İletişim Ağları
Communication Networks

Hazırlayan: M. Ali Akcayol
Gazi Üniversitesi
Bilgisayar Mühendisliği Bölümü

Bu dersin sunumları, "Behrouz A. Forouzan, Data Communications and Networking 4/E, McGraw-Hill, 2007." kitabı kullanılarak hazırlanmıştır.

İçerik

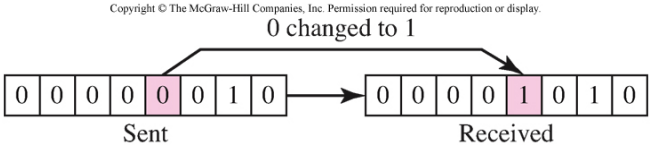
- Hata denetimi
- Blok kodlama
- Lineer blok kodlar
- Cyclic kodlar
- Checksum

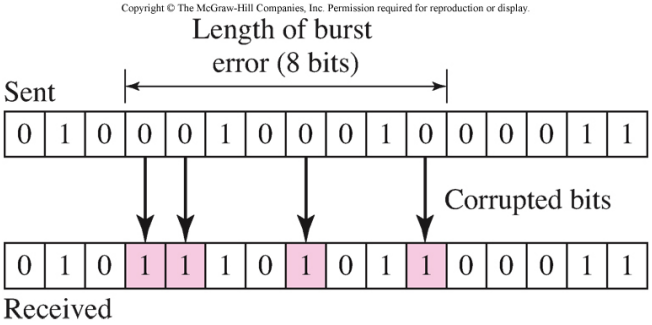
Hata denetimi

- Ağlar iki cihaz arasında **kabul edilebilir doğrulukta** veri transferi yapmak zorundadır.
- **Veri iletişim sırasında bozulabilir.**
- Bazı uygulamalarda bozulmaları kontrol etmek ve düzeltmek gerekir.
- **Bazı uygulamalar** belirli bir **hata** seviyesini **tolere** edebilir.
- Single-bit error, sadece bir bit bozulur.
- Burst error, çok sayıda bit bozulur.
- **1200 bps** hızında iletişim yapılırken eğer **0,01 sn** burst error (impulse noise) oluşmuşsa, toplam **12 bit bozulur.**
- **1 Mbps** hızında iletişim yapılırken **her bit için 1µs** gerekir.
- **Bir bitin bozulması** için gürültünün **1µs** süreye sahip olması gerekir. Gürültü genellikle daha uzun süre devam eder.

Hata denetimi

- **Burst error,** en az iki veya daha fazla bit bozulur.
- Bozulan bitler art arda olmayabilir.
- Şekilde single-bit error ve burst error görülmektedir.



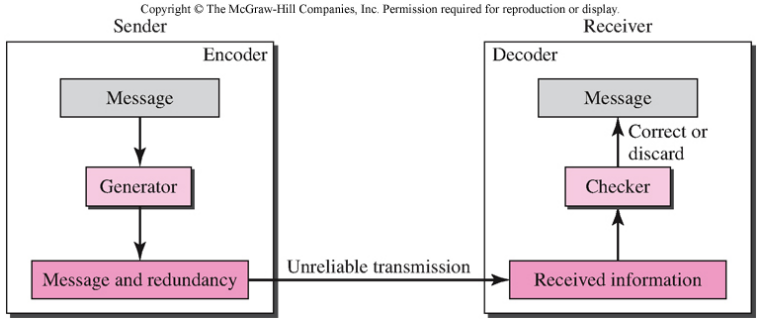


Hata denetimi

- **1 kbps** hızında iletişim yapılırken **1/100 sn noise oluşursa**, **10 bit bozulur.**
- **1Mbps** hızında iletişim yapılırsa toplam **10.000 bit bozulur.**
- Hata denetimi ve düzeltmede redundancy temel yaklaşımdır.
- Gönderilen veriyle birlikte **hata kontrol** ve **düzeltme bitleri gönderilir.**
- Redundant bitler göndericide eklenir, alıcıda çıkartılır.
- Error detection (hata denetleme), hata olup olmadığına bakılır.
- Error correction (hata düzeltme), hata düzeltilir.
- **Hata denetlemede hatanın boyutuyla ilgilenilmez.**
- **Hata düzeltme** hata denetlemeden **çok daha zordur.**
- **Hata düzeltmede** tam olarak **kaç bitte bozulma** olduğunu bilmek gerekir.

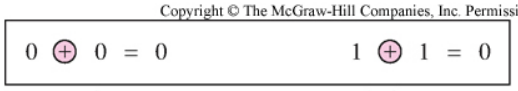
Hata denetimi

- Forward error correction, alıcı tarafından redundant bitler kullanılarak **mesajın tamamı elde edilmeye çalışılır.**
- Retransmit, alıcı hatayı denetler ve göndericiden **mesajı tekrar göndermesini ister.**
- Redundancy, blok kodlama ve convolution kodlama yöntemleri kullanılarak oluşturulur.

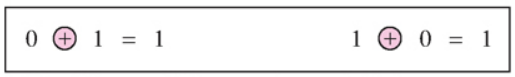


Hata denetimi

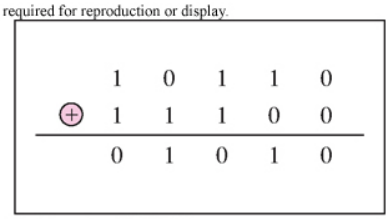
- **Modüler aritmetikte,** sınırlı aralıkta tamsayı kullanılır.
- Veriye eklenecek fazlalık bitleri genellikle **XOR** işlemiyle gerçekleştirilirler.
- XOR işlemi **aynı girişler için 0**, **farklı girişler için 1** sonucunu üretir.



a. Two bits are the same, the result is 0.



b. Two bits are different, the result is 1.



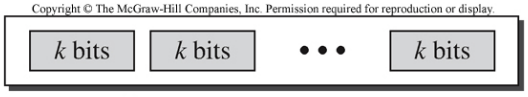
c. Result of XORing two patterns

İçerik

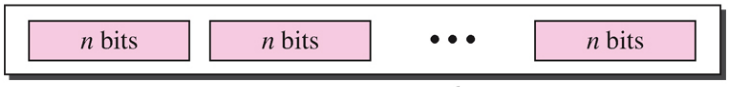
- Hata denetimi
- Blok kodlama
- Lineer blok kodlar
- Cyclic kodlar
- Checksum

Blok kodlama

- Blok kodlamada mesaj **(k-bit) bloklara** (dataword) **bölünür.**
- Her bloğa **(r-bit) redundant eklenir.**
- Oluşan **(n = k+r)** bit blok codeword olarak adlandırılır.
- Toplam **2^k** adet **dataword** ve **2^n** adet **codeword** üretilebilir. Ancak, iletişimde **2^k adet codeword kullanılır.**



2^k Datawords, each of k bits

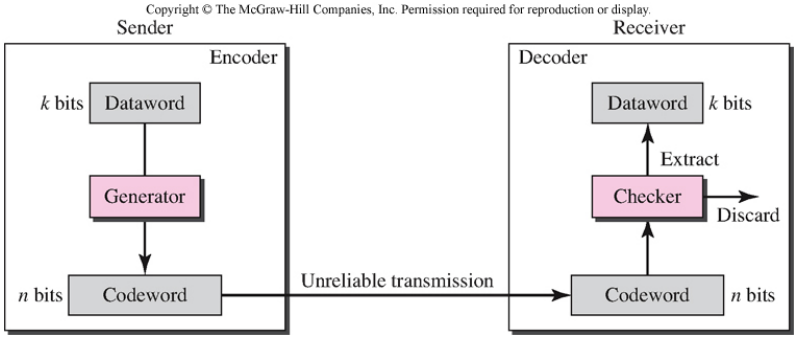


2^n Codewords, each of n bits (only 2^k of them are valid)

Blok kodlama

Hata denetimi (error detection)

- Gönderici **dataword kullanarak codeword oluşturur**.
- **Alıcı geçerli codeword listesine sahiptir.**
- **Orijinal codeword** geçerli olmayan bir tanesi ile **değiştiğinde hata algılanır.**



Blok kodlama

Hata denetimi - örnek

- **k=2** ve **n=3** için dataword ve codeword listesi aşağıdadır.

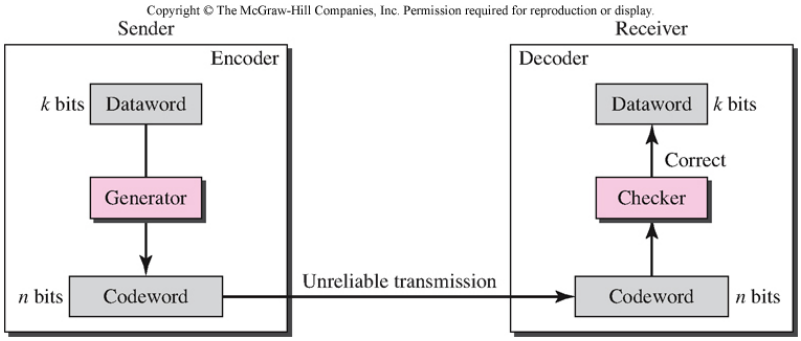
Dataword	Codeword
00	000
01	011
10	101
11	110

- Alıcı **011** geldiğinde tabloda olduğundan **geçerli** kabul eder.
- İletim sırasında codeword bozulur ve **alıcı 111 alırsa** tabloda olmadığı için **geçersiz kabul eder.**
- Codeword bozulur ve **alıcı 000 alırsa** (sağdaki iki bit 0 olmuş) tabloda olduğundan **geçerli** olarak **belirlenir**. Aslında codeword bozulmuştur !!!

Blok kodlama

Hata düzeltme (error correction)

- Hata denetiminde **alıcı** sadece **codeword'ün geçerli olup olmadığına bakar.**
- Hata düzeltmede bozulan bitler düzeltilir.
- **Hata düzeltmede** gereken **redundant bit sayısı** hata denetimine göre **daha fazladır.**



Blok kodlama

Hata düzeltme - örnek

- Önceki tabloya **iki bit daha eklenirse** aşağıdaki tablo oluşur.

Dataword	Codeword
00	00000
01	01011
10	10101
11	11110

- **Dataword 01** için **codeword 01011** olur.
- **Code bozulur** ve **alıcı 01001 alırsa** tabloda bulunamaz.
- Alıcı **1-bit bozulma olduğunu tahmin eder** ve ilk sıradan **kontrol etmeye başlar.**
- 2.sıradaki hariç diğerleri 2-bit farklıdır.
- **2.sıradaki 1-bit farklı** olduğundan **codeword ikinci sıradakiyle değiştirilir.**

Blok kodlama

Hamming uzaklığı (hamming distance)

- Hamming distance **iki word arasında farklı bit sayısını** gösterir.
- **x** ve **y** word'leri **arasındaki hamming uzaklığı d(x,y)** şeklinde gösterilir.
- Hamming distance **XOR** kapısıyla hesaplanır.
- Elde edilen **1 sayısı Hamming uzaklığını** gösterir.

- (000 ⊕ 011 = 011)
  d(000, 011) = 2 olur.

- (10101 ⊕ 11110 = 01011)
  d(10101, 11110) = 3 olur.

Blok kodlama

Minimum Hamming uzaklığı

- Tüm çiftler içinde **en küçük Hamming uzaklığıdır**.
- Aşağıdaki codeword'ler için Hamming distance değerleri eşittir.

d(000, 011) = 2	d(000, 101) = 2	d(000, 110) = 2
d(011, 101) = 2	d(011, 110) = 2	d(101, 110) = 2

- Bir kodlama yöntemi **(C) 3 parametreyle ifade edilir:**
  - **Codeword boyutu** (n)
  - **Dataword boyutu** (k)
  - **Minimum Hamming uzaklığı** (d_{min})

Blok kodlama

Minimum Hamming uzaklığı

- **Soldaki tablodaki kodlama** C(3,2) ve d_{min}=2.
- **Sağdaki tablodaki kodlama** C(5,2) ve d_{min}=3.
- İletişim sırasında **s adet hata** olursa, gönderilen ve alınan codeword'ler arasındaki **Hamming uzaklığı s olur.**
- Bir kodlamada **s adet hatayı algılamak için,** minimum Hamming **uzaklığının s+1 olması gerekir.**

Dataword	Codeword
00	000
01	011
10	101
11	110

Dataword	Codeword
00	00000
01	01011
10	10101
11	11110

Blok kodlama

Minimum Hamming uzaklığı - örnek

- Aşağıdaki tabloda **minimum Hamming uzaklığı = 2**'dir.
- **1-bit hata denetimi garanti edilir.**

Dataword	Codeword
00	000
01	011
10	101
11	110

- Eğer 3.satırdaki codeword gönderilirse **ve 1-bit hata olursa**, alınan codeword tablodakilerin **hiçbirisiyle aynı değildir.**
- Eğer **2-bit hata olursa**, alınan hatalı codeword tablodaki codeword'lerden **birisiyle aynı olabilir** ve hata algılanamaz.

Blok kodlama

Minimum Hamming uzaklığı - örnek

- Aşağıdaki tabloda **minimum Hamming uzaklığı = 3**'tür.
- **2-bit hata denetimi garanti edilir.**

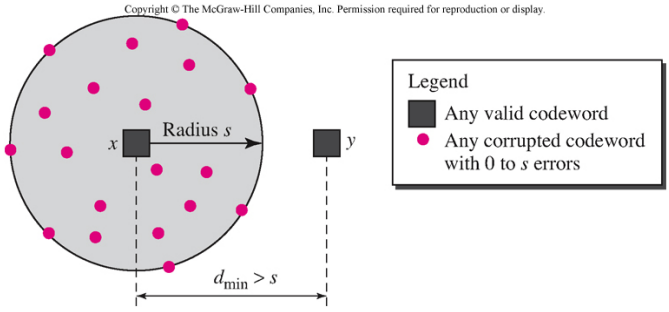
Dataword	Codeword
00	00000
01	01011
10	10101
11	11110

- Eğer geçerli bir codeword gönderilirse ve **2-bit hata olursa**, alınan codeword tablodakilerin **hiçbirisiyle aynı olmaz.**
- Aynı tabloda **3-bit hataların bazıları** da tabloda **geçerli bir codeword oluşturabilir.**

Blok kodlama

Minimum Hamming uzaklığı – hata denetimi

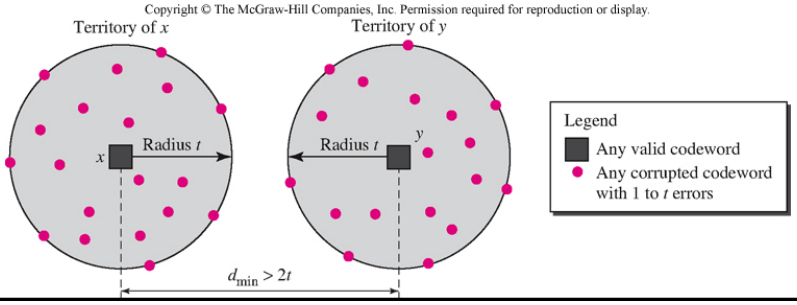
- Aşağıdaki şekilde bir **x codeword** ve etrafında **s yarıçapında codeword kümesi vardır.**
- Minimum Hamming distance değeri s'den büyüktür. **(d_{min}=s+1)**
- Diğer geçerli olan tüm codeword'ler çemberin dışındadır.



Blok kodlama

Minimum Hamming uzaklığı – hata düzeltme

- Alıcı **codeword'ün geçersiz olduğunu algılarsa**, tablodan hangi codeword'ün gönderildiğini bulmaya çalışır.
- Her geçerli **codeword**, kendi etrafında **t yarıçapında bir çember oluşturur.**
- Alıcı aldığı geçersiz codeword'ün yerine **ait olduğu çemberin ortasındaki codeword'ü** gerçek codeword olarak alır.
- **Hata düzeltmede d_{min} > 2t olmalıdır.**



İçerik

- Hata denetimi
- Blok kodlama
- Lineer blok kodlar
- Cyclic kodlar
- Checksum

Lineer blok kodlar

- **İki codeword** arasındaki **XOR işlemi,** geçerli bir **codeword oluşturursa,** bu kodlar lineer blok kodlar olarak adlandırılır.
- Aşağıdaki tabloda lineer blok kodlar görülmektedir.
- Lineer blok kodlamada, **minimum Hamming uzaklığı, tümü 0'dan farklı** codeword'lerdeki **minimum 1 sayısıdır.**
- Soldaki tabloda 0'dan farklı codeword'lerdeki 1 sayısı **2, 2** ve **2**'dir **(d_{min}=2).**
- Sağdaki tabloda **3, 3** ve **4**'tür **(d_{min}=3).**

Dataword	Codeword
00	000
01	011
10	101
11	110

Dataword	Codeword
00	00000
01	01011
10	10101
11	11110

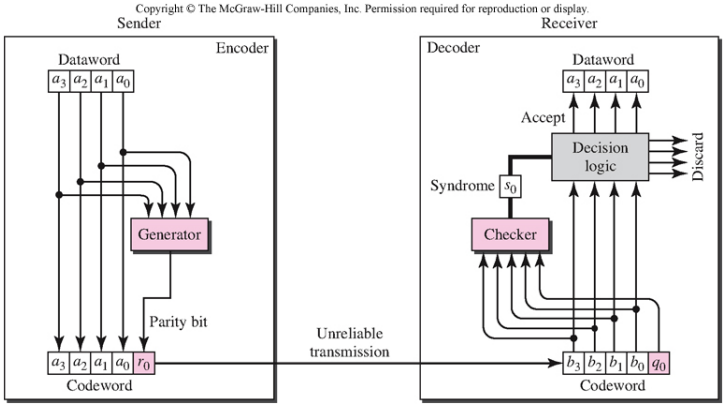
Lineer blok kodlar

- Simple parity-check code, **hata denetim kodudur.**
- **k-bit dataword n-bit codeword**'e dönüştürülür **(n=k+1)**.
- Eklenen bit ile toplam **1 sayısı çift** veya **tek sayıda** yapılır.
- **d_{min}=2**'dir. Hata düzeltme yapılamaz, 1-bit hata denetimi yapılır. Tabloda **C(5,4) parity check kodu** verilmiştir.

Dataword	Codeword	Dataword	Codeword
0000	00000	1000	10001
0001	00011	1001	10010
0010	00101	1010	10100
0011	00110	1011	10111
0100	01001	1100	11000
0101	01010	1101	11011
0110	01100	1110	11101
0111	01111	1111	11110

Lineer blok kodlar

- Şekilde 4-bit dataword'e **1-bit parity biti eklenmiştir.**
- Parity bit **5-bit codeword'deki 1 sayısını çift** yapmaktadır.
- Dataword'deki **4-bit mod 2'ye** göre toplanarak ek hesaplanır. **($r_0 = a_3+a_2+a_1+a_0$)**
- Alıcıda, **sonuç** (syndrome) **0 ise doğrudur, 1 ise hatalıdır.**



Lineer blok kodlar

Hamming kodları

- Hamming kodları $\mathbf{d_{min}=3}$ ile tasarlanabilir.
- $\mathbf{d_{min}=3}$ için **2-bit hatayı bulur** ve **1-bit düzeltir.**
- **m** kontrol için eklenen bit sayısıdır.
- Tabloda **m=3, n=7** ve **k=4**'tür. Kod **C(7, 4)**, $\mathbf{d_{min}=3}$.

Dataword	Codeword	Dataword	Codeword
0000	0000000	1000	1000110
0001	0001101	1001	1001011
0010	0010111	1010	1010001
0011	0011010	1011	1011100
0100	0100011	1100	1100101
0101	0101110	1101	1101000
0110	0110100	1110	1110011
0111	0111001	1111	1111111

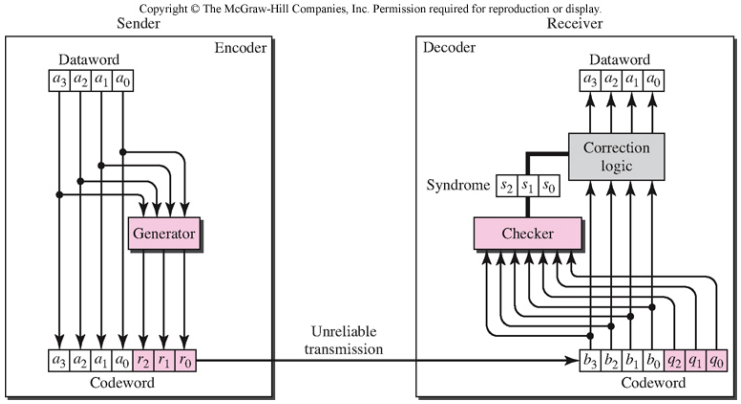
Lineer blok kodlar

Hamming kodları - örnek

- **C(7, 4)**, $\mathbf{d_{min}=3}$ için gönderici ve alıcıda syndrome hesaplanır.

$$r_0 = a_2 + a_1 + a_0 \quad \text{modulo-2}$$
$$r_1 = a_3 + a_2 + a_1 \quad \text{modulo-2}$$
$$r_2 = a_1 + a_0 + a_3 \quad \text{modulo-2}$$

$$s_0 = b_2 + b_1 + b_0 + q_0 \quad \text{modulo-2}$$
$$s_1 = b_3 + b_2 + b_1 + q_1 \quad \text{modulo-2}$$
$$s_2 = b_1 + b_0 + b_3 + q_2 \quad \text{modulo-2}$$



Lineer blok kodlar

Hamming kodları - örnek - devam

- **3 bit syndrome** ile **8 farklı durum** oluşturulur.
- **Bu durumlar** hata durumunu ve **hata varsa hatalı biti gösterir.**

Syndrome	000	001	010	011	100	101	110	111
Hata	Yok	q_0	q_1	b_2	q_2	b_0	b_3	b_1

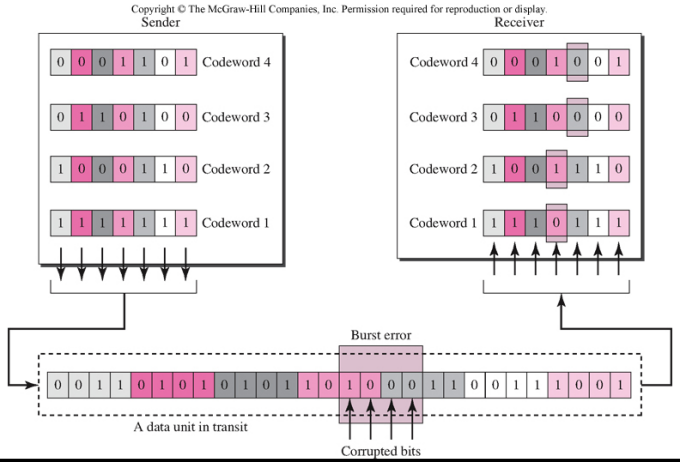
Örnek

- **0100 dataword**, **0100011 codeword** olarak gönderiliyor. Codeword **0100011** alınırsa, **syndrome 000** olur ve hata yoktur.
- **0111 dataword**, **0111001 codeword** olarak gönderiliyor. **Codeword 0011001** alınıyor. **Syndrome 011** hesaplanır. Tabloya göre **b_2 hatalıdır**. b_2 0'dan 1'e değiştirilir ve dataword **0111** olur.
- **1101 dataword**, **1101000 codeword** olarak gönderiliyor. **Codeword 0001000** alınıyor **(2-bit hatalı)**. **Syndrome 101** hesaplanır. Tabloya göre **b_0 hatalıdır**. b_0 1'den 0'a değiştirilir ve **dataword 0000** olur. (yanlış !!!)

Lineer blok kodlar

Hamming kodları - performans

- Hamming kodları 1-bit hata düzeltir ve 2-bit hata denetler.
- Ayrıca, **burst error denetleyebilir.**
- **Burst error** oluşan kısım **farklı codeword'lere dağıtılır.**



İçerik

- Hata denetimi
- Blok kodlama
- Lineer blok kodlar
- Cyclic kodlar
- Checksum

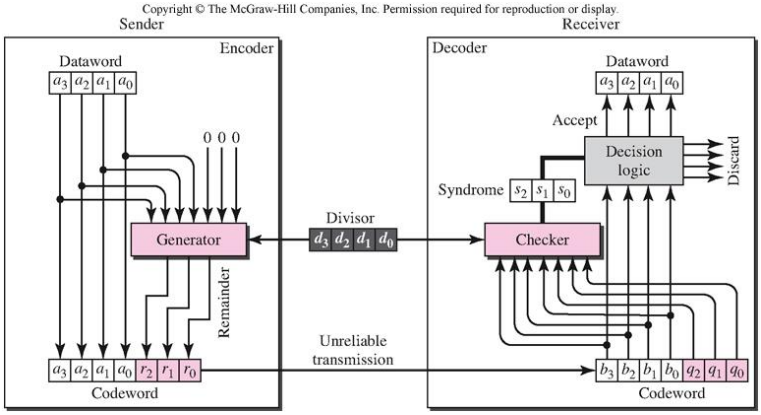
Cyclic kodlar

- **Cyclic kodlar özel lineer blok kodlardır** (rotate edilebilir).
- **Rotate edildikten sonra** oluşan **kod** geçerli **codeword'tür.**
- Codeword **1011000 left-rotate** yapılırsa, codeword **0110001 olur** ve **geçerlidir.**
- Cyclic Redundancy Check (CRC), **LAN** ve **WAN**'larda kullanılır.

Dataword	Codeword	Dataword	Codeword
0000	0000000	1000	1000101
0001	0001011	1001	1001110
0010	0010110	1010	1010011
0011	0011101	1011	1011000
0100	0100111	1100	1100010
0101	0101100	1101	1101001
0110	0110001	1110	1110100
0111	0111010	1111	1111111

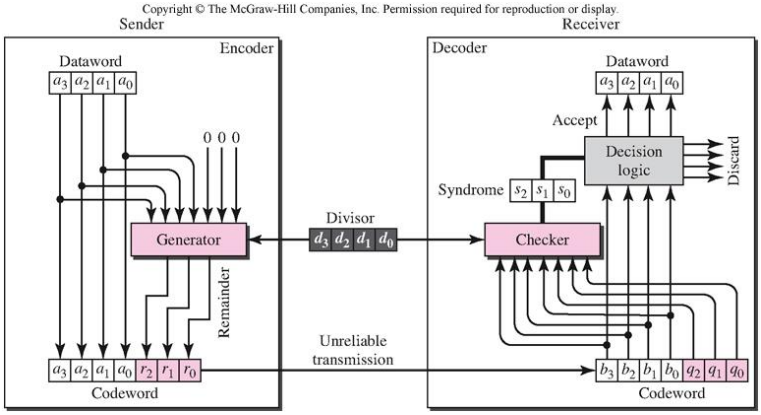
Cyclic kodlar

- Kodlayıcıda 4-bit **dataword** (k=4) ve 7-bit **codeword** (n=7).
- Dataword'e **000 eklenerek** 7-bit **codeword** elde edilmiştir.
- Generator dataword'ü, belirlenen **divisor ile mod 2'ye göre böler.** Bölüm atılır ve **kalan dataword'e eklenerek codeword oluşturulur.**



Cyclic kodlar

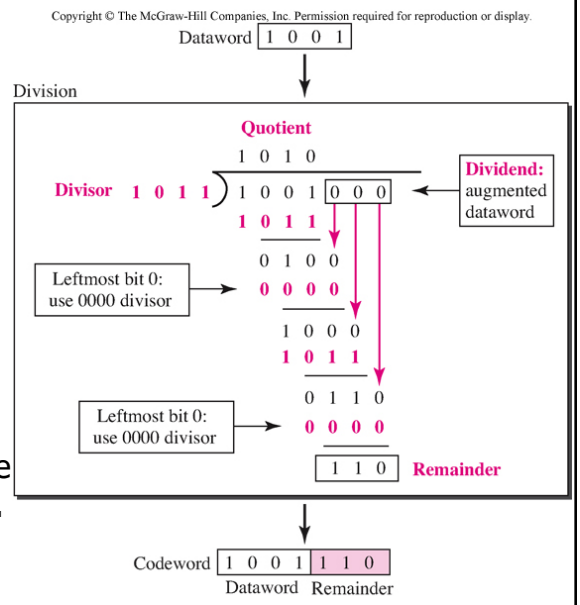
- Alıcıda **checker** ile **kalan kısım tekrar hesaplanır.**
- **$s_2s_1s_0$ syndrome** oluşturulur.
- **Decision logic**, gelen 4-bit dataword'ü doğru ise alır, yanlış ise atar.



Cyclic kodlar

Encoder

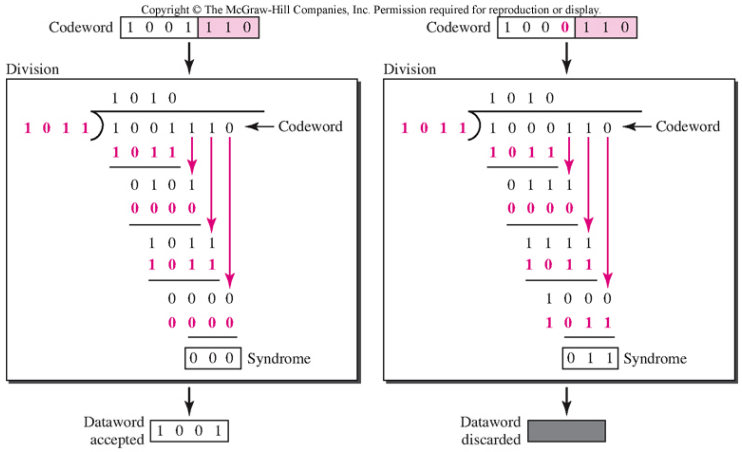
- Encoder dataword'e **n-k adet 0 ekler.**
- Elde edilen **7-bit codeword, divisor ile bölünür.**
- Her aşamada **dividend ile divisor XOR'lanır.**
- En soldaki bit 0 ise divisor 0000 alınır.
- Sonuçta, **kalan 3-bit dataword'e eklenir** ve **codeword elde edilir.**



Cyclic kodlar

Decoder

- **Decoder** ile encoder'da yapılan **bölme işlemi aynen yapılır.**
- Bölmeden **kalan syndrome oluşturur**.
- **Syndrome 000 ise hata yoktur.**



İçerik

- Hata denetimi
- Blok kodlama
- Lineer blok kodlar
- Cyclic kodlar
- Checksum

Checksum

- Checksum **İnternet'te yaygın kullanılmaktadır.**
- **Gönderilen sayıların toplamı alınır** ve birlikte gönderilir. **(7, 11, 12, 0, 6)** için **(7, 11, 12, 0, 6, 36).**
- **Alıcı** gelen **sayıları toplar** ve **gelen toplamla karşılaştırır.**
- **Aynı ise data alınır, değilse atılır.**
- Checksum değeri one's complement (bir'in tümleyeni) alınarak elde edilir (tüm bitler terslenir.).

Örnek

- Toplam değeri 21 ise sadece 4 bitle gösterilir.

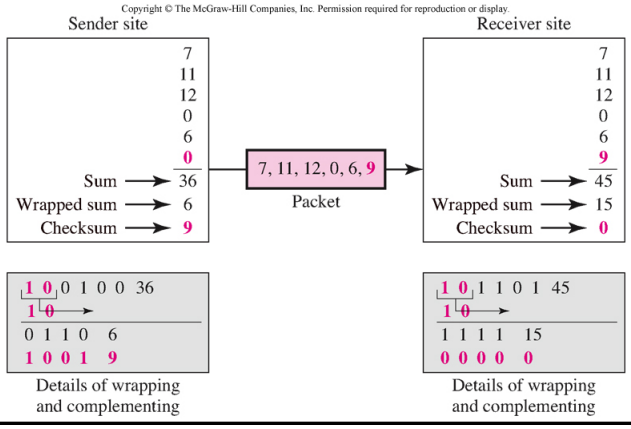
  21 = 10101, en soldaki 1 sağa alınır ve toplanır.

  0101 + 1 = 0110 (6) olur.

  1001 (9) -> checksum

Checksum

- Şekilde **4-bit checksum** kullanılarak **gönderici** ve **alıcı** tarafta yapılan **işlemler görülmektedir** (İnternet 16-bit).
- **Gönderen parçaların hepsini 1 tümleyene göre toplar.**
- **Toplamın 1 tümleyeni** ile **checksum elde edilir.**
- Alıcı aynı işlemleri tekrarlar. **Checksum 0 ise hata yoktur.**



Ödev

- İletişim ağlarında hata düzeltme kodları hakkında detaylı bir araştırma ödevi hazırlayınız.