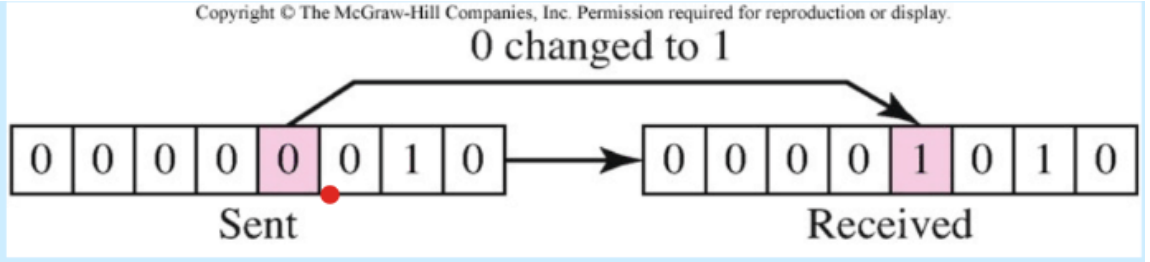
8 - Hata Sezme ve Düzeltme Teknikleri

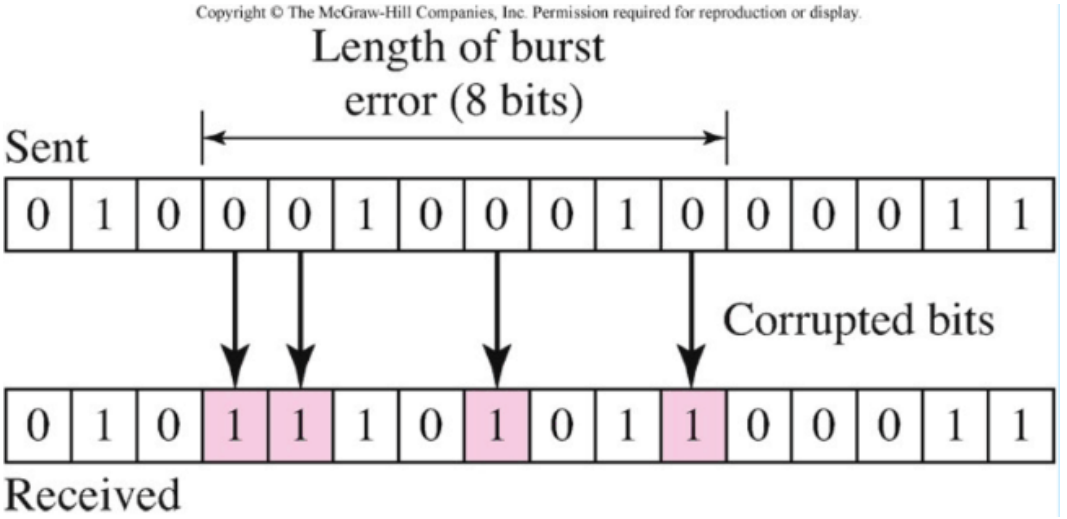
Veri iletilirken bir hata meydana gelip gelmediğini anlamak için kullanılan tekniklerden bahsedilecektir.

Veri paketleri iletilirken bazı bitler, rastgele hatalar veya patlama hataları nedeniyle bozulabilir.

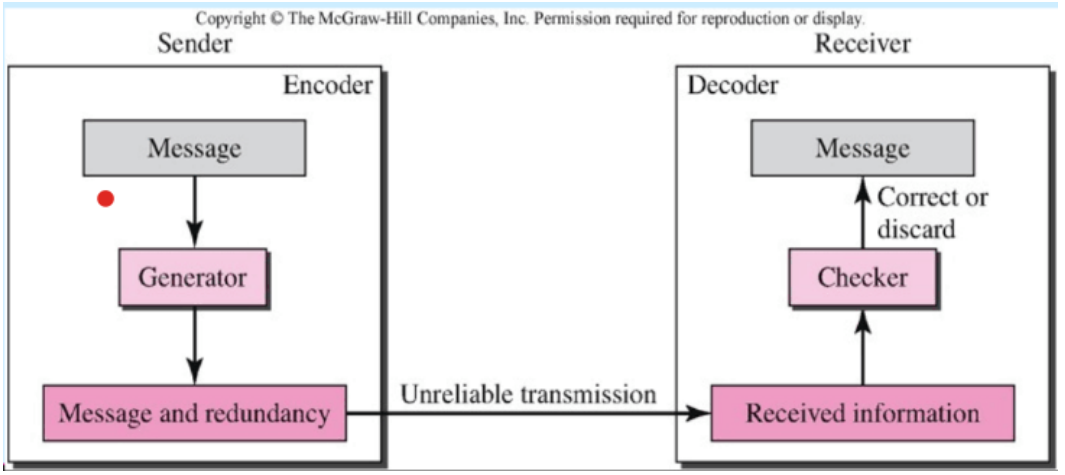
Rastgele hatalar: İletim ortamından iletilen bilgideki bir veya birkaç bitin özellikle gürültü nedeni ile bozulmasıdır (değer değiştirmesidir). Genellikle sadece 1 bitin değeri değişir. Veri paketleri içerisinde 1 bit bozulmuş olsa bile, tüm verinin yanlış anlaşılmasına neden olabilir.



Patlama hataları (burst error): Kısa süreli güçlü elektromanyetik ortamlarda oluşan darbe gürültülerinde veya yıldırım oluşması gibi atmosferik olaylarda iletilen bilginin / bitlerin bir kısmı (iki veya daha fazlası) değer değiştirebilir. Bu hatalar, birbirine yakın konumdaki çok sayıda bitin bozulması şeklinde sonuçlanır. 1200 bps hızında iletişim yapılırken eğer 0,01 sn patlama hatası ya da gürültü oluşursa, toplam 12 bit bozulur.



Aktarmak istediğimiz veriyi Generator kısmında özel bir fonksiyona sokuyoruz. (Hata sezme tekniklerinden biri olabilir.) Bunun sonucunda mesaj ile beraber ek bit (yedekleme bitleri) iletim ortamında gönderilir. Alıcı tarafında veri alındıktan sonra Checker ın sağladığı bir kontrol işlemi vardır. Kontrol sonucunda doğru veya yanlış kararı verilir.



Hata sezme tekniklerinde sadece hata olup olmadığına bakılır, hatanın boyutuyla ilgilenilmez.

Hata düzeltme tekniğinde hata varsa düzeltme işlemi yapılır. Önce kaç bitte bozulma olduğu bulunur.

Parity (Eşlik) Biti – Hata sezme Tekniği
Bu teknik, daha çok boyu 7 ya da 8 bit gibi kısa olan veri bloklarının aktarılmasında kullanılır.

Bir veri bloğu içerisindeki tek sayıdaki hatayı sezmek için kullanılır. (Bu kadar düşük bitlerin iletiminde genelde 1 bitlik hatalar meydana gelebilir. Parity de 1-3... kadar hatayı tespit edebildiği için işimize yarayacaktır.)

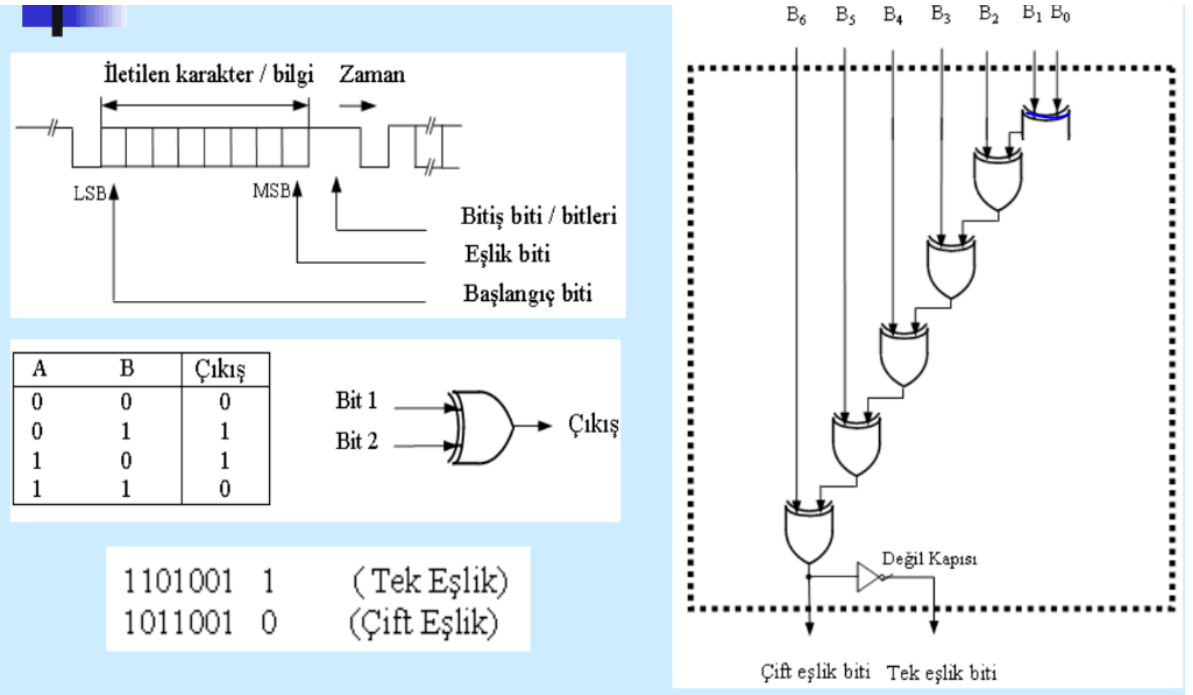
Odd (Tek) yada even (çift) olarak adlandırılan iki uygulaması vardır. Gönderilen veriye koyacağımız parity bitini de hesaba kattığımızda gönderilen toplam mesajdaki 1 lerin sayısının tek olmasını istemek tek eşlik, 1 lerin sayısının çift olmasını istemek çift eşlik olarak adlandırılır.

Vericinin ve alıcının başlangıçta, hangi eşlik bitini kullanacağı belirlenmelidir. Aşağıdaki örnekte verici ve alıcının tek eşlik bitine kurulduğu kabul edilerek inceleme yapılmıştır.

Sistem tek eşlik bitine kurulduğu için gönderilecek veriye bakıyoruz eğer tek sayıda 1 bitine sahipsek parity bitine 0 koyuyoruz ki gönderilecek toplam mesajdaki 1 lerin sayısı tek olsun. Eğer veride çift 1 varsa parity biti 1 olur gönderilen toplam mesajdaki 1 lerin sayısı tek olur. Sistem çift olarak ayarlansaydı parity bitleri ona göre tasarlancaktı.

Gönderilen			Alınan		
10110110	0	→	10**0**10110	0	Çift Eşlik, Hata Algılanır
10110110	0	→	1**1**010110	0	Tek Eşlik, Hata Algılanamaz
10010110	1	→	10010110	**0**	Çift Eşlik, Hata Algılanır

Sistem aşağıdaki sağdaki tasarlanan donanım ile veriyi göndermeden parity bitinin ne olduğunu hesaplayarak veriye ekleyebilir. Ancak hangi parity türünün kullanılacağının set edilmesi gereklidir.



Alıcı tarafında ise aynı parity türüne set edildiği için toplam mesajdaki 1 lerin sayısını sayacaktır. Tek parity e set edilmişse ve 1 lerin sayısı çift ise hata algılanır.

Çevrimli Fazlalık Sınaması (CRC - Cyclic Redundancy Check) – Hata sezme Tekniği
Ethernet, 802.11 (WiFi), Token Ring, ATM gibi protokoller bu tekniği kullanırlar. Daha büyük boyutlu verileri aktarmak istediğimiz kullanabiliriz.

Gönderilen veri bitlerinden hesaplanan bir sınama kodu, gönderilecek veri bitlerinin sonuna eklenir. CRC kodunu hesaplamak için donanım desteği veren iletişim yongaları mevcuttur.

Üreteç fonksiyonu, CRC yönteminde anahtar kelimedir. Üzerinde çalışılmış özel bir fonksiyondur.

Ethernet, Token Ring ve Token Bus protokollerinde kullanılan üreteç fonksiyonu:

- $U(x) = x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^{8}+x^{7}+x^{5}+x^{4}+x^{2}+x+1$

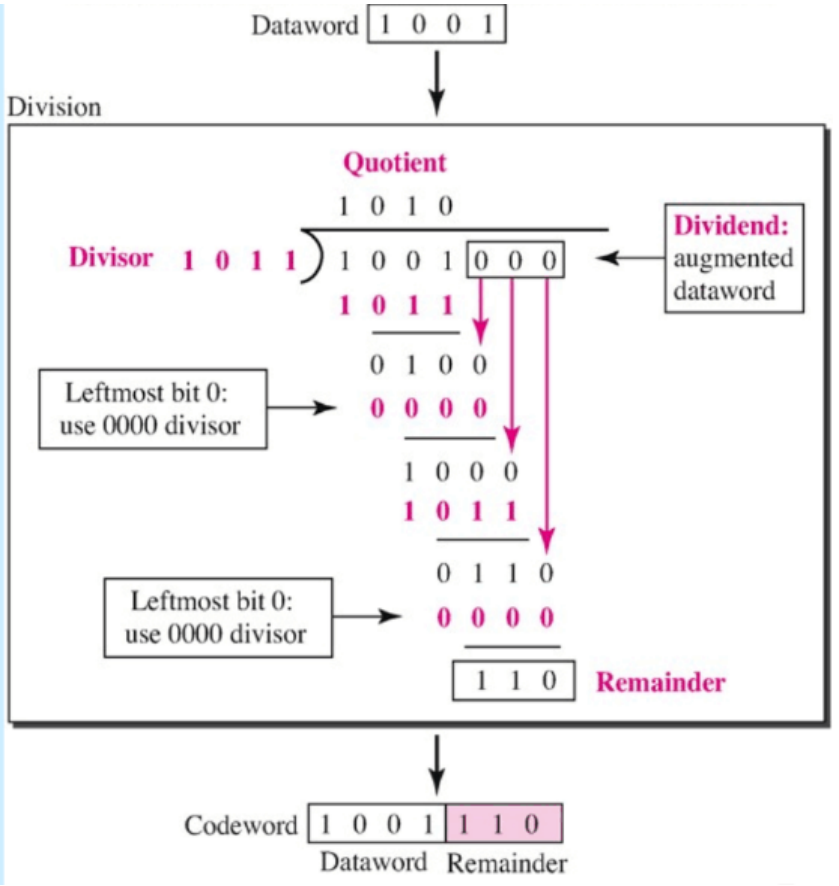
ATM başlığı içerisinde kullanılan üreteç fonksiyonu:

- $U(x) = X^{8}+x^{2}+x+1$

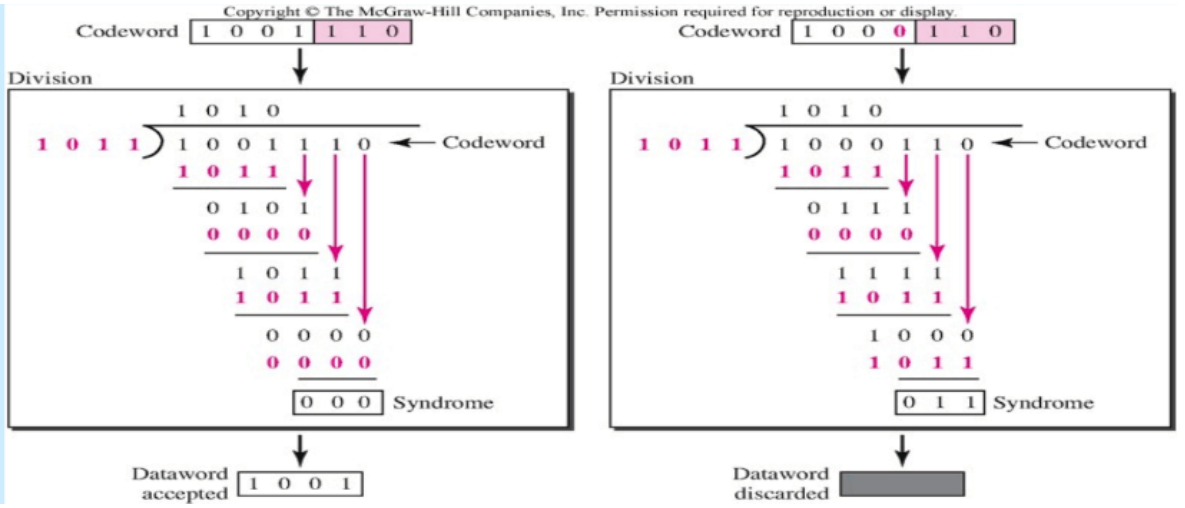
Örnek olarak geçmişten günümüze üretilmiş tüm ethernet kartları bu üreteç fonksiyonunu kullanır. Bu sayede parity deki gibi hangi eşlik biti kullanılacağının set edilmesine gerek kalmadan tüm ethernet kartları arasında konfigürasyon oluşturulmuştur.

CRC yi gerçekleştirmek için 2 farklı yöntem vardır.

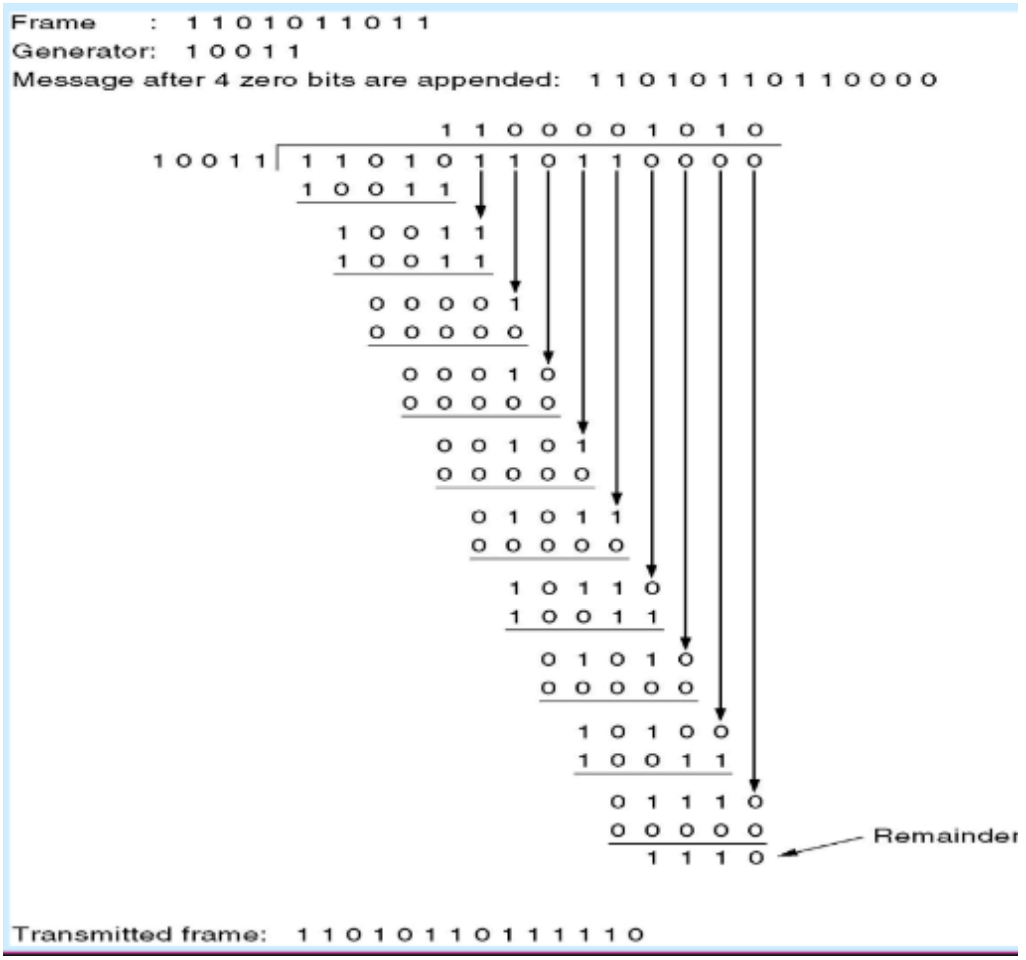
CRC – İkili Aritmetik İşlemli (derste anlatmadı)
Gönderici tarafı gönderilecek veriye (1001) 3 sıfır biti ekler. Elde edilen 7-bit (bölünen), bölen (1011) ile bölünür. Toplama ve çıkarma işlemleri XOR ile yapılır. Her aşamada bölünen ile bölen XOR'lanır. Sonuçta kalan 3-bit, gönderilecek veriye eklenerek iletim ortamına verilir.



Alıcı taraf gönderici tarafta yapılan işlemin aynısını yapar. Bölme işlemi sonucu 000 ise hata yoktur. Sıfırdan farklı ise hata vardır.

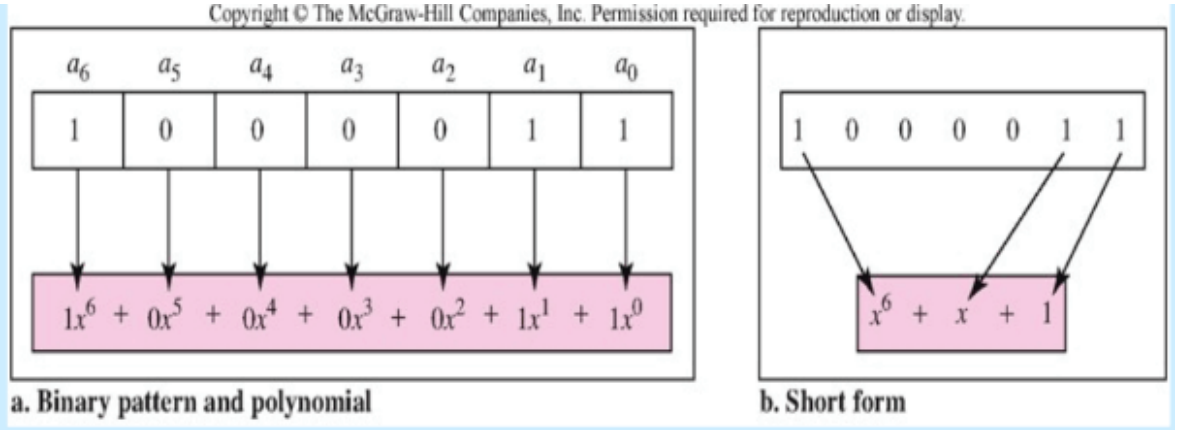


Örnek



CRC – Polinomlarla İşlem

Bir patern 0 ve 1 lerle birlikte polinom halinde gösterilerbilir. Şekilde 7-bit patern 3 terimle gösterilebilmiştir. Dolayısıyla veri bitleri polinom haline dönüştürülür.



1011100101 şeklinde bir veriyi iletmek istediğimizi düşünelim VERİ+CRC alanı nasıl olmalıdır?

Öncelikle gönderilecek veri polinoma dönüştürülür.

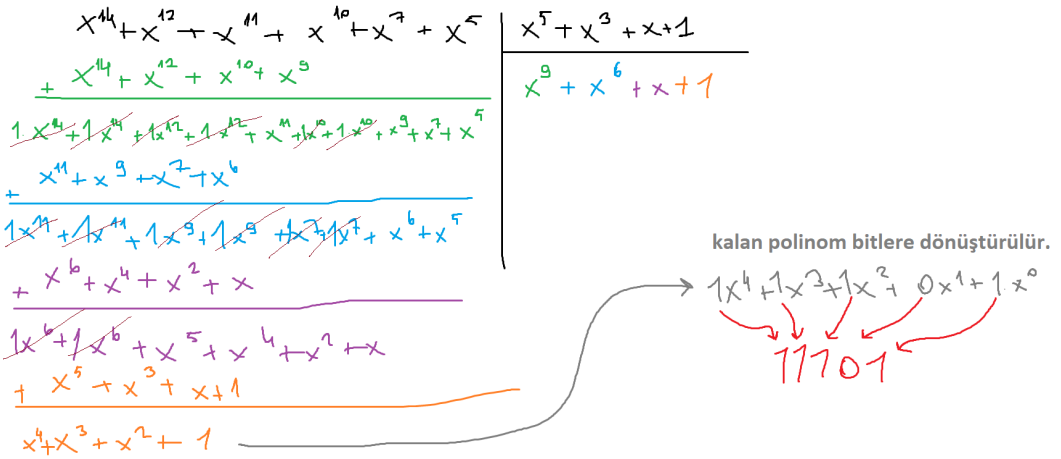
$P(x) = x^9 + x^7 + x^6 + x^5 + x^2 + 1$ daha sonra bu fonksiyon üreteç fonksiyonunun en yüksek derecesi ile çarpılmalıdır. Bu üreteç fonksiyonu soruda verilmiş olması gerekir.

Üreteç fonk. $G(x) = x^5 + x^3 + x + 1$ olsun.

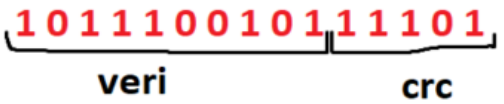
$x^5.P(x) = x^{14} + x^{12} + x^{11} + x^{10} + x^7 + x^5$

Çıkan sonuç üreteç fonksiyonuna bölünür.

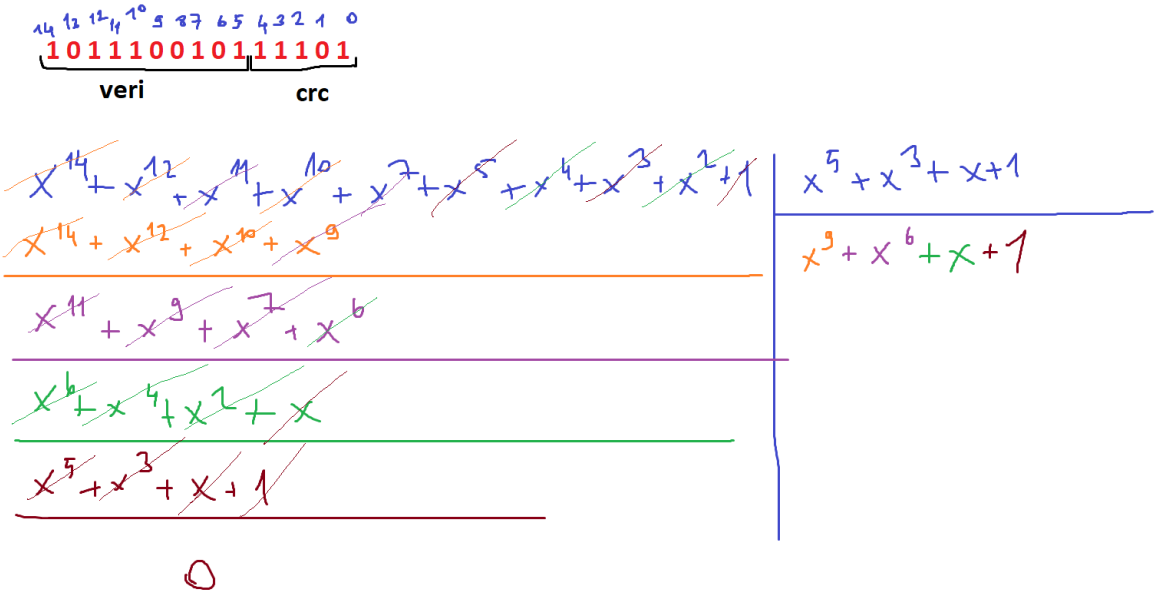
$x^{14} + x^{12} + x^{11} + x^{10} + x^7 + x^5$



Bölüm sonucu elde edilen kalan bulunur, bu değer gönderilecek veri bitlerinin sağına eklenir. Nihai olarak bulunan bit dizisi iletim ortamından alıcısına gönderilir.



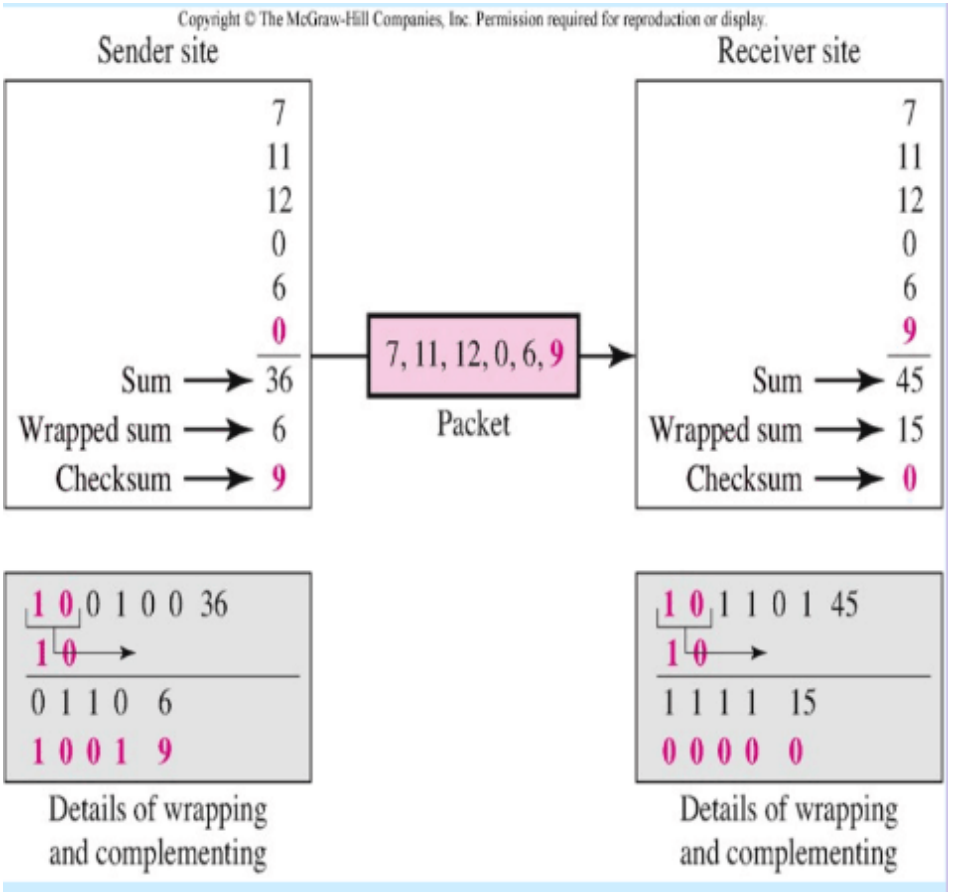
Alıcı kendisine gelen bit dizisine karşılık düşen polinomu üreteç fonksiyonuna, U(x), böler. Bölme işleminin sonucu sıfıra eşitse hatasız iletim olduğuna karar verir.



Gelen bit dizisinin sonundaki crc bitlerini atar ve geriye kalan bit dizisini bir üst katmana (servise) iletir. Eğer alıcı hatalı bit iletimi sezerse, yani bölüm sonucunda kalan sıfıra eşit değilse, veri göndericiden tekrar istenir.

Kontrol Toplamı (Checksum) – Hata Sezme Tekniği
Checksum (kontrol toplamı), TCP/IP yığınındaki protokollerde (TCP, UDP, IP, vb.) başlık ya da başlıkla beraber verinin iletiminde bir hatanın olup olmadığını anlamada kullanılır. Gönderilecek veri bitlerinin toplamı alınır ve sonra bulunan toplam veri bitlerine eklenerek öyle gönderilir.

Toplam kaç bit ile temsil edilmek isteniyorsa sağdan o kadar bit sayılır solda kalan bitler sağdan seçilen bitler ile toplanır. Elde edilen sayıya ambalajlanmış toplam denir. Bu sayının 1 e tümleyeni alınır ve mesajın en sonuna eklenerek gönderilir.



Alıcı taraf mesajı aldıktan sonra kaçar bitlik sayılar halinde almışsa hepsini toplar. Bu toplamaya bütün bitler dahildir. Daha sonra bu elde edilen toplam tekrardan gönderilen veri kaç bit ile temsil ediliyorsa o bite indirgenir. Sonucun 1 e tümleyeni alınır. Cevap 0 ise hata yoktur.

Örnek 1

4500 0095 0000 4000 3f11 **be40** ce37 ed04 c0a8 0133 bit dizisi için kontrol toplamını hesaplayınız. Bu bit dizisi bir IP başlık örneğini göstermektedir.

1. 16 bitlik alanlar şeklinde ayrılır:

   4500

   0095

   0000

   4000

   3f11

   0000 □ be40

   ce37

   ed04

   c0a8

   0133

▪

2. 16 bitlik alanlar ikili değerlere dönüştürülür ve sonra 1'e tümleyeni alınır (1'e tümleyen en son toplamlar yapıldıktan sonra da alınabilir):

Hex	Binary	1's Complement
4500	0100 0101 0000 0000	1011 1010 1111 1111
0095	0000 0000 1001 0101	1111 1111 0110 1010
0000	0000 0000 0000 0000	1111 1111 1111 1111
4000	0100 0000 0000 0000	1011 1111 1111 1111
3f11	0011 1111 0001 0001	1100 0000 1110 1110
ce37	1100 1110 0011 0111	0011 0001 1100 1000
ed04	1110 1101 0000 0100	0001 0010 1111 1011
c0a8	1100 0000 1010 1000	0011 1111 0101 0111
0133	0000 0001 0011 0011	1111 1110 1100 1100

3. 16 bitlik 1'e tümleyeni alınan haneler ikili aritmetiğe göre satır satır toplanır

Hex	Binary	1's Complement
4500	0100 0101 0000 0000	1011 1010 1111 1111
0095	0000 0000 1001 0101	1111 1111 0110 1010
		1011 1010 0110 1001
		→ 1
0000	0000 0000 0000 0000	1111 1111 1111 1111
		1011 1010 0110 1001
		→ 1
4000	0100 0000 0000 0000	1011 1111 1111 1111
		0111 1010 0110 1001
		→ 1
3f11	0011 1111 0001 0001	1100 0000 1110 1110
		0011 1011 0101 1000
		→ 1

ce37	1100 1110 0011 0111	0011 0001 1100 1000
		0110 1101 0010 0001
ed04	1110 1101 0000 0100	0001 0010 1111 1011
		1000 0000 0001 1100
c0a8	1100 0000 1010 1000	0011 1111 0101 0111
		1011 1111 0111 0011
0133	0000 0001 0011 0011	1111 1110 1100 1100
		1011 1111 0111 0011
		→ 1
		1011 1110 0100 0000
Checksum		be40

Örnek 2

153.18.8.105	
171.2.14.10	
All 0s	17 | 15
1087	13
15	All 0s
T E	S T
I N	G All 0s

10011001	00010010	→ 153.18
00001000	01101001	→ 8.105
10101011	00000010	→ 171.2
00001110	00001010	→ 14.10
00000000	00010001	→ 0 and 17
00000000	00001111	→ 15
00000100	00111111	→ 1087
00000000	00001101	→ 13
00000000	00001111	→ 15
00000000	00000000	→ 0 (checksum)
01010100	01000101	→ T and E
01010011	01010100	→ S and T
01001001	01001110	→ I and N
01000111	00000000	→ G and 0 (padding)
10010110	11101011	→ Sum
01101001	00010100	→ Checksum

Source IP address		
Destination IP address		
Zero	Protocol	UDP Length

Hamming Kodlaması – Hata Düzeltme Tekniği

Göndericinin bilgiyi bozulma durumunda tekrar göndermesinin güç olduğu bazı uygulamalarda uygun kodlama ile hatanın alıcıda düzeltilmesine çalışılır. İletim ortamında bozulabilecek bit sayısının üst sınırının bilindiği varsayılır. Bu durum;

alıcının, gönderilen veriyi, belirli bir bozulma ölçüsüne kadar düzeltebileceğini ifade eder. Bu tür yöntem, iletim yolu çok pahalı ise yada yeniden iletim büyük bir gecikme oluşturuyorsa kullanılır.

Hamming tekniği, mesafe (distance) özelliği ile beraber kullanılır. Örneğin mesafe değeri 2 ise, alıcıda 1 bitlik hatalar sezilir ve düzeltilir, 2 bitlik hatalar sadece sezilir. Bu literatürde HD,2 olarak isimlendirilir. PROFIBUS endüstriyel iletişim protokolü Hamming kodlamasını kullanır ve mesafe değeri 4'tür. Hamming kodlaması HD,4 olarak ifade edilir.

8 bitlik göndermek istediğimiz veri olsun. Bu veriyi gönderirken 4 tane de hata test biti bulup verinin uygun yerlerine yerleştirmemiz gereklidir. Test biti koyacağımız yerler 2 nin üssü olarak giden slotlardır. (1,2,4,8) geri kalan slotlara verinin bitleri yerleştirilir. Yerleştirilen bu hata test bitlerinden örneğin C1 i bulmak için C1 in slotu ikilik haline bakılır. 0001 dir. 0. sıradaki bit 1 dir. Bu yüzden geri kalan data bitlerinde 0. slotunda 1 değeri olan veri bitleri XOR lanır sonuç C1 test bitine yazılır.

Bit Konumu	Konum Numarası	Test Biti	Data Biti
12	1100	-	M8
11	1011	-	M7
10	1010	-	M6
9	1001	-	M5
8	1000	C4	
7	0111	-	M4
6	0110	-	M3
5	0101	-	M2
4	0100	C3	
3	0011	-	M1
2	0010	C2	
1	0001	C1	

$$C1 = M1_{0001} \oplus M2_{0011} \oplus M4_{0101} \oplus M5_{0111} \oplus M7_{1001} \quad (1011)$$

$$C2 = M1_{0010} \oplus M3_{0011} \oplus M4_{0110} \oplus M6_{0111} \oplus M7_{1010} \quad (1011)$$

$$C3 = M2_{0100} \oplus M3_{0101} \oplus M4_{0110} \oplus M8_{0111} \quad (1100)$$

$$C4 = M5_{1000} \oplus M6_{1001} \oplus M7_{1010} \oplus M8_{1011} \quad (1100)$$

Örnek - 00111001 bilgisi gönderilmek istensin. Bu bilgi bitlerine karşılık test bitlerini bulunuz.

hata test bitleri hesaplanır. daha sonra uygun formatta yazılıp iletim ortamına verilir. Alıcı ise alınan datayı vericinin gönderdiği yöntemlerin birebir aynısını kullanarak hata test bitlerini bulur. kendi bulduğu hata test bitleri ile kendisine gelen hata test bitlerini XOR ile toplar. Çıkan sonuç 0 ise hiç bir bit bozulmamıştır. Çıkan sonuç hangi sayı ise örneğin 3, 3. sıradaki bit bozulmuş demektir. 0 ise 1, 1 ise 0 yapılır ve hata düzeltilmiş olur.

00111001

C1 = 1 ⊕ 0 ⊕ 1 ⊕ 1 ⊕ 0 = 1
C2 = 1 ⊕ 0 ⊕ 1 ⊕ 0 = 1
C3 = 0 ⊕ 0 ⊕ 1 ⊕ 0 = 1
C4 = 1 ⊕ 1 ⊕ 0 ⊕ 0 = 0

M8	M7	M6	M5	C4	M4	M3	M2	C3	M1	C2	C1
0	0	1	1	0	1	0	0	1	1	1	1