. Parity Kontrolü.....	- 11 -
7.7. Hatanın Düzeltilmesi.....	- 11 -
8. Reed-Solomon Göndermede Hata Düzeltim Kodları.....	- 12 -
9. Raptor Göndermede Hata Düzeltim Kodları.....	- 13 -
SONUÇ.....	- 13 -
KAYNAKÇA.....	- 14 -

1. Giriş

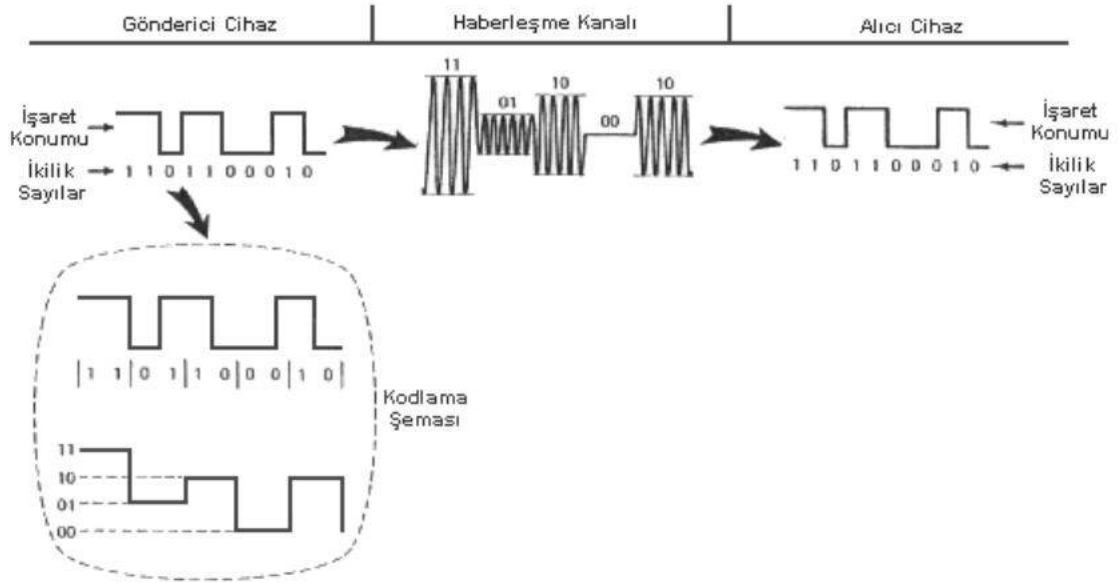
Ağda ilerleyen paketler, bazı hata risklerine sahiptir. Özellikle ağın koşulları iyi değilse paketler kaybolabilir ya da bozulabilir. Ancak bu tip hataların düzeltilmesi mümkündür. Bu çalışma ile hata düzeltme teknikleri üzerinde durulmuş, En yaygın kullanılan Hamming Kodlama Algoritması ve ARQ Teknikleri üzerine ağırlık verilmiştir. Gelişen uygulamalarla Reed-Solomon ve Raptor Göndermede Hata Düzeltim Kodlarına da kısaca değinilmiştir. Ancak bunları daha iyi yorumlayabilmek için konuyla ilgili birkaç önemli hatırlatma da çalışmaya dahil edilmiştir.

BÖLÜM I

1. Sayısal İletişimin Temelleri

Sayısal iletişim ikili tabanda (binary, 0 veya 1'ler biçiminde) sistemde kodlanmış bilgi veya verinin sistemler arasında aktarılması konularını kapsar. Bir bilgisayardaki bitler elektrik işaretinin polarizasyon seviyeleri ile gösterilirler. Bilgisayardaki saklama elemanı içindeki yüksek-seviye işareti 1'i ve alçak-seviye işareti 0'ı gösterebilir.

Bu elemanlar birlikte dizilerek belirlenmiş kodlara göre sayı ve karakterleri oluştururlar. Veri; haberleşme yolu üzerinden (örneğin telefon hattı) bilgisayar-yönlendirmeli cihazlar arasında elektrik işaretleri ve bit katarları ile iletilir. Bu elektrik işaretleri ve bit katarları harf ve karakterleri belirtir. Bazı durumlarda, veri ışık işaretleri ile gösterilir (fiber optik hatlarda). Bit dizileri kullanıcı verisini ve kontrol verisini tanımlar. Verinin bu sayısal iletişimi sırasında birçok standart ve protokol tanımlanmıştır.



Şekil 1 Verinin İletilmesi

Kontrol verisi, haberleşme ağını ve kullanıcı verisi akışını yönetmek için kullanılır. Şekil 1'de, verinin gönderici cihazdan çıkışı, haberleşme ortamından geçişi ve alıcı cihaza gelişi görülmektedir.

İkili sistemde kodlanan veri, terminaller ve çıkışlarda on tabanına çevrilerek gösterilir. Saniye başına bit (bit/sn) terimi iletim hızını belirtmek üzere kullanılır. Bu terim haberleşme yolu veya parçası üzerinden saniyede iletilen bit sayısını verir.

Örneğin 2400 bit/sn'lik bir hat, bir sayı veya karakteri belirtmek için 8-bit'lik kodlar kullanıyorsa, saniyede iletilen karakter sayısı 300 ($2400 / 8$) olur.

Haberleşme hızı genelde bit/sn oranı ile verilir.

1.1. Veri iletimi karakteristikleri:

Veri haberleşmesini anlamak için, elektriğin iletim karakteristikleri hakkında genel bir bilgiye sahip olunmalıdır. Hat kapasitesi, hata kontrol teknikleri, haberleşme yazılımı ve diğer pek çok ağ bileşeni elektriğin yapabildikleri ve sınırlamaları çerçevesinde analiz edilir ve tasarlanır.

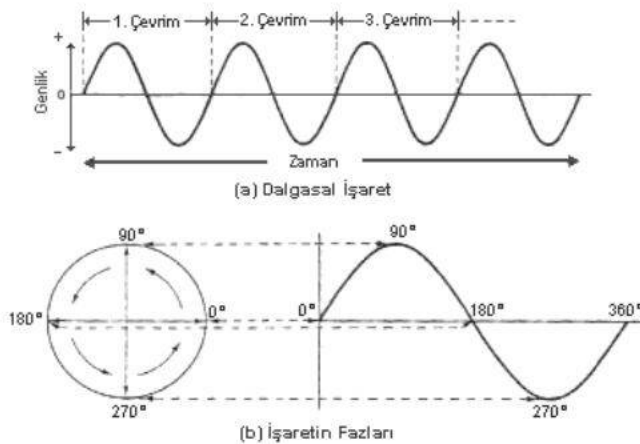
Bir işaretin iletim yolu üzerindeki hareketine **işaret yayılması** denir. Bir kablo yolu üzerinde, işaret yayılması elektrik akımı şeklindedir. Kablosuz uygulamalarda ise, havada elektromanyetik dalga olarak yayılan elektrik işaretlerinin algılanması ile başılır. İletim Yolu olarak Fiber optik kablolar kullanılmışsa veri iletimi ışık ile gerçekleştirilir. Elektrik akımı haberleşme yolu veya iletkeninin bir ucundan elektrik yükü girişi ile sağlanır.

Bilgisayar verilerinin taşınmasını sağlamak için salınım yapan işaretlerin üç parametresi değiştirilebilir : * **Genlik** * **Frekans** * **Faz**

Genlik veya gerilim kablo üzerine düşen elektrik yükü miktarı ile belirlenir. Şekil 1'de bu gerilimin ikilik konumlara (1 veya 0) bağlı olarak yüksek veya düşük olduğu durumlar görülmektedir. İşaret gücü, işaretin bir kablolu haberleşme devresinde gidebileceği veya yayılabileceği mesafeyi belirler.

Baud terimi de veri haberleşmesinde hat üzerindeki işaretin değişme oranını tanımlar. Bunu işaret hızı olarak da açıklayabiliriz. Örnek olarak, Şekil 1'deki gönderici cihaz bitleri ikili gruplar halinde toplamakta (00, 01, 10, 11) ve bunlardan her bir grup için farklı genliklerde salınım yapan dalga şekilleri elde etmektedir. Bu örnekte, bit transfer oranı baud'un (ya da işaret değişme oranının) iki katıdır. Günümüzde kullanılan modemler her bir baud için 8-bit oranlarına kadar çıkarak daha büyük bir işaret transfer kapasitesine ulaşırlar.

Şekil 1'de görülen işlemde veri katarının haberleşme yolu için değiştirilmesi veya modüle edilmesine **modülasyon** denir. İşaret aynı zamanda frekansından, başka bir deyişle belli bir zaman aralığında yaptığı tam salınım sayısından tanınır. **Frekans** saniye başına yapılan salınım sayısı ile ölçülür. Elektrik endüstrisinin tanımladığı 1 Hz birimi, saniyede bir **salınım** demektir.



Şekil 2 Salınım Yapan İşaret

1.2. Cps (cycles per second):

Genlik işaret seviyesini ve negatif veya pozitif gerilim değerini belirtirken frekans, işaret salınım oranını belirtir. İşaretin fazı, işaretin çevrimine ne kadar ileriden başladığını tanımlar.

Şekil 3 (b)'de işaretin fazı; başlangıç (0°), $\frac{1}{4}$ çevrim (90°), $\frac{1}{2}$ çevrim (180°), $\frac{3}{4}$ çevrim (270°) ve tam çevrim (360°) noktalarında gösterilmektedir. Dalga, sinüs dalgasında veya bir çemberde olduğu gibi dereceler ile de etiketlenebilir. Bir yoldaki veri işaretinin bilgi oranı kısmi olarak işaretin genlik, frekans ve fazına bağlıdır. Bilgi oranı (bit/sn), işaretin hangi sıklıkta durum değiştirdiğine bağlıdır. İşaretin genlik, frekans ve fazındaki değişiklikler hat üzerinde bir durum değişimi oluştururlar. Bu değişim 0'ı 1'e veya 1'i 0'a çevirir. İkilik 1'ler ve 0'lar, hatta bilgisayarlar arasında, veri mesajlarındaki karakter ve harfleri temsil etmek üzere kodlanırlar.

1.3. Analog iletim:

Sürekli, yani ayırık olmayan bir karakteristik gösteren işaretlere analog işaret denir. Bu şekildeki bir iletim, bilgisayarlarda kullanılan ayırık ikilik sayıların iletimi için tasarlanmamıştır. Geniş bir kullanım alanına sahip olmasının nedeni, ilk zamanlarda veri haberleşme ağları geliştirilirken analog kolaylıklar sağlayan telefon sisteminin hâlihazırda mevcut olmasıdır.

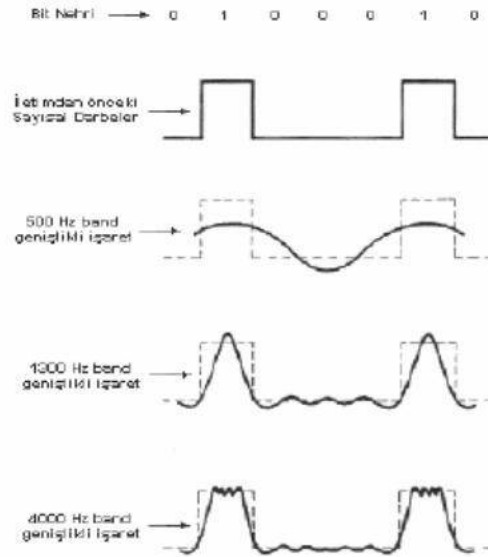
1.4. Band genişliği:

Bir haberleşme hattının taşıyabildiği frekans aralığı, hattın band genişliği olarak tanımlanır. Haberleşme hattının kapasitesini (bit/sn), hattın band genişliği belirlediği için çok önemlidir. Daha büyük band genişliği kullanılarak daha iyi bir veri iletim oranı sağlanabilir, daha yüksek hat kapasitesi demektir. Band genişliğinin etkileri Shanonon, Fourier ve Nyquist gibi bilim adamları tarafından saptanmıştır.

Yaklaşık Frekans	İsim	Kullanım Yeri
10^3	----	Telefon sesi Frekansları
10^4	VLF	Yüksek hızlı modemlerdeki ses frekansları
10^5	LF	Koaksiyel denizaltı kabloları, bazı yüksek hızlı batch veri transferleri
10^6	MF	Kara koaksiyel kabloları, AM radyo yayınları
10^7	HF	Kara koaksiyel kabloları, kısa dalga radyo yayınları
10^8	VHF	Kara koaksiyel kabloları, VHF Ses ve TV yayınları
10^9	UHF	UHF TV yayınları
10^{10}	SHF	Kısa-link Dalga kılavuzları; mikrodalga yayın
10^{11}	EHF	Sarmal dalga Kılavuzları
10^{12}	----	Kızılötesi iletim
10^{13}	----	Kızılötesi iletim
10^{14}	----	Fiber optikler, görünür ışık

10^{15}	----	Fiber optikler, morötesi
$10^{19} - 10^{23}$	----	X-ışınları ve gamma ışınları

Tablo 1 Frekans Spektrumu



Şekil 3 Band Genişliğinin Etkisi

1.5. Periyot ve Dalga Boyu:

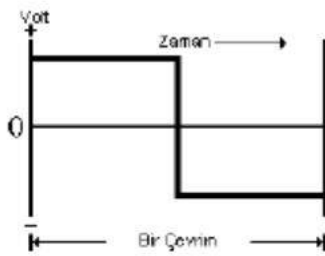
Bir çevrim için gereken süreye **periyot** denir. Örneğin, 2400 Hz'deki bir işaret, 0.000416 sn'lik bir çevrim periyoduna sahiptir ($1 \text{ sn} / 2400 = 0.000416 \text{ sn}$).

Periyot (T), $1/F$ olarak hesaplanır ki burada F frekanstır (Hz).

$$WL = S / F \text{ (1-1)}$$

WL: Dalga boyu S: İşaretin yayılma hızı

İşaretin dalga boyu ağ cihazı seçiminde, protokol tasarımında ve cevap-zamanı analizinde çok önemlidir.



Şekil 4 Kare Dalga

Diğer dalga şekilleri: Çok yaygın olan diğer bir yaklaşım da ikilik değerleri simetrik kare dalga kullanarak iletmektir (Şekil 4).

Kare dalga pozitif polarizasyondan negatif polarizasyona anlık sürede geçen bir gerilimi gösterir. Kare dalga, sayısal veri iletimi için mükemmel bir şekildir, çünkü ikilik durumlar olan 1 ve 0'ları, pozitif ve negatif değerler ile gösterebilir.

1.6. DC işaretler:

DC işaretler, yalnızca ayırık 1'ler ve 0'ları gösterebilen simetrik kare dalgaya benzerler. Ancak DC iletici, salınım yapan dalga şekli yerine, açık-kapalı elektrik enerjisi darbelerini kullanır. DC işaret olduğu gibi iletilir, üzerine başka işaret veya frekans bindirilmez (modüle edilmez). Bir AC işaret, başka frekanslar tarafından taşınmak üzere (yeterlilik, hız ve iletim mesafesi

etkenleri yüzünden) yeniden şekillendirilir. Bu işaret şekillendirilmesi **modülasyon** olarak anılır. Birçok sistem sınırlı bir mesafede çalıştığı için daha güçlü ve daha pahalı olan AC iletme ihtiyaç duymaz, bunun yerine DC işaretlemeyi kullanır. Sinüzoidal dalga şekli, simetrik kare dalga gibi, uzak mesafe veri haberleşme hatları için gerekli iletim tipidir. Sayısal bit katarları hem DC hem de AC işaretlerle taşınabildiği halde, uzak mesafe iletiminde AC işaretler kullanılır. Telgraf, DC işaretlemeye iyi bir örnektir.

1.7. İletim kapasitesi, hız ve gecikme:

Bir haberleşme sisteminin iletim sığası (kapasitesi) bit/sn olarak gösterilir. Bilgisayar üzerinde çalışan kullanıcı uygulamaları için cevap süresi ve veri akışı, sistemin kapasitesine bağlıdır. Örneğin; 4800 bit/sn'lik hat, 2400 bit/sn'lik hattın iki katı sığaya sahiptir. Bu da arttırılmış bir akış ve daha kısa bir cevap-süresi sağlar.

Telefon ağı ses taşımak için üretilmiştir ve düşük band genişlikli işaretlerle çalışır. Band genişliği, işaret gücü ve iletken üzerindeki gürültü, iletim sığasını sınırlayan etkenlerdir. Gerçekten de arttırılmış bir işaret gücü hat sığasını artırır ve aynı zamanda daha uzak mesafelere işaret yayılımı yapılabilmesini sağlar. Ancak aşırı güç, sistemdeki parçalara zarar verebilir ve/veya ekonomik olarak karşılanamayabilir. Hattaki gürültü problemi hattın tabiatında olan ve ortadan kaldırılamayan bir problemdir. Gürültü (Termal, Gaussian, beyaz veya arka plan gürültüsü), elektronların iletken üzerindeki sabit, rastgele hareketlerinden meydana gelir ve kanal sığasına bir sınırlama getirir.

Gürültü gücü, band genişliği ile doğru orantılıdır, yani band genişliğini arttırmak ek gürültüye yol açacaktır. Eklenen gürültüyü azaltmak için süzme olarak bilinen bir elektronik teknik kullanılır. Haberleşmenin temel kanunlarından biri **Shannon Kanunu**'dur.

Shannon bir iletim yolunun sığasını aşağıdaki formülle göstermiştir:

$$C = W \log_2 (1+S/N) \quad (1-2)$$

C = bit/sn olarak maksimum sığa,

W = Band genişliği,

S/N = İşaret gücünün (S) gürültü gücüne (N) oranı.

Bir kanal üzerinden gönderilebilecek maksimum bilgi miktarı '**kanal sığası**' olarak adlandırılır.

Formül incelendiğinde W'yi arttırmanın, işaret gücünü arttırmanın veya gürültü seviyelerini düşürmenin müsaade edilen bit/sn oranını arttıracığı görülebilmektedir. 1000'e 1 S/N oranı olan bir ses-sınıfı hattın müsaade edebileceği maksimum sığa 25900 bit/sn'dir.

Shannon kanunu ile bulunan teorik limit, pratikte daha düşüktür. İletimde oluşan hatalar nedeniyle Shannon kanunu tam sınırları ile kullanılamaz.

İşaretin iletim veya yayılma gecikmesi, mühendis ve kullanıcılar için göz önüne alınacak bir başka konudur. **Yayılma gecikmesi**; kullanılan devrenin türü, alıcı ile verici arasındaki ara

noktaların sayısı ve bu noktaların türü gibi çeşitli etkenlere bağlıdır. Yaklaşık olarak koaksiyel kablo ve mikrodalga yolları üzerindeki iletim, 130,000 mil/sn hızındadır. Ancak, işaretin hızı frekansa bağlı olarak değişir.

Örneğin, tipik bir telefon hattı (19 gauge) 10 kHz'de yaklaşık 110,000 mil/sn hızında ve 50 kHz'de 125,000 mil/sn hızında çalışmaktadır. Frekans ve kablonun belirli elektriksel karakteristikleri nedeniyle bu hızlar hattın teorik hızı olan 186,000 mil/sn'den daha yavaş olmaktadır.

Mesaj ağ üzerinde ara istasyonlara girip çıkarken ek ve önemli gecikmeler meydana gelebilir. Ancak öncelikli iletim gecikmesi, hattın kendinden kaynaklanmaktadır. Switchler ve bilgisayarlar gibi ara parçalar gecikmeye sebep verseler de, genelde çok yüksek hızlarda çalışırlar (nanosaniyeler veya saniyenin milyarda biri mertebelerinde). Tabii ki, bu istasyonların mesajları disk veya teyplerine saklamaları durumunda göz önüne alınması gereken ek gecikmeler meydana gelebilir.

2. Tanımlar

Sayısal iletişimi daha iyi anlamak için bu konuda kullanılan bazı terimlerin ve kavramların anlamlarına bilmemiz gerekir.

2.1. Kodlama:

Verilerin sayısal tabanda gösterimi için kullanılan yöntem **kodlama** denir. Örneğin verinin gösterilimi 32 farklı parçanın değişik kombinasyonlarıyla yapılıyorsa her bir parça için 5 bit kullanılır. Yani 5 bitlik parçalar birleşerek veriyi oluştururlar. Sayısal iletişimde birçok kodlama biçimi kullanılır. Metinlerin iletimi için ASCII kodu kullanılırken, görüntü iletimi için görüntünün doğrudan bit haritası kullanılır. ANSI tarafından tanımlanan ASCII kodlamada her bir karakter için 7 bit kullanılır.

N tane bit ile kodlanabilen sembol sayısı 2^n 'dir. Bu nedenle ASCII'de 128 tane karakter vardır. Bu koda hata sezme amacıyla 8. bit de eklenebilir. Ayrıca IBM tarafından (1962 yılında) tanımlanan EBCDIC kodlamada 8 bit kullanılır.

2.2. Protokol:

Veri formatlarını ve iletimin zamanlamasını düzenleyen, alıcının ve vericinin; kullanacağı işaretler, veri formatları ve iletim yöntemleri gibi belirledikleri kurallar bütününe **protokol** denir. İki veya daha fazla bilgisayarın birlikte çalışabilmesi için aynı protokolü kullanmaları zorunludur. Zayıf bir protokol veri transferini yavaşlatabilir, fakat standart protokolleri izleyen yazılımlar farklı sistemler arasında iletişim kurabilirler. Örneğin: TCP/IP protokolü

değişik mimarilere ve işletim sistemlerine sahip bilgisayarlar arasında veri transferine izin verir. Protokolün anahtar elemanları **sözdizimi, semantik ve zamanlamadır**.

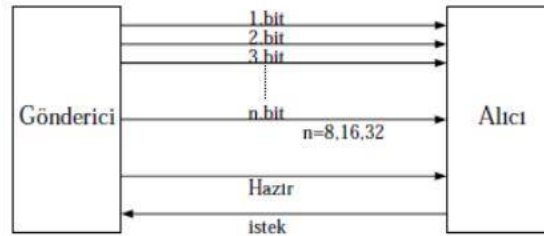
Sözdizimi kullanılacak sinyalin seviyesini ve gönderilecek verinin biçimini belirtir.

Semantik makineler arasında koordinasyonu sağlamak için gereken bilgi yapısını içerir.

Zamanlama hız ayarlanması yapar.

2.3. Paralel İletim:

Bu tip iletimde n bitlik verideki her bit ayrı bir yoldan iletilir. İletim sırasında, göndericinin alıcıya verideki bitleri yola çıkardığını belirten veri hazır (data ready) mesajı yollaması gerekir. Alıcının da göndericiye veri alabileceğine dair istek (demand) mesajı göndermesi gerekir. Paralel iletim genellikle birbirine yakın cihazlar arasındaki (1-2 metre) iletimlerde kullanılır. Örneğin bilgisayar ile yazıcı arasındaki iletimde paralel iletim sıkça kullanılır.



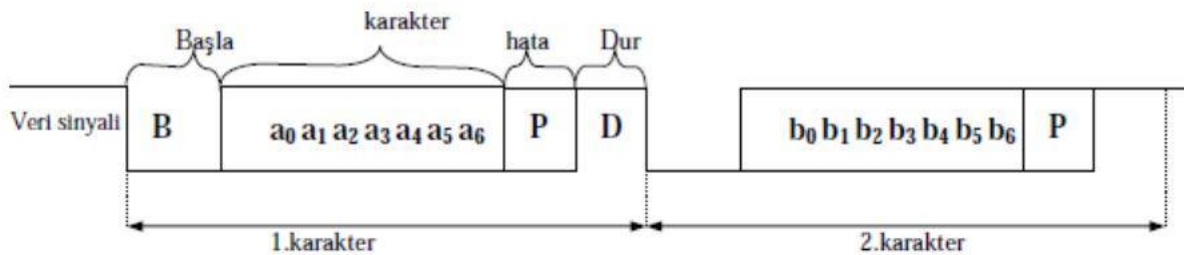
Şekil 5 Paralel iletim

2.4. Seri İletim :

Seri iletimde bilgi sadece bir veri yoluyla aktarılır. N bitlik veri sıra ile aktarılır. İşaret aktarım hızı baud birimidir. **Baud**, birim zamanda aktarılan ayrık işaretlerin sayısıdır. Bilgisayar ağlarında iletim seri iletimle gerçekleşir. Seri iletişim zamanlama bakımından asenkron, senkron, isokron olmak üzere üçe ayrılır.

2.5. Asenkron seri iletim :

Gönderici ve alıcının ayrı saatler kullandıkları seri iletim şeklidir. 1 bit için ayrılan süre kullanılan iletim saat periyodunun n katı olur. Bu iletimde gönderilecek bilgi karakter denilen bloklara ayrılır. Bir blok 7 veya 8 bittir. Karakterin başına özel tanımlanan başla biti getirilir. Karakterden sonra ise istenilirse hata sezme biti getirilebilir. En sona dur biti getirilmelidir.

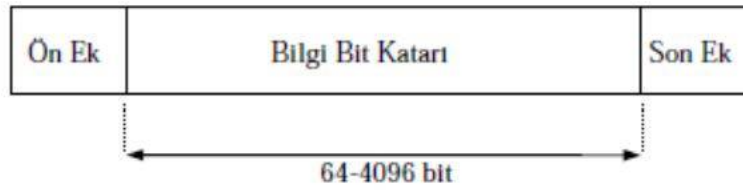


Şekil 6 Asenkron seri iletim

Şekil 6'da her karakter başla biti ile başlar böylece alıcı karakteri hemen alır karakterin bitişini ise dur biti belirler. Yeni bir karakter gelene kadar dur seviyesi muhafaza edilir. Hata sezme için kullanılan bit çift veya tek eşlik kontrolü yapar burada gönderilen karakterdeki 1'lerin sayısı tekse 1 olur çiftse 0 olur. Gönderici ve alıcının saat frekansları arasında %5 kayıklık bile olsa her karakterin başında alıcı veriye yeniden sekronize olduğundan kayıklık hatası hissedilmez.

2.6. Senkron seri iletim :

Senkron iletimde karakterlerin başına başla sonuna dur biti konulmaz. Gönderici alıcıya saat işaretini veri ile birleştirerek gönderir. Alıcı gelen bu saat frekansını kendi içerisinde faz kilitleme sistemiyle bu frekansa eşit bir frekans oluşturur.



Şekil 7 Senkron Seri İletim

Sekron iletimde bilginin bir katarının başına ve sonuna özel kodlar konularak alıcının bilginin başını ve sonunu anlaması sağlanır. Ard arda bilgi katarlarının gönderilmesinde aradaki son ekler kullanılmayabilir. Ön ve son ekler bilgi içerisinde olmayacak şekilde kodlanmış olmalıdır.

2.7. Isokron Seri İletim :

Sekron iletimin bir türevi gibidir. Bu sistemde bilgisayarların haberleşmesi periyodik olarak yapılır. Bu periyotlar ile iletimde kullanılan yol kapasitesi garanti altında tutulur. Örneğin : her 100 s de 200 bit aktarılacak gibi bir gereksinim belirtilir ve kesin sağlanmalıdır. Bu iletim genellikle gerçek zamanlı uygulamalar için kullanılır. (ses video aktarımı vb.)

Genelde iletim hattı üç tipte incelenir.

*Simplex * Half-duplex * Full-duplex.

1- Simpleks : Bir gönderici ve bir alıcının olduğu sistemlerdir. İletişim sadece göndericinin yayın yapmasına izin verir alıcı göndericiden gelen bilgiyi dinleyebilir. Alıcı göndericiye mesaj gönderemez. Örneğin; TV yayınlarını alıcı izlerken, aynı hat üzerinden yanıt gönderemez.

2- Half Dupleks: Hattı ilk olarak gönderici kullanır. Cevap istendiği zaman, karşı taraf yani alıcı hattı kullanır. Örnek olarak; telsiz konuşmalarını gösterebiliriz.

3- Full Dupleks: Bir yandan hatta veriler gönderilirken bir yandan da hattan cevap niteliği taşıyan veriler alınabilir. Mesela telefon görüşmelerinde, aynı hat hem alıcı hem de verici tarafından kullanılır.

2.8. Gerçek Zamanlı ve Gerçek Zamanlı Olmayan İletişim:

Gerçek zamanlı iletişimde karşı sistemden hızlı bir yanıt beklenir. Bu yanıt gelmeden iletişimde ilerleme yapılmaz. Alıcıda belli bir süre değerlendirilmezse bilgi gönderme bir işe yaramayabilir. Elektronik bankacılık ve video konferans iletişimi örnek olarak verilebilir.

Gerçek zamanlı olmayan iletişimde ise hızlı bir yanıt beklenmez, verinin doğruluğu ve güvenliği sağlanması gerekir. Örnek olarak elektronik posta, dosya aktarımı verilebilir.

BÖLÜM II

1. Hata Sezme ve Düzeltme Teknikleri (Error Detection and Correction)

Verinin iletimi sırasında bazı bitlerin bozulması söz konusu olabilir. Bitlerin bozulma ihtimali az da olsa ihmal edilemez zira bir bit bile bozulsa verinin tamamı anlamsız hale gelebilir. Bu sebeple iletişim sırasında bozulma olup olmadığının anlaşılması için hata sezme teknikleri kullanılır. Eğer bozulma olmuşsa hata düzeltme teknikleri kullanılarak hata giderilmeye çalışılır. Hata giderilemezse veri paketi tekrar gönderilir.

Hata Sezme ve Düzeltme Teknikleri OSI Başvuru Modelinde Veri Bağı Katmanının işlevlerinden biridir. Veri Bağı Katmanı, yerel sistemler arasındaki topoloji ve haberleşme özelliklerini tanımlar. Paket başlıklarını ve checksum dizilerini hazırlar. Datagramları çerçevelere paketler. Hataları anlar. Veri akışını düzenler. Donanım adreslerini dönüştürür. Switch ve Bridge cihazları bu katmanda çalışır.

Gönderilecek bilginin hatalara bağımsız bir yapıda lojik işaretlere dönüştürülmesi alıcıda hataların sezilmesi düzeltilemiyorsa doğrusunun elde edilmesi için göndericinin uyarılması gibi işlevleri vardır.

Gönderilen ve alınan lojik işaret bloklarına **Çerçeve** denir. Çerçevelerin içerdiği bit sayısının alt ve üst sınırları standartlarla belirlenmiştir. Genellikle değişken uzunluktadırlar.

Veri bağlantı katmanının işlevleri aşağıdaki şekilde sınıflandırılabilir.

1.1. Ağ katmanına uygun tanımlanmış servis arabirimi sağlamak

Genel yaklaşım; dizilerini çerçeveler haline getirmek ve her bir çerçeve için sağlama (checksum) bilgisini oluşturmaktır. Bir bilgi varış tarafına iletdikten sonra sağlama bilgisi hesaplanır. Eğer yeni hesaplanan değer gelenden farklı ise, veri bağlantı katmanı bir hata olduğu sonucuna varır.

Birçok sağlama bilgisi hesap yöntemi olmakla birlikte başlıca yöntemler Karakter sayma, karakter doldurulmuş bayrak baytları, bit doldurulmuş başlama ve bitiş bayrakları, fiziksel katman kodlama çığnemeleri

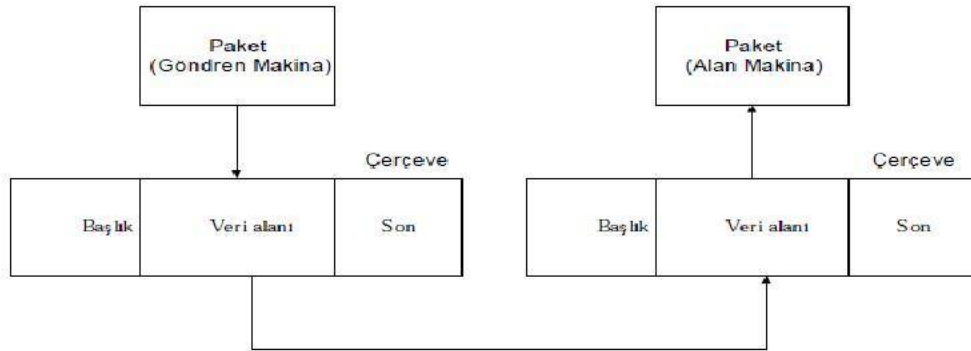
1.2. İletim hatalarını gidermek (Hata Denetimi)

Veri iletimi sırasında göndericinin gönderdiği verinin sorunsuz olarak alındığına ilişkin geri besleme bilgisi alması, eğer veri hatalı alınmış ise bununla ilgili düzeltme önlemlerini uygulamak veri bağlantı katmanının diğer görevidir. Bu işlem bağlantısız iletimde bir zamanlayıcı kullanarak, bağlantı temelli protokollerde ise geri besleme bilgisi yollanarak sağlanır. Eğer bir çerçeve herhangi bir nedenle bozulmuş veya kaybolmuş ise en kolay çözüm onu tekrar yollamaktır. Ancak birkaç kere aynı çerçeve yollanırsa bunun sadece bir tanesi veri bağlantı katmanı tarafından ağ katmanına iletilmelidir.

1.3. Yavaş alıcılardaki veri taşmasını önlemek için veri akışını ayarlamak (Akış denetimi)

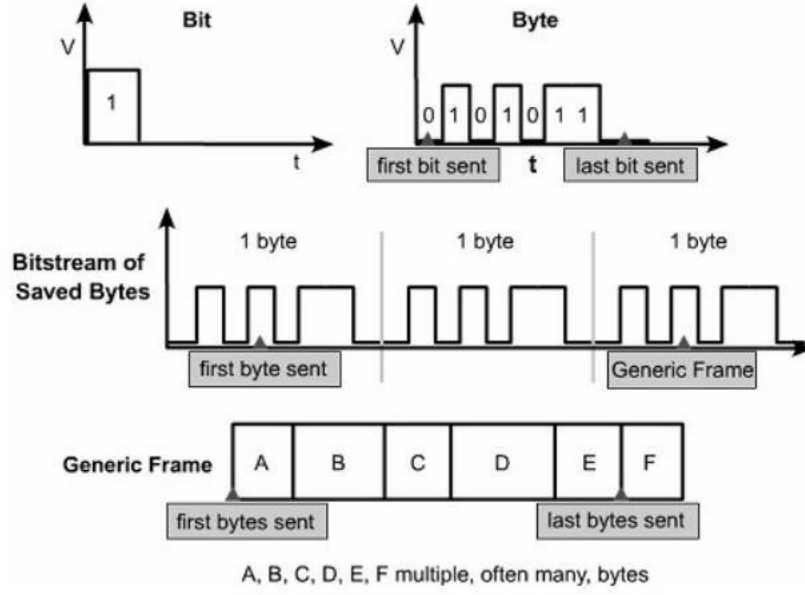
Eğer alıcıdan daha hızlı olan gönderici sürekli veri göndermek ister, alıcı da bu hıza yetişemez ise veri bağlantı katmanı tarafından akış denetimi yapılması gerekir. Bu denetim için **geri besleme tabanlı akış denetimi** (alıcı göndericiye daha fazla veri göndermesi için geri besleme bilgisi gönderir) ve **hız-tabanlı akış denetimi** (geri besleme olmaksızın göndericinin hızı sınırlanır) şeklinde iki yöntem kullanılır.

Veri bağlantı katmanı bu görevleri yerine getirebilmek için ağ katmanından veri paketlerini alır ve iletim için onları çerçeveler halinde paketler (kapsül içine alır) Her çerçeve bir başlık, bir veri alanı ve bir son bilgisinde oluşur. (Şekil 8)



Şekil 8 Veri paketi ve Çerçeve arasındaki ilişki

Veri bağı katmanı fiziksel katmandan aldığı veriyi bir üst katmana iletirken bu verinin doğruluğunu kontrolünü gerçekleştirir. Bunu her frame sonuna bir kontrol dizisi ekleyerek sağlar. (FCS Frame Check Sequence)



Şekil 9 Çerçeve (Frame)

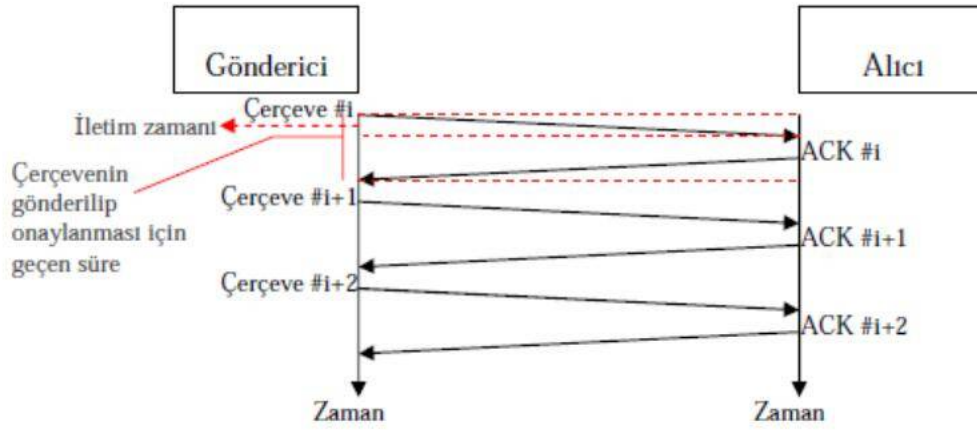
FCS çerçeve üzerinde çeşitli matematiksel işlemler yapar ve sonucu çerçeve içerisinde FCS alanına kaydeder. Alıcı cihaz aynı matematiksel işlemleri tamamladığında elde ettiği değeri aldığı çerçeve üzerindeki FCS alanı ile karşılaştırır. Bu iki değer eşit değilse iletim esnasında bazı bit hataları olduğu tespit edilir.

Hata olduğu anlaşıldığında gelen paket işlenmez. Hata giderme ise, protokolün veri kaybı olduğunda harekete geçerek verinin yeniden istenmesini sağladığı anlamına gelir. Bir çok protokol hata giderme mekanizmaları sunarlar. İletilen veri etiketlenir ya da numaralandırılır. Veri alındıktan sonra alıcı hata tespit ederse hatalı olan veri paketini bu numara ile yeniden ister.

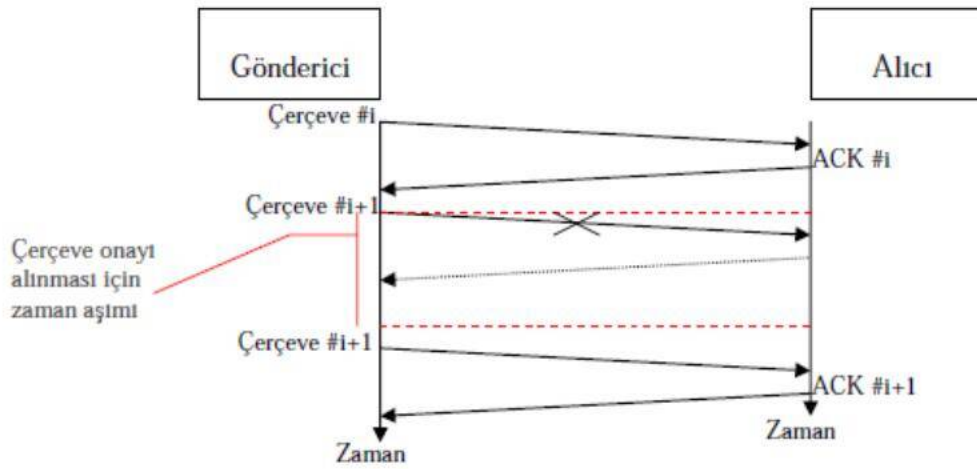
2. Protokoller

2.1. Dur ve bekle Protokolü:

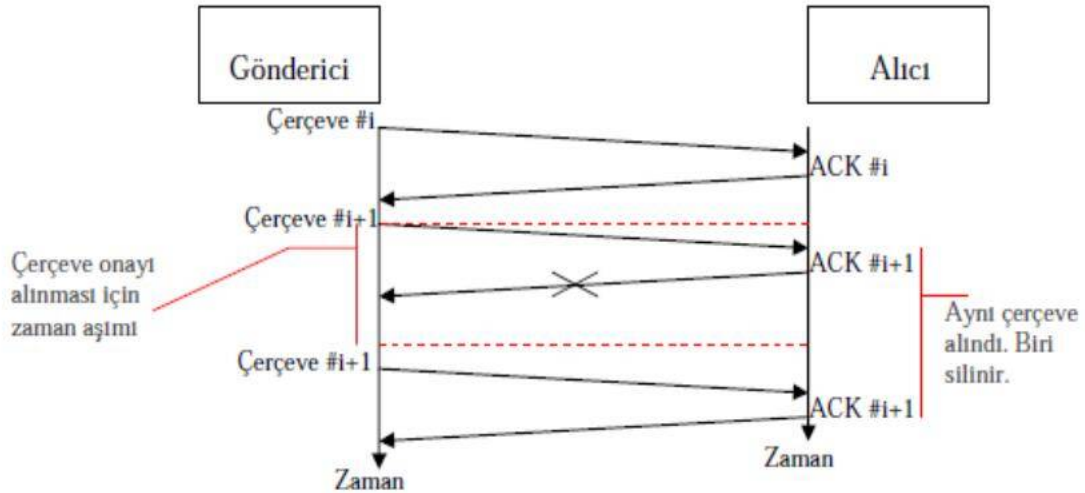
Beklemeli tekrar isteme hata denetimi, veri çerçevelerinin gönderen ile alıcı arasında güvenilir bir biçimde iletilmesinden sorumludur. Veri çerçevesi alıcıya gönderildikten sonra doğru olarak gönderilip gönderilmediği bilgisi geri dönene kadar beklenir. Veri hatasız alındığı onaylandıktan sonra bir sonraki çerçeve gönderilir.



Şekil 10 Normal veri akışı



Şekil 11 Gönderilen Çerçevenin Bozulması

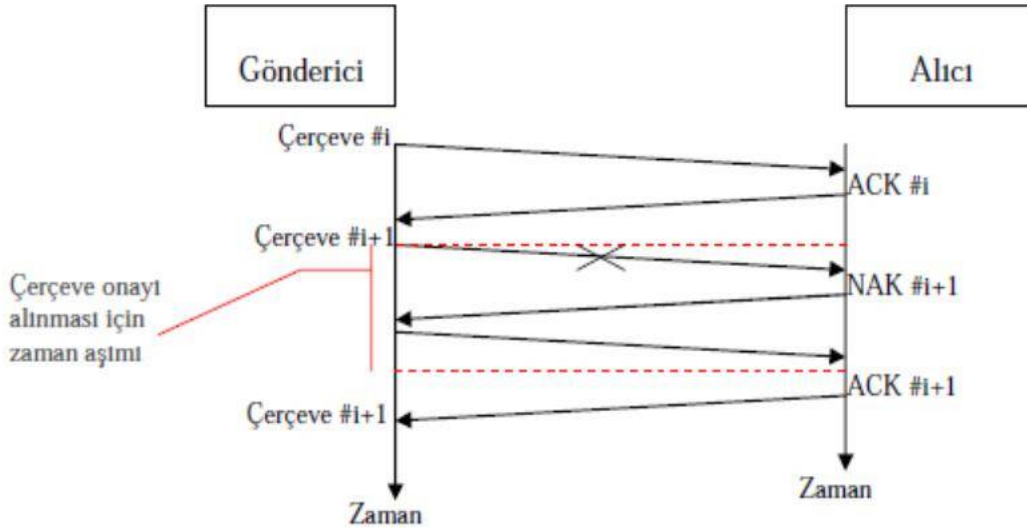


Şekil 12 ACK çerçevesinin bozulması

Veri iletim sırasında gönderilen çerçevelerden biri iletim sırasında bozulursa alıcı taraftan bir onay cevabı gelmeyecektir. Onay alınması için daha önceden belirlenen çerçeve iletim zamanı içerisinde onay alınmadığından zaman aşımı işlemi devreye girecektir. Son gönderilen veri tekrar gönderilir.

Bu işlem onay cevabının bozulması durumunda da gerçekleşir. Fakat alıcı tarafa bu kez aynı çerçeveden iki tane alınmış olur. Bunlardan biri silinir. Bu şekilde veri bütünlüğü sağlanır.

Dur bekle protokolünde bir başka durum da hatalı verilerin de bildirilmesidir. Alıcı tarafta alınan çerçevede bozulma tespit edildiğinde NAK hatalı çerçeve bilgisini gönderir. Gönderici taraf gelen bu hatalı çerçeveyi tekrar gönderir. NAK hatalı çerçeve bilgisinin kullanılması zaman aşımı süresini beklemeksizin iletişimi devam ettirilir ve iletişim hattı daha verimli kullanılır.

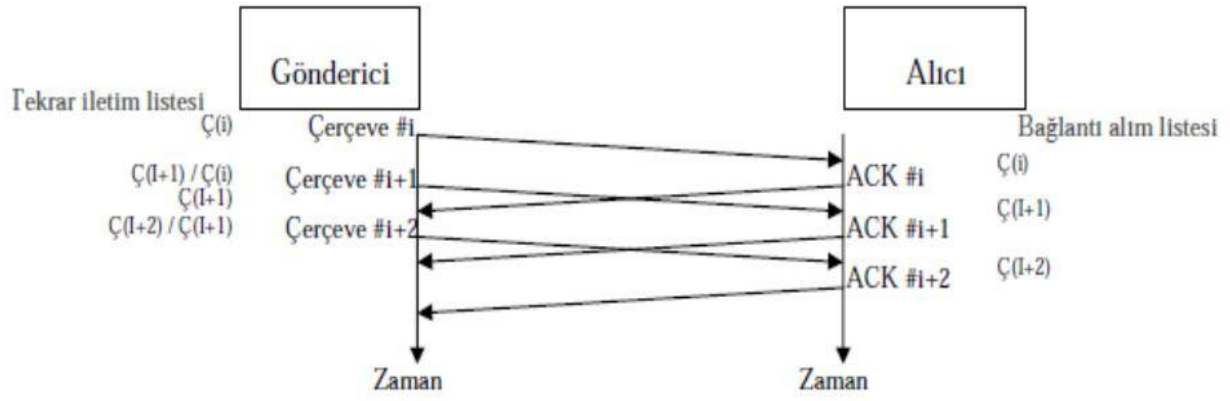


Şekil 13 Gönderilen Çerçevenin Bozulması ve NAK gönderilmesi

2.2. Sürekli Tekrar İstemi (Continuous RQ)

Sürekli tekrar isteminde gönderici alıcı taraftan ACK çerçevelerinin gelmesini beklemeden art arda veri çerçevelerini gönderir. Kaynak birden çok çerçevenin doğrulamasını bekler bu durumda çerçevenin tekrar gönderilme olasılığına karşı kopyasını saklar. ACK çerçevesi alındığında ise tekrar iletim listesinden ACK de belirtilen çerçeve silinir. Alıcı tarafta sırası farklı olarak gelebilir.

Alınan çerçeveler bağlantı alım listesinde tutulur. Sürekli tekrar istemi ve hata denetim işlemi çift yönlü bir iletişim gerektirir. Gönderici tarafta ACK çerçevelerinin alınmasını beklemeden gönderilen çerçeve sayısı tekrar iletim listesinin büyüklüğü ile sınırlıdır.

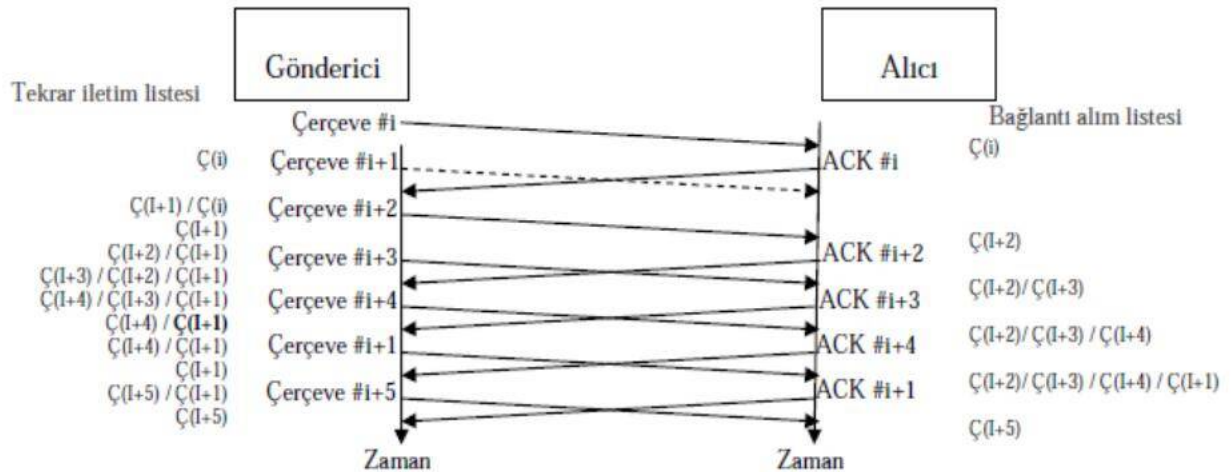


Şekil 14 Normal Veri Akışı

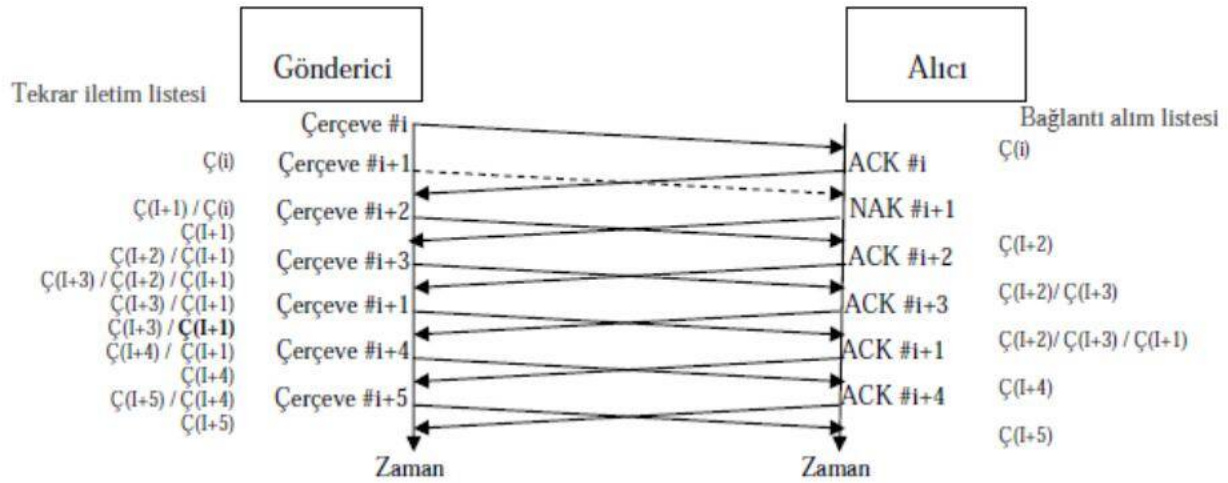
Veri iletişiminde bir hata oluştuğunda hatanın giderilmesi için seçmeli tekrar (selective repeat) veya N çerçeve gerile (go-back-N) yöntemden biri kullanılır. Seçmeli tekrar yönteminde alıcı taraf veri dizisindeki hatalı çerçeveyi saptar ve yalnızca hatalı çerçevelerin tekrar iletilmesini sağlar. N çerçeve gerile yönteminde ise alıcı sıra dışında bir çerçevenin alındığını saptar ve doğrulanmış son çerçeveden başlayarak tekrar iletim listesinde bekleyen tüm doğrulanmamış çerçeveleri yeniden ister.

2.3. Seçmeli Tekrar (Selective repeat)

Seçmeli tekrar işlemi iki şekilde yürütülür. Belirsiz tekrar gönderim ve belirli tekrar gönderim. Belirsiz tekrar gönderimde alıcı hatasız alınan çerçeveleri doğrular ve gönderici ACK çerçevelerini kontrol ederek eksik çerçeve olup olmadığına karar verir. Belirgin istekte ise alıcı sıralamada eksik olan bir çerçeve için özel bir olumsuz doğrulama gönderir. Her iki durumda da çerçeve sıra dışı alınırsa alıcı bunları bir sonraki çerçeve gelinceye kadar bağlantı alım listesinde tutar.



Şekil 15 Çerçevenin bozulma durumu ve belirsiz tekrar gönderim



Şekil 16 Çerçevenin bozulma durumu ve belirgin tekrar gönderim

2.4. Çerçeve Gerile (Go-Back-N)

Alıcı taraf sıra dışı bir çerçeve aldığı zaman belirtilen çerçeve numarasından başlayarak çerçevelerin yeniden iletilmesini ister. Bunu özel bir olumsuz doğrulama çerçevesi göndererek yapar. Alıcı taraftan hatalı olan çerçeve istendiği zaman gönderici tekrar iletim listesinden bu çerçeveden sonrakileri gönderir.

Göndericiye herhangi bir NAK hata mesajı gelmediğinde veri iletimine devam edilir. Gönderici ACK $i+2$ doğrulama mesajını aldığı anda $i+2$ ve önceki verilerin doğru olduğunu saptar ve iletim tablosundan bunları çıkarır. N Çerçeve gerile stratejisi doğru çerçeve sıralamasının korunduğu ve iletişim için tampon belleğin en aza indirildiği gösterir. Ancak doğru olarak gönderilen çerçevelerin bazıları tekrar iletildiğinden iletişim ortamını seçmeli tekrar iletim yöntemine göre daha fazla kullanır.

2.5. Kayan Pencere Protokolleri

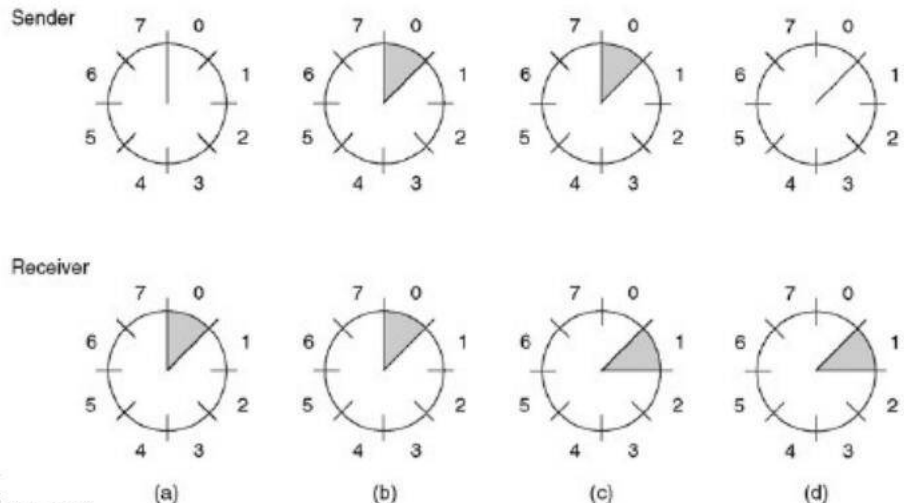
Gönderici ve alıcı arasındaki çerçeve iletim durumları:

(a) başlangıç

(b) ilk çerçeve gönderildi

(c) ilk çerçeve alındı.

(d) ilk ACK alındı.

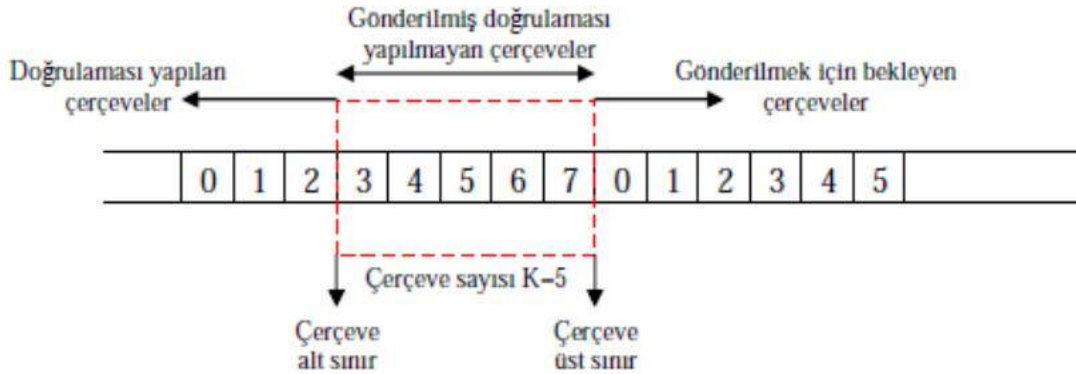


Şekil 17 Şekil 23 Gönderici ve Alıcı çerçeve durumları

Kayan pencere veri bağlantılarında çerçeve akışını denetlemek için kullanılır. Sürekli tekrar istemi protokolünde gönderici bir doğrulama olmadan sürekli veri çerçevelerini yollayabilir. Fakat veri iletişimi gönderici ile alıcı arasındaki bağlantı hızları farklı ise bu tampon depolamada sorunlar çıkaracaktır. Bu yüzden ek bir denetim yapılır.

Kayan pencere gönderici tarafından doğrulama olmadan gönderilebilecek çerçeve sayısına bir kısıtlama getirir. Bu beklemeli tekrar istemine benzer. Yani gönderici tekrar iletim listesindeki doğrulanmamış çerçevelerin sayısını kontrol eder. Alıcı taraf gelen çerçeveleri yeterli hızda işleyemiyorsa doğrulama mesajlarını durdurur. Gönderici doğrulama mesajlarını alamadığında tampon bellekte belirli sayıda çerçeveyi tuttukten sonra iletimi durdurur.

Bu yöntem kullanılırken iletimin için kullanılacak tampon bellekteki çerçeve sayısı için bir sınır belirlenir. Bu K gönderim çerçevesidir. K'yı 1 olarak alırsak bu işlem beklemeli tekrar istemi gibi çalışır.



Şekil 18 Çerçeve sayısı 5 olan kayan pencere yöntemi

Çerçevelerin iletilmesi işleminde üst pencere sınırı bir arttırılır. Alıcı taraftan çerçeveler doğrulandıkça alt pencere sınırı bir arttırılır. Pencere boyu daha önceden belirlenir ve iletim sırasında bu sınır korunur. Pencere boyu $K=5$ seçilmişse ve üst pencere ile alt pencere sınırı arasında 5 çerçeve varsa iletim durdurulur. Hatasız iletim olduğu sürece belirlediğimiz pencere iletim verisi üzerinde sürekli kayarak ilerler.

Veri iletiminde iki tip hata oluşabilir:

Patlama (burst) ve Rastgele (random) hata.

Patlama hatasında alıcıya bir süre anlamsız bilgi gelir ve bu süre içerisinde ortamdan geçen tüm bitler bozulabilir. Rastgele hata ise ortamdaki gürültü sebebiyle veri içindeki rastgele bir bitin bozulması durumudur.

Veri iletiminde 2 nedenle hata oluşabilir:

Bit Hataları : Dijital verinin ağ üzerinden taşınması esnasında maruz kalacağı bazı dışsal faktörler (örneğin: elektromanyetik dalgalar, ses vb.) dijital veriyi bozabilir. Bu etkenler verinin bit bazında bozulmasına neden olabilir. Alıcı bu tip bozulmuş veriyi tanıyamaz.

Paket Kayıpları : IP tabanlı ağlar gibi, güvenilir olmayan (TCP verinin güvenli bir şekilde iletilip iletilmediğini garanti etmez) ağlarda, paket gönderim esnasında kaybolabilir. Özellikle gerçek zamanlı verilerde gecikme faktörü önemlidir. Uzun süre geciken veriler “kayıp” olarak nitelendirilir. Ağda bulunan bir düğüm noktasının kapasitesinin “ağ tıkanıklığı” nedeniyle taşması da ağda ilerleyen paketlerin kaybolmasına neden olabilir.

3. Hata Düzeltme Teknikleri (Error Correction)

En çok kullanılan üç hata düzeltme yöntemi vardır. Bunlar

3.1. Sembol yerine koyma :

Alıcı tarafta veriyi analiz edecek ve verinin doğruluğu hakkında karar verecek bir insanın söz konusu olduğu durumlarda kullanılmak üzere tasarlanmıştır. Eğer bir karakter yanlış alınmışsa onun yerine, kullanılmayan bir sembol koyulur. Eğer operatör; hatalı karakterin doğru karşılığını anlayamazsa tekrar iletimi gerekir. Yani sembol yerine koyma, seçici bir tekrar iletim biçimidir.

3.2. Tekrar iletmeye :

Bir mesaj hatalı olarak alındığında, alıcı terminali otomatik olarak tüm mesajın tekrar iletimini gönderme terminalinden istemektedir. Tekrar iletime genellikle ARQ denir. ARQ, *tekrar iletim için otomatik istek* anlamına gelen eski bir radyo iletişim terimidir. ARQ muhtemelen en güvenilir hata düzeltme yöntemidir, ancak her zaman en verimli yöntem değildir. İletim ortamlarının yol açtığı hatalar tek bir karakterin değil, tüm mesaj birimin tekrar iletimini gerektirir. Eğer kısa mesajlar kullanılırsa, iletim sırasında bir arızanın oluşma olasılığı çok azdır. Ancak, kısa mesajlar uzun mesajlardan daha çok alındı bildirimleri ve hat çevrimi gerektirir. Hata düzeltmede ARQ kullanıldığında ideal mesaj boyutunun 256 – 512 karakter arası mesaj blokları olduğu istatistiksel olarak görülmüştür.

3.3. İleriye dönük hata düzeltme (FEC) :

Hatalı veriyi tekrar iletmeksizin alıcı tarafta hataları bulan ve düzelten tek hata düzeltme yöntemidir. FEC’de; bitler, mesaja iletilmeden önce eklenir. Yaygın olarak kullanılan bir hata düzeltme kodu, R.W.HAMMING tarafından Bell Laboratuvarları’nda geliştirilen *Hamming Kodu*’dur. Hamming Kodu’ndaki bit sayısı, veri karakterindeki bit sayısına bağlıdır.

- FEC yalnızca hata denetimi yapar.
- ARQ ise hata denetimi ile birlikte bozulan verinin tekrar iletilmesini sağlar.

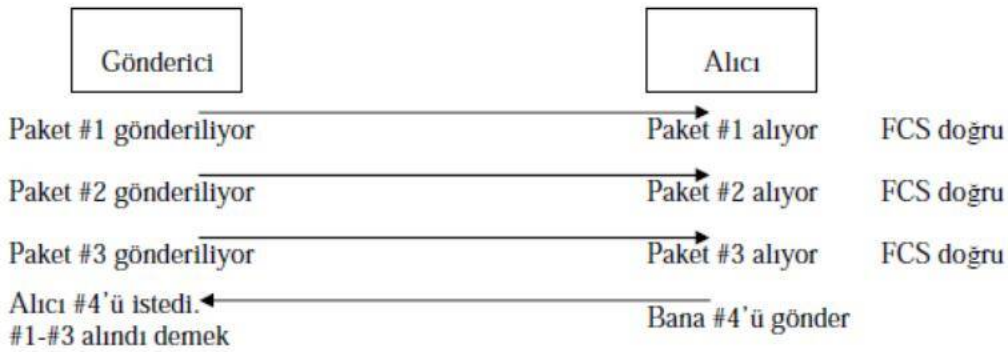
FEC yeniden iletimin çok zor veya imkânsız olduğu durumlarda kullanılır. FEC çoğunlukla donanımda uygulanır. Gönderici hataların tespiti ve düzeltilebilmesi için takviye bitler gönderir. Bu ARQ'dan çok daha fazladır. Kanal Kodlama algoritmalarını kullanır.

FEC için kullanılan algoritmalar;

- Katlamalı Kodlar
- BCH Kodlar
- Hamming Kodlar
- Reed-Solomon Kodlar
- Raptor Kodlar

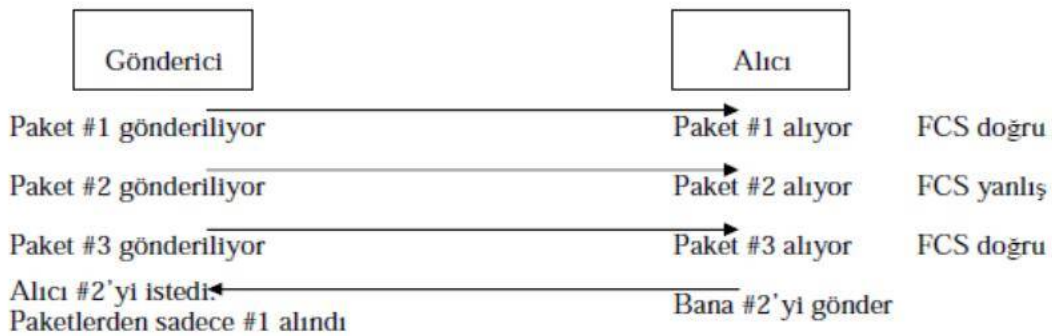
ARQ için kullanılan algoritmalar;

- CRC kodları
- Seri eşlik
- Blok eşlik
- Modül Toplamıdır.



Şekil 19 Sıralı veri iletimi

Gönderici paketleri sıralı bir şekilde gönderir. Alıcı taraf her paket için FCS kontrolü yapar. Ve bir diğer paketi alır. Bu işlem belirli bir paket alındıktan sonra alıcı taraf paket isteğinde bulunur. Bu şekildeki gibi 4. paket isteği gittiğinde ilk 3 paketin doğru bir şekilde iletildiğini gösterir. Paketlerin numaralandırılması bazı protokollerde 0'dan başlar bazılarında gönderici makine tarafından belirlenir. Bazı protokollerde numara çerçeveye, pakete verilirken bazı protokollerde gönderilen byte büyüklüğünü verir.



Şekil 20 Hata giderme

Hata giderme işlemi göndericiden gelen veriler alıcı tarafında FCS kontrolünden geçer ve hata tespit edildiğinde hatalı olan veri tekrar istenir. Alıcının hatalı istediği veri başarılı veri iletişimin olduğu son veridir. Bazı protokollerde hata düzeltme işlemi sadece hatalı veriyi içerir. Bazılarında ise hatalı olan veriden sonraki veriler de tekrar gönderilir.

4. Hata Bulma Teknikleri (Error Detection)

Hata bulma, alınan verileri gözetleme ve ne zaman bir iletim hatası meydana geldiğini gösterme sürecidir. Hata bulma teknikleri, hangi bitin (ya da bitlerin) hatalı olduğunu belirtmez, yalnızca hata meydana geldiğini belirler. Hata bulmanın amacı hatayı önlemek değil, meydana gelen bütün hataları bulmaktır. Her sistem bulunan hatalara karşı farklı bir şekilde karşılık verir. En yaygın hata bulma yöntemleri: artıklık, tam sayım kodlaması, eşlik, düşey ve boyuna artıklık denetimi ve çevrimsel artıklık denetimi.

4.1. Artıklık:

Artıklık her karakteri iki defa iletmektir. Eğer ardışık olarak iki kere aynı karakter veya mesaj alınmazsa, bir iletim hatası meydana gelmiş demektir.

4.2. Tam sayım kodlaması:

Tam sayım kodlamasında, karakterlerdeki 1 sayısı aynıdır. ARQ kodunda her karakter üç tane bir içerir. Bu şekilde her karakterdeki 1'lerin sayısı kontrol edilerek hatalar bulunur.

4.3. Yankılama: (Echoplex):

Bu yöntem daha çok bir merkeze bağlı terminallerde kullanılır. Terminalde bir işlem yapıldığında bu işlem merkeze bildirilir ve merkezden yankılama yapılarak terminal üzerinde ekrana veya yazıcıya basılarak görüntülenir. Kullanıcı buna bakarak doğruluk kontrolü yapar.

4.4. Eşlik Denetimi (Parity Check)

En sık kullanılan hata saptama yöntemidir. Eşliğin (Parity) hesaplanması iletilen verinin içerisindeki birlerin bulunmasıyla yapılır. Tek eşlik kullanılıyorsa birlerin sayısı eşlik biti ile beraber tek olur. Çift eşlik kullanılıyorsa birlerin sayısı eşlik biti ile beraber çift olur.

Örneğin; dört bitlik veri 0110 ise çift denklik kullanılmışsa denklik biti 0 olur. Tek denklik kullanılmışsa denklik biti 1 olur.

Çift eşlik bitini bitlerin birbirleriyle XOR'lanması (Dışlayıcı VEYA) ile hesaplanır. Çıkan sonuç gönderilen eşlik biti ile kontrol edilir. Tek eşlik biti kullanılmış ise hesaplama NXOR (Ters Dışlayıcı VEYA) ile yapılır.

0110 - 0 Çift Denklik

0110 - 1 Tek Denklik

Even Parity

“Even parity” Çift Parite devresinde sayısal bilgidaki “1” ler çift olacak şekilde parite biti hesaplanır. Örneğin:

1101 bilgisinde “1” sayısı tektir, o halde eklenecek parite biti “1” dir. 1101**1**

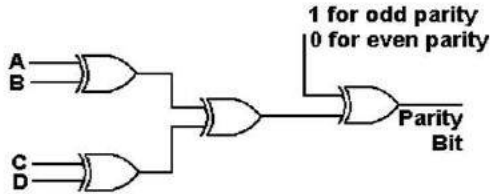
1100 bilgisinde “1” sayısı çifttir, o halde eklenecek parite biti “0” dır. 11000

Odd Parity

“Odd parity” Tek Parite devresinde sayısal bilgideki “1” ler tek olacak şekilde parite biti hesaplanır. Örneğin:

1101 bilgisinde “1” sayısı tektir, o halde eklenecek parite biti “0” dir 11010

1100 bilgisinde “1” sayısı çifttir, o halde eklenecek parite biti “0” dır. 11001



Yanda örneğin 4 bitlik bir veri için Tek veya Çift parite bitinin hesaplayan örnek bir sayısal devre görülmektedir.

4.5. İki boyutlu Eşlik Denetimi (2-Dimentional Parity)

İki boyutlu eşlik sınamasında verilerin her birinin Eşlik bitlerinin haricinde bit sırasına göre de eşlik biti oluşturulur. Bu yöntemde hata tespit edilebildiği gibi hangi bitin hatalı olduğu da bulunabilir. Her veri için bir eşlik bitinin haricinde ayrıca eşlik verisinin gönderilmesi gerekmektedir.

1	0	0	1	0	1	0	1
0	0	0	1	1	0	0	0
0	1	0	0	0	1	1	1
0	1	1	0	1	0	1	0
0	0	1	0	1	1	0	1
1	0	0	1	0	1	0	1
1	0	1	0	1	0	1	0
1	0	1	1	0	0	1	0

Düşey artıklı denetleme (VRC), bir karakterde iletim hatası meydana gelip gelmediğini bulmak üzere eşlik kullanan bir hata bulma tekniğidir. Bu nedenle, VRC'ye bazen *karakter eşliği* de denir. VRC'de her karaktere iletilmeden önce bir eşlik biti eklenir. VRC'de çift ya da tek eşlik kullanılabilir. Eşlik kavramını açıklarken verdiğimiz ASCII C karakteri örneği, VRC'nin nasıl kullanıldığına gösteren bir örnektir.

Yatay ya da boyuna artıklık denetleme (HRC ya da LRC), bir mesajda iletim hatası meydana gelip gelmediğini belirlemek için eşlik kullanan bir hata bulma tekniğidir; bu nedenle bazen *mesaj eşliği* adını alır. LRC'de, her bit konumunun bir eşlik biti vardır. Başka bir deyişle, mesajdaki karakterin b0'ı, Mesajdaki öteki karakterlerin hepsinin b0'larının eşliklerini. Benzer şekilde b1,b2,b3,.. karakterlerinin hepsinin eşliklerini kontrol eder. Temel olarak LRC, bir mesajı oluşturan karakterlerin eşlikleridir. LRC'de yalnızca çift eşlik kullanılır.

Verileri göndermeden önce LRC bit sırası vericide hesaplanır, daha sonra sanki mesajın son karakteriymiş gibi iletir. Alıcıda, LCR alınan verilerden tekrar hesaplanır

ve tekrar hesaplanan LRC, mesajla iletilen LRC'yle karşılaştırılır. Eğer iki LRC aynı ise, iletim hatasının gerçekleşmediği varsayılır. Eğer LRC'ler farklıysa, bir iletim hatası meydana gelmiş demektir.

ÖRNEK 1: ASCII kodlanmış şu mesajın VRC ile LRC'sini bulalım: THE CAT. VRC için tek eşliği LRC için de çift eşliği kullanalım.

Karakter		T	H	E	sp	C	A	T	LRC
On altılı		54	48	45	20	43	41	54	2F
LSB	b0	0	0	1	0	1	1	0	1
	b1	0	0	0	0	1	0	0	1
	b2	1	0	1	0	0	0	1	1
	b3	0	1	0	0	0	0	0	1
	b4	1	0	0	0	0	0	1	0
	b5	0	0	0	1	0	0	0	1
MSB	b6	1	1	1	0	1	1	1	0
VRC	b7	0	1	0	0	0	1	0	0

LRC bütün iletim hatalarının %95 ile %98'ini bulur. LRC, aynı bit konumunda çift sayıda karakter hatalıysa, iletim hatası bulunmaz. Örneğin, iki farklı karakterde b4 hatalıysa, birden fazla iletim hatası meydana gelmiş olmasına karşın LRC gene de hata göstermez.

LRC ve VRC aynı anda kullanılırsa, hatanın algılanmadığı tek durum şu olur: çift sayıda karakterde çift sayıda bit hatalı olduğunda ve bu iki karakterde aynı bit konumları hatalı olduğunda; böyle bir şeyin meydana gelme olasılığı çok düşüktür. VRC bir karakterde hangi bitin hatalı olduğunu bulamaz; LRC de hangi karakterde hata olduğunu bulamaz. Ancak, tek bit hatalarda, LRC ile birlikte kullanıldığında VRC hangi bitin hatalı olduğunu bulur. Aksi taktirde, tek başlarına VRC ve LRC, yalnızca bir hatanın meydana geldiğini bulabilirler.

4.6. LRC- Boyuna Fazlalık Sınaması:(Longitudinal Redundancy Check)-Toplama Denetimi

Veri Dizilerini Modül toplamalarının kullanıldığı hata denetleme yöntemidir. Toplama işlemi genellikle son byte'ın toplama sınaması numarasını ifade ettiği byte paketleri üzerinde gerçekleştirilir. Paket içindeki tüm sayıları topladığınızda mod 256 üzerinden toplam sıfır olmalıdır. Eğer değilse hata oluştuğu tespit edilir.

Bu yöntem iki veya daha fazla alt dizi içeren veri dizileri içindeki hataları sezmek için kullanılır. Alt veri dizilerinin aynı pozisyonundaki bitlere tek ya da çift eşlik uygulanarak bir sına ma dizisi oluşturulur ve bu diğerlerine eklenerek gönderilir. Bu yöntemle aynı pozisyon daki çift sayıda bitin bozulması sezilemez.

$B_{0,0} \dots B_{0,k-1}$	$p'0$	
$B_{1,0} \dots B_{1,k-1}$	$p'1$	
$\dots \dots \dots$		
$B_{j-1,0} \dots B_{j-1,k-1}$	$p'j-1$	
$p_0 \dots p_{k-1}$	p	$\rightarrow$ sına ma katarı

k bit uzunluğunda karakterlerden **j** adet gönderilsin karakter bitlerini gönderme sırasına göre yazalım ve alt alta sıralayalım karakterlerin her bir biti ilgili sütunun eşlik bitine karşılık gelir. Ve bunların toplam ları alınarak sına ma karakteri oluşturulur. Bu yöntem (**checksum**) **toplama** sına ması olarak da bilinir.

4.7. CRC (Cyclic Redundancy Check) Çevrimli Fazlalık Sına ması:

Uygulaması kolay ve güvenilirliği güçlü bir tekniktir. CRC tekniği veri çerçevelerinin korunması için kullanılır. Gönderici her çerçeveye **n** bitlik FCS (Frame Check Sequence) dizi ekler. CRC (dönüşümlü fazlalık sına ması) veri iletiminde kullanılan en yaygın hata denetimi yöntemidir. Az bir ek bilgi ile daha fazla hata tespiti sağlar.

Muhtemelen en güvenli hata bulma tekniği, çevrimsel artıklık denetleme (CRC). CRC'de, bütün iletim hatlarının yaklaşık %99.95'i bulunur. CRC genellikle EBCDIC gibi 8 bitli kodlarla ya da eşlik kullanılmadığı zaman 7 bitlik kodlarla kullanılır.

Amerika Birleşik Devletlerinde en yaygın CRC kodu, CRC-16'dır. CRC-16'da, BCS için 16 bit kullanılır. Temel olarak CRC karakterleri, bir bölme işleminin kalanıdır. Veri mesaj polinomu $G(x)$, bit üretme polinom fonksiyonuna $P(x)$ bölünür, Bölüm dikkate alınmaz; kalan, 16 bite indirilip, BCS olarak mesaja eklenir. CRC üretiminde bölme işlemi, standart bölme işlemi ile gerçekleştirilmez. Doğrudan çıkarma kullanma yerine, kalan özel veya işlemi ile üretilir. Alıcıda, veri akışı ve BCS aynı üretme fonksiyonuna ($P(x)$) bölünür. İletim hatası meydana gelmemişse, kalan sıfır olur.

CRC-16'nın üretme polinomu şudur:

$$P(x) = x^{16} + x^{12} + x^5 + x^0 \text{ burada } x^0 = 1.$$

CRC kodundaki bitlerin sayısı, üretme polinomundaki en yüksek kuvvete eşittir. Kuvvetler, 1 içeren bit konumlarını belirler. Dolayısıyla, b_{16}, b_{12}, b_5 ve b_0 1; öteki bit konumlarının hepsi 0'dır.

CRC veri bitlerini polinom kodlar olarak ele alır. Veriler, gönderici tarafından çerçevenin içeriğine göre her çerçevenin sonuna eklenecek bir denetim seti (check digits) hesaplanarak gönderilir. Alıcı taraf veri üzerinde aynı denetim işlemlerini gerçekleştirerek aldığı veride hata olup olmadığını kontrol eder. İki sonuç uymuyorsa bu hata olduğunu gösterir. n bitlik FCS için n+1 bit bir P polinomu kullanılır. Amaç FCS'yi Q/P sıfır olacak şekilde elde etmektedir.

$$\frac{(M \cdot 2^n + R)}{P} = Q + \left(\frac{R}{P} + \frac{R}{P} \right)$$

M: k bitlik Asıl veri

R: kalan n bitlik sayı

P: üreteç polinomu n+1 bit

Q: FCS dizisi

CRC oluşturulması ilk olarak veri içeriği 2^n ile çarpılır. Bu verinin sonuna n adet 0 ekleme demektir. İkinci adımda P üreteç polinomu ile bölünmesi, üçüncü adımda kalanın FCS olarak iletilmesidir.

CRC kontrolü ise ilk olarak veri çerçevesi alınır. İkinci adımda P üreteç polinomuna bölünür. Üçüncü adım kalan kontrol edilir. Kalan sıfır ise veri doğru olarak alınmıştır. Eğer sıfırdan farklı ise hata oluşmuştur.

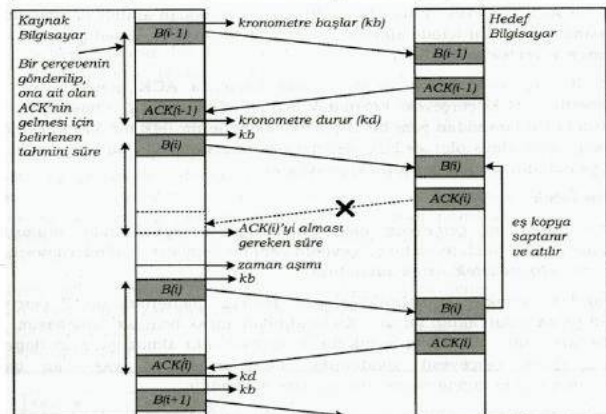
Bu metotta CRC adı verilen ve gönderilen veri dizisinden hesaplanan bir sınama dizisi, bu veri dizisinin sonuna eklenir. CRC dizisini hesaplamanın değişik yöntemleri vardır.

5. CRC katarı hesaplaması :

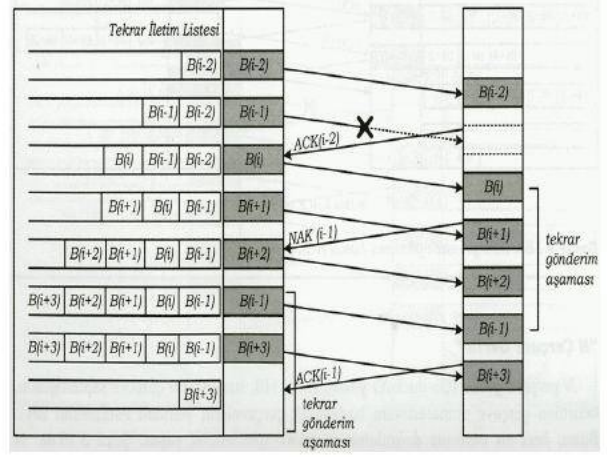
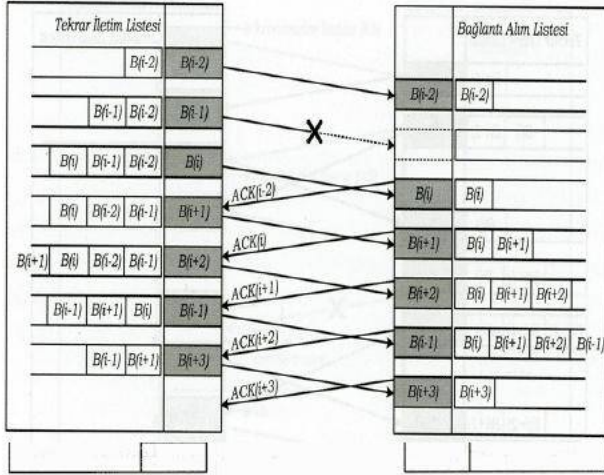
- Veri katarı $P(x)$ polinomu ile gösterilir. Polinomun katsayıları 1 veya 0 ilgili pozisyondaki bitin değeridir.
- $P(x) = b_{n-1} \cdot x^{n-1} + b_{n-2} \cdot x^{n-2} + b_{n-3} \cdot x^{n-3} + \dots + b_1 \cdot x^1 + b_0 \cdot x^0$ bit değerleri $\rightarrow b$
- $P(x)$ polinomu x^p ile çarpılır. Bu işlem sonucunda elde edilen bit katarı ilk bit katarının sağına düşen p adet 0 bitinden oluşur.
- $x^p P(x)$ polinomu p. Dereceden $G(x)$ üreteç polinomuna bölünür.
- $x^p P(x) = Q(x) G(x) + R(x)$ Göndericiden $x^p P(x) + R(x)$ polinomu gönderilir. (Asıl veri ve p uzunluğunda ek dizi) $G(x)$ alıcı taraftan bilindiği için herhangi bir bitin bozulup bozulmadığı $Q(x)G(x)$ doğru gelip gelmediğine bakılarak anlaşılır. $G(x)$ 'e tam bölünüyorsak veri hatasız gelmiştir demektir.

6. ARQ (Automatic Repeat Request–Otomatik Tekrar İsteği)

Alıcı tarafta düzeltilemeyecek bir hata oluştuğunda bu veri göndericiden tekrar alınması gerekir. Bunun için ARQ protokolleri tanımlanır. ARQ alıcı tarafın hatalı veri kümesinin göndericiden yeniden alıcıya istenmesini tanımlar. ARQ protokolleri veri bağı katmanında çalışmaktadır. İletim sırasında verinin bütünlüğü ve doğru bir şekilde iletilmesi için ARQ kullanılır. ARQ için uygulanan yöntemler olumlu doğrulama, zaman aşımı sonrasında yeniden gönderim, olumsuz doğrulama ve yeniden gönderim.



Sürekli Tekrar İstemi (IDLE RQ)



7. Hamming Kodlama Sistemi

Hamming kodlama sistemi telekomünikasyonda kullanılan bir çeşit doğrusal hata kontrol ve düzeltme yöntemidir. Hamming kodlama yöntemi ile verileri paketlerindeki tek bitlik hatalar bulunup düzeltilebilir. Yine bu yöntem ile iki bitlik hataları da tespit edebiliriz fakat düzeltemeyiz.

Richard Hamming 1940 yılında Bell Laboratuvarlarında Bell Model V adı verilerin bir çeşit elektromekanik makine üzerinde çalışmaktaydı. Bu makine verileri Hollerith card ya da şimdilerde IBM card olarak da bilinen kağıttan kartlar üzerinden okumaktaydı. Özel bir kod sistemi bu kartları kontrol eder ve bir hata bulduğunda makinenin ışığı yanıp sönyordu, böylelikle o sırada çalışmakta olan operatör bu hatayı düzeltebilmekteydi. Fakat çalışma saatleri dışında ve hafta sonları, yani makinenin başında çalışan bir operatör bulunmadığı zamanlarda veriler kontrol edilmeksizin makine çalışmaya devam etmekteydi.

Hamming hafta içi çalışmaktaydı ve her hafta başında bu güvenilir makine yüzünden çalışmalarına yeniden başlamak zorunda kaldığı için bu konudaki kızgınlığı giderek artmaktaydı. Sonraki birkaç sene boyunca hata kontrolü problemi ile ilgili çalışmalar yaptı ve gitgide bu konuda daha etkili algoritmalar geliştirdi. Sonunda 1950 yılında kendi ismini verdiği, bu gün bile halen kullanılmakta olan "Hamming Kodlama" yöntemini açıkladı.

7.1. Hamming Kodu

Eğer daha fazla hata kontrolü biti dataya eklenir ve bu bitler her bir hatalı bit farklı hata sonuçları üretecek şekilde ayarlanabilirse hatalı bit bulunur ve düzeltilebilir. 7 bitlik bir veri olası 7 tane farklı tek bitlik hata durumu vardır ve bu farklı durumlar toplam 3 bitlik hata kontrol bitleri kullanarak ifade edilebilir. Böylece hatalı bir durum yakalandığında hatanın hangi bitten kaynaklandığını da muhtemelen 3 bitlik kontrol bitleri yardımıyla öğrenilebilir.

Hamming, beşte ikilik de dahil olmak üzere, mevcut hata kontrolü yöntemleri üzerinde çalıştı ve içeriklerini genelleştirerek kendi yöntemi üzerinde düşünmeye başladı. İşe sistemi tanımlamak için blokların içerdiği data bitlerinin ve kontrol bitlerinin sayısını gösteren bir terminoloji geliştirerek başladı. Örneğin parity yönteminde gönderile her veri paketi için bir bitlik kontrol biti eklenmekteydi, bunun 7 bitlik bir ASCII kodu olduğunu varsayalım. Hamming bunu 7'si data olmak üzere toplam 8 bitten oluşan bir (8,7) olarak tanımladı. Yine aynı şekilde düşünecek olursak repetition yöntemi de bir (3,1) dir. *Bilgi oranı* (information ratio) ise ikinci sayı birinciye bölünerek elde edilebilir. Örneğin yine repetition yöntemi için bu oran 1/3' tür.

Hamming ayrıca iki veya daha fazla bitin değişmesiyle meydana gelen hataları da fark etti ve bunu "mesafe" (*distance*) olarak tanımlamıştır ki bu terim de kendisinden sonra "Hamming's distance" olarak anılmaktadır. Yine diğer hata düzeltme yöntemlerinden örnek verecek olursak, parity yönteminin mesafesi 2dir. Çünkü eğer 2 bitlik bir değişme olursa bu hata yöntem için görülemez hale gelir. (3,1) repetition yönteminin mesafesi ise 3 tür, çünkü hatanın anlaşılamayacağı yeni bir 3lü blok oluşması için 3 bitin birden değişmesi gerekir.

Hamming bu iki temel sorun üzerinde yoğunlaştı. Amacı mesafeyi mümkün olduğunca arttırırken bir yandan da bilgi oranını olabildiğince yüksek tutmaktır.

1940'lı yıllar boyunca yaptığı çalışmalarında mevcut hata kontrolü yöntemlerine göre çarpıcı yeniliklere sahip birkaç yöntem geliştirdi. Oluşturduğu sistemlerin hepsinin püf noktası parity bitlerinin aynı anda hem birbirlerini hem datayı kontrol edebilecek şekilde örtüşerek kullanılmasıydı.

Hamming'in son olarak oluşturduğu "Hamming Kodlama" yönteminde parity bitlerinin kullanılması için geliştirdiği algoritma basitti;

1. İkinci katı olan bütün bit pozisyonları parity bitleri için kullanılacak.
(ör: 1,2,4,8,16,32,64,128....vb.)
2. Diğer tüm pozisyonlar paketlenen datalar için kullanılacak.
(ör: 3,5,6,7,9,10,11,12,13,14,15,17.....vb.)
3. Her bir parity biti veri paketindeki bazı diğer bitlerin kontrolünü yapar. Parity bitinin pozisyonu kontrol edeceği ya da atlayacağı bitlerin sıklığını belirtir.
 - 1. pozisyonda ($n = 1$): 0 bit atla ($0 = n-1$), 1 bit kontrol et (n), 1 bit atla (n),

1 bit kontrol et (n), 1 bit atla (n).... vb.

- 2. pozisyonda (n = 2): 1 bit atla (1 = n-1), 2 bit kontrol et (n), 2 bit atla (n),

2 bit kontrol et (n), 2 bit atla (n).... vb.

- 4. pozisyonda (n=4): 3 bit atla (3 = n-1), 4 bit kontrol et (n), 4 bit atla (n),

4 bit kontrol et (n), 4 bit atla (n).... vb.

- 8. pozisyonda (n=8): 7 bit atla (7 = n-1), 8 bit kontrol et (n), 8 bit atla (n),

8 bit kontrol et (n), 8 bit atla (n)... vb.

- 16. pozisyonda (n=16): 15 bit atla (15 = n-1), 16 bit kontrol et (n), 16 bit atla (n),

16 bit kontrol et (n), 16 bit atla (n).... vb.

- 32. pozisyonda (n=32): 31 bit atla (31 = n-1), 32 bit kontrol et (n), 32 bit atla (n),

32 bit kontrol et (n), 32 bit atla (n).... vb.

7.2. (11,7) Hamming Code Kullanımı Örneği

Daha önce açıklandığı gibi (11,7) lik gösterim 7'si data olan toplam 11 bitlik bir veri paketini ifade etmektedir. Örneğin bu 7 bitlik datamız "0110001" olsun. Hamming kodlama sisteminde parity bitlerinin nasıl hesaplanıp, hata tespitinde kullanıldığını aşağıdaki tablolarda görebilirsiniz. Bu tablolarda **d** data bitlerini, **p** ise parity bitlerini ifade etmek için kullanılmıştır. Öncelikle data bitleri uygun pozisyonlarına yerleştirilir. Daha sonra parity bitleri ilk pozisyondan itibaren hesaplanarak sırayla eklenir.

	p1	p2	d1	p3	d2	d3	d4	p4	d5	d6	d7
Parity bitsiz data			0		1	1	0		0	0	1
p1	0		0		1		0		0		1
p2		0	0			1	0			0	1
p3				0	1	1	0				
p4								1	0	0	1
Parity bitli data	0	0	0	0	1	1	0	1	0	0	1

Eklenen parity bitleri ile birlikte yeni datamız "00001101001" oldu. Veri iletilirken baştan 3. bitin bozulduğunu ve karşı tarafa "0" değil de "1" olarak iletilildiğini varsayalım. Bu durumda "00001101001" olan datamız alıcı terminal tarafından "00101101001" olarak algılanacaktır. Aşağıda hamming kodlama yöntemiyle paketlenen datada hatalı bitin nasıl tespit edileceğini göstermektedir. Yeni veri paketinde parity bitler teker teker kontrol edilip doğru olanlar için "0", yanlış olanlar için "1" yazılır.

	p1	p2	d1	p3	d2	d3	d4	p4	d5	d6	d7	Parity Kontrol	Parity bit
Alınan data	0	0	1	0	1	1	0	1	0	0	1		
p1	0		1		1		0		0		1	Yanlış	1
p2		0	1			1	0			0	1	Yanlış	1
p3				0	1	1	0					Doğru	0
p4								1	0	0	1	Doğru	0

Son olarak da elde ettiğimiz bu 4 bitlik veri düşük anlamlı p1 olacak şekilde sağdan sola doğru sıralanır ve 4 bitlik “0011” elde edilir. Artık hatalı olan bitin yerini tespit edebiliriz;

	p4	p3	p2	p1	
İkilik	0	0	1	1	
Ondalık			2	1	$\Sigma = 3$

Böylelikle hatalı bitin 3. sıradaki olduğunu tespit etmiş olduk. Bu sayede “00101101001” olan hatalı veriyi, sorun olduğunu bildiğimiz 3. biti değiştirerek tekrar ilk hali olan “00001101001” e dönüştürebiliriz.

7.3. (7,4) Hamming Kodlama

Günümüzde Hamming kodlama deyince akıllara ilk olarak 1950 yılında açıklanan (7,4) lük Hamming kodlama yöntemi gelmektedir. Bu Hamming kodlama yönteminde her 4 bitlik data için 3 bit kontrol biti eklenir. Hamming' in geliştirdiği bu (7,4) algoritması her hangi bir tek bitlik hatayı düzeltebilir ya da bütün tek veya iki bitlik hataları tespit edebilir.

7.4. Hamming Matrisleri

Hamming kodu parity matrislerini çoklayarak genişletilmesinden oluşan ve “Haming matrisi” adı verilen matrislerle çalışır. (7,4) lük Hamming kodlama da birbirleriyle bağlantılı olan 2 çeşit matris kullanırız; *Kod üretici matris G* ve *Parity kontrol matrisi H*.

$$\mathbf{G} := \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{pmatrix} \text{ ve } \mathbf{H} := \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}.$$

G matrisinin ilk 4 satırı, 4×4 'lük Identity matrisi (**I**) iken son 3 satırı ise 4 veri bitinin 3 parity biti ile 4×3 lük matriste adreslenişinden elde edilir. **G'** deki satır vektörleri ise **H** matrisinin kernelinin temellerini oluşturur. Identity matrisi çarpma işlemi sırasında data vektörünü aktarır. Yukarıdaki açıklamanın aksine data bitleri ilk 4 pozisyonudadır, parity bitleri ise son 3 pozisyona yerleştirilmiştir.

Ayrıca bu matrisler asıl Hamming matrislerinden biraz farklı olsa da, Hamming kodlamayı anlamamızı kolaylaştıracak temel özelliklerini taşımaktadır.

H matrisinin oluşturulması da **G** matrisine benzerlik göstermektedir. **H** matrisinde her satırın son 3 sütunu 3*3'lük Identity matrisidir. İlk 4 sütun ise yine **G** matrisinde de kullandığımız ve

data bitleri ve parity bitlerinin adreslenmesinden oluşan 4*3' lük matristir.

Yukarda bahsedildiği ve isminden de anlaşılabilceği gibi bu yöntemde 4 bit veri iletilir. Örneğin göndereceğimiz 4 bitlik data "1001" olsun. Bu datayı göndermek için kullanacağımız vektör; olur.

7.5. Kanal kodlama

Datamızı paketlemek için **G** matrisini kullanırız. 7*4'lük **G** matrisi ile asıl datamızı taşıyan 4*1'lik **p** matrisinin mod2 ye göre çarpılmasından elde edeceğimiz 7*1'lik bir **x** matrisi (7,4) lük Hamming kodlama yöntemi ile paketlenmiş veriyi oluşturacaktır.

$$\mathbf{Gp} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \mathbf{x}$$

7.6. Parity Kontrolü

Eğer veri iletimi sırasında bir hata oluşmazsa alınan veri matrisi **r**, iletilen veri matrisi **x'** e eşit olmalı (**r = x**).

Alıcı terminal, eğer varsa hatalı bitin tespiti için kullanacağı **z** matrisini elde etmek için **H** ve **r** matrislerini çarpar. Matrislerin çarpımı yine mod2 ye göre yapılır çünkü bitlerle çalışıyoruz. (her bir bit sadece 0 ya da 1 değerlerinde birini alabilir.)

$$\mathbf{Hr} = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} = \mathbf{z}$$

7.7. Hatanın Düzeltilmesi

Eğer iletim sırasında tek bitlik bir hata meydana gelirse **r** ve **x** matrisleri farklı olacaktır ve bu farkı şu matematiksel ifade ile gösterebiliriz;

$$\mathbf{r} = \mathbf{x} + \mathbf{e}_i$$

burada **n**. pozisyonunda "1" bulunan bir zero (sıfır) vektörüdür. Aşağıdaki ifadeler **n**. sıradaki tek bitlik hatanın belirlenişini gösterir;

Eğer **r** matrisini **H** matrisi ile çarparsak;

$$\mathbf{Hr} = \mathbf{H}(\mathbf{x} + \mathbf{e}_i) = \mathbf{Hx} + \mathbf{He}_i$$

İfadesini elde ederiz. Bu ifadede $\mathbf{x}$ asıl iletilen data olduğu için $\mathbf{H}$ matrisi ile çarpımı sıfır olacaktır. Buna göre;

$$\mathbf{Hx} + \mathbf{He}_i = \mathbf{0} + \mathbf{He}_i = \mathbf{He}_i$$

3*7 lik H matrisi ile 7*1'lik nin çarpımı olan $\mathbf{z}$ matrisi bize hatalı olan bitin konumunu bildirecektir. Burada iletim süresince $\mathbf{x}$ matrisinde de oluşan hata, örneğin $i = 3$ için $\mathbf{r}$ matrisine şu şekilde yansır;

$$\mathbf{r} = \mathbf{x} + \mathbf{e}_3 = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

Şimdi ise bu hatalı $\mathbf{r}$ matrisini $\mathbf{H}$ ile çarptığımız zaman elde edeceğimiz $\mathbf{z}$ matrisi bize hatalı olan bitin konumunu verir;

$$\mathbf{Hr} = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} = \mathbf{z}$$

Burada $\mathbf{z}$ matrisinin gösterdiği 3*1'lik ifade H matrisindeki sütunlardan birine eşittir. İşte bu sütunun sırası aynı zamanda bize iletilen datada ki bozulan bitin sırasıdır. Yukarıdaki örnek için hatalı bit;

$$\mathbf{Hr} = \begin{pmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} = \mathbf{z}$$

1. 2. 3. 4. 5. 6. 7.

3. sıradakidir. Bunu bildiğimiz için 3. sıradaki biti çevirerek asıl datamıza ulaşırız;

$$\begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} \Rightarrow \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix}$$

8. Reed-Solomon Göndermede Hata Düzeltim Kodları

Reed-solomon GHD (Gönderimde Hata Düzeltim) kodları hataların düzeltilmesi için çözümleyicide donanımsal düzeyde yapılandırma gerektirir. Bundan dolayı oldukça karmaşıktır. Reed-Solomon GHD kodları hata düzeltmede kullanıldığında $\lfloor (n-k)/2 \rfloor$ nin herhangi bir kombinasyonunu çözme yeteneğine sahiptir. Diğer taraftan, Reed-Solomon kodları alıcı tarafında k tane sembolü elde ettiğinde $(n-k)$ tane ekstra sembolü başarıyla çözümleyebilir.

Reed-Solomon GHD kodları güçlü lineer hata düzeltim kodlarıdır. Ancak paket düzeyinde kayıpların önlenmesi için bazı kısıtlamalara sahiptir ve etkin değildir. Temel kısıtlama şudur: Reed-solomon algoritması "Galois Alanları" üzerinde çalışır. Her bir alan 8 bit (1 byte) ile GHD Kodlayıcısı GHD Çözücüsü sınırlıdır. Özellikle gerçekleştirim yazılım tabanlı ise ya da işlemci gücü sınırlı ise uygulamak kolaydır.

9. Raptor Göndermede Hata Düzeltim Kodları

Raptor ve Reed-Solomon kodlarının ortak değerleri olan "n", "k", ve "S" farklı şekillerde kullanılır. N kodlayıcıdan çıkan sembol sayısı, S bir sembolün uzunluğu ve kaynak sembollerin sayısıdır. Belirli bir Reed-Solomon kodu $R-S(n,k,S)$, $n-k$ tane bozuk veri düzeltilebilir. N tane sembolün maksimum kayıp ile gönderiminde $(n-k)/n$ tane sembole tahammül edilebilir.

Raptor kodunda ise alıcı tarafında $k' > k$ tane sembol alınır ve k tane kaynak sembolü tekrar yaratmada kullanılır. N tane sembolün maksimum kayıp ile gönderiminde $(n-k')/n$ tane sembole tahammül edilebilir. $kP = (R+1)k$, ve $R > 0$ olmak kosulu ile alıcının Raptor kodu ile sembolleri kabul etmesinde getirdiği ek yükü ifade eder. Raptor kodu belirli bir olasılık değerinde çözümleme başarısını garanti eder. $(k'-k)$ tane ekstra sembol alıcı tarafında alınır. Alıcı ek yükü $R = (k'-k)/k$ formülü ile hesaplanır. Bu formül k değerine ve istenen olasılık değerine (kaynak sembollerin alınan sembollerden yaratılması) bağlıdır.

Raptor kodunun alıcı ek yükü %1 den daha azdır. Reed-Solomon kodundan farklı olarak, Raptor kodu verinin büyüklüğü ne olursa olsun herhangi bir sınırlamaya sahip değildir. Raptor kodu için kaynak sembol sayısı (k) kaç olursa olsun, sembolün uzunluğu olan S bundan bağımsızdır ve bir paketin boyuna eşit olabilir ya da ondan kısa olabilir. Sonuç olarak S değeri

Raptor kodlarında ve Reed-Solomon kodlarında oldukça küçük olabilir. İşlemci gücü ise Raptor ve Reed-Solomon da biraz farklıdır. Şu açıktır ki Raptor kodunun ihtiyaç duyduğu işlemci gücü, Reed-Solomon kodunun ihtiyaç duyduğu işlemci gücünden oldukça azdır. Reed-Solomon kodunun ihtiyaç duyduğu işlemci gücü, kaynak bloğunun uzunluğuna bağlı olarak katlamalı artış gösterirken, Raptor kodununki lineer artış gösterir.

SONUÇ

Farklı Hata Düzeltme Tekniklerini ve kodlarını incelerken göz önüne alınması gereken unsurlar, paket kayıp oranı, gerekli olan ağ kaynak miktarı (ağ bant genişliği), gönderici ve alıcı kısımlarında işlemciye getirdiği yük ve algoritmanın karmaşıklığıdır.

KAYNAKÇA

TUNCEL Cengiz, **Bilgisayar Haberleşmesi ve Ağ Teknolojileri Bitirme Ödevi**, Fırat Üniversitesi, Elazığ, 2003

RFC 3453 (rfc3453) – The Use of Forward Error Correction (FEC) in Reliable Multicast. M.Luby Digital Fountain, L.Vicisano Cisco, J.Gemmell Microsoft, L.Rizzo Univ. Pisa, M. Handley ICIR, J. Crowcroft Cambridge Univ., Kasım 2002

David J.C. MacKay (2003) “Information theory, inference and learning algorithms”, CUP

Ar.Gör. Demir, Ufuk, Raptor-Reed Solomon Göndermede Hata Düzeltim Kodları, İzmir

<http://ccnap.marmara.edu.tr>

<http://www.cisco.com>

<http://cisco.netacad.net>

<http://www.ciscoedu.com>

<http://www.cisco.org>

www.microsoft.com/turkiye/akademik/itacademy_benefits.msp

<http://www3.itu.edu.tr/~oktug/BH/notlar/bolum1.pdf>

www.bilisimogretmeni.com

www.wikipedia.org

http://en.wikipedia.org/wiki/Hamming_code

<http://www.cobanoglu.fws1.com/iletsi.htm>