

B. Bölüm : HATA SEZME VE DÜZELTME

Error Detection and Correction

Veri paketleri iletilirken bazı bitler bozulabilir. Birçok uygulamada bu kabul edilemez bir durundur. Bu nedenle iletisim yapılırken, bozulma olup olmadığını anasılmas için "Error Detection" teknikleri; bu hataları düzeltmek için de "Error Correction" teknikleri kullanılır.

Error Detection

- Echoplex
- Parity Check
- LRC - Longitudinal Redundancy Check
- CRC - Cyclic Redundancy Check

Error Correction

- Hamming Code
- ARQ - Automatic Repeat Request

ERROR DETECTION / HATA SEZME

1. Echoplex

↳ Bu yöntem genellikle merkeze bağlı terminalde kullanılır.

Terminalde tuşa basıldığında bu tuşa karşı düşen karakter merkeze gelir.

Merkez tarafından bu karakter terminalde geri yansıtınarak kullanıcının hataları görmesi sağlanır.

2. Parity Check

↳ Bir veri katona içindeki tek sayıdaki hatayı sezmek için kullanılır.

Bu amaçla katona bir parity bit eklenir

Parity Bit → Even Parity
→ Odd Parity

1'lerin sayısı ile parity bit değeri uyusmuyorsa hata olduğu anasılır.

Eğer hata bulunan bit sayısı çift ise hata ALGILANAMAZ.

3. LRC - Longitudinal Redundancy Check

↳ Bu teknik bİndelerde çok alt katar raçen veri katarı içindeki hatayı识mek için kullanılır. Bu yöntemle veri katarını oluşturan alt veri katarlarının aynı pozisyonlarındakı bitlere Even ya da Odd parity eklenerek bir sınama katarı elde edilir. Bu da değerlerine eklenerek gönderilir.

Bu yöntemde de çift sayıda bitin bozulması sezikmez

$$\begin{array}{ccccccc} b_{0,0} & \cdots & \cdots & b_{0,k-1} & p_0 \\ b_{1,0} & \cdots & \cdots & b_{1,k-1} & p_1 \\ \vdots & \ddots & \ddots & \vdots & \vdots \\ b_{j-1,0} & \cdots & \cdots & b_{j-1,k-1} & p_{j-1} \\ p_0 & \cdots & \cdots & p_{k-1} & p \end{array}$$

$p_0 \cdots p_{k-1} p \rightarrow$ Sınama Karakterleri

- 1) Karakterler k bit uzunluğunda olsun
- 2) Her parçaya giden karakterleri gönderme sırasına göre alt alta yazalım
- 3) Karakterlerin parity bitlerini de gönderme sırasına göre soldan sağa doğru yazalım.
- 4) Sınama karakterlerinin her biri her iki ilgili sütunun eşlik katını kontrol etse.

4. CRC - Cyclic Redundancy Check

↳ Bu yöntemde CRC olarak adlandırılan ve gönderilen veri katarından hesaplanan bir sınama katarı, bu veri katarının sonuna eklenir.

CRC katarını hesaplamak için donanım desteği veren iletişim yangoları mevcuttur.

CRC Katarı Hesaplaması:

- ① Veri katarı $P(x)$ adı verilen bir polinom ile gösterilir. Polinomin katsayıları, gönderilen veri katarının ilgili pozisyonlarının bit değeridir. n bit için :

$$P(x) = b_{n-1} \cdot x^{n-1} + b_{n-2} \cdot x^{n-2} + \cdots + b_1 \cdot x^1 + b_0 \cdot x^0$$

- ② Bu $P(x)$ polinomu x^r ile çarpılır.

Bu işlem sonucu elde edilen yeni polinoma karsi düşer bit katarı, aslında önceki bit katarı ile onun sonuna ilişkilendirilmiş p tane 0 bitinde olusur.

$$10100101100 \dots 00$$

$\underbrace{\quad}_{\text{p tane } 0}$

④ $x^P \cdot P(x)$ polinomu, P . dereceden $G(x)$ adı verilen bir generating polinomuna bölünür

Üreteç polinomları belirli hata sezarı özellikleri bulunan polinomlardır. Bazi üreteç polinomları verilmiştir.

$$x^{16} + x^{15} + x^2 + 1$$

$$x^{16} + x^{12} + x^5 + 1$$

$$x^{12} + x^{11} + x^3 + x^2 + x + 1$$

⑤ $x^P \cdot P(x) / G(x)$ bölümünü gösterici hesaplar

Bölüm $\rightarrow Q(x)$

Kalan $\rightarrow R(x)$

$$x^P \cdot P(x) = Q(x) \cdot G(x) + R(x)$$

İşlemlerde modulo 2 aritmetiği kullanılır. Bu durumda toplama ve çıkarma aynı işlemi tanımlar

$$0 + 0 = 0; 0 - 0 = 0$$

$$0 + 1 = 1; 0 - 1 = 1$$

$$1 + 0 = 1; 1 - 0 = 1$$

$$1 + 1 = 0; 1 - 1 = 0$$

$$x^P \cdot P(x) - R(x) = Q(x) \cdot G(x)$$

$$x^P \cdot P(x) + R(x) = Q(x) \cdot G(x)$$

ÖRNEK

Bit dizisi $\Rightarrow 1010010111$ eklenerek CRC biti nedir?

① Polinom $P(x) \Rightarrow x^9 + x^7 + x^4 + x^2 + x + 1$

Üreteç Fonksiyonu $\Rightarrow x^4 + x^2 + x + 1$

② Bu $P(x)$ polinomu x^4 ile çarpılır.

$$x^4 \cdot P(x) = x^{13} + x^{11} + x^8 + x^6 + x^5 + x^4$$

③ $x^4 \cdot P(x)$ polinomu. 4. dereceden seçilen üreteç Fonksiyonuna bölünse

$$\begin{array}{r}
 x^{13} + x^{11} + x^8 + x^6 + x^5 + x^4 \\
 - x^{13} - x^{11} - x^{10} - x^9 \\
 \hline
 - x^{10} - x^8 - x^7 - x^6 - x^5 - x^4 \\
 - x^{10} - x^8 - x^7 - x^6 \\
 \hline
 - x^9 - x^7 - x^5 - x^4 \\
 - x^9 - x^7 - x^5 - x^4 \\
 \hline
 - x^6 - x^4 \\
 - x^6 - x^4 - x^3 - x^2 \\
 \hline
 \text{kalan} \Rightarrow x^3 + x^2
 \end{array}$$

$x^4 + x^2 + x + 1$
 $x^9 + x^6 + x^5 + x^2$ bölüm

$$\begin{aligned}
 P(x), P_{G(x)} &= x^{13} + x^{11} + x^8 + x^6 + x^5 + x^4 && - \text{bölen} \\
 G(x) &= x^4 + x^2 + x + 1 && - \text{bölen} \\
 Q(x) &= x^9 + x^6 + x^5 + x^2 && - \text{bölüm} \\
 R(x) &= x^3 + x^2 && - \text{kalan}
 \end{aligned}$$

④ Bu işlemler sonucu gönderilecek bit dizisi

$$\begin{aligned}
 Q(x).G(x) &= x^P P_{G(x)} + R_{G(x)} \\
 &\Rightarrow x^4(x^9 + x^6 + x^5 + x^2 + x + 1) + x^3 + x^2 \\
 &= x^{13} + x^{11} + x^8 + x^6 + x^5 + x^4 + x^3 + x^2
 \end{aligned}$$

$$\begin{array}{r}
 1 \oplus 1 \otimes 0 \oplus 1 \otimes 1 \otimes 1 \\
 \hline
 \text{bilyi bitleri}
 \end{array}
 \quad
 \begin{array}{r}
 1 \otimes 0 \oplus 1 \\
 \hline
 \text{CRC bits}
 \end{array}$$

- $E(x), G(x)$ 'in tam katı değilse, bölüm alıcıda ④ olmayacağından alıcı hatayı farkeder
- Tek sayıda bit bozulduğunda $E(x), G(x)$ 'e tam bölünmez çift sayıda bit bozulduğunda ise $E(x)$ 'in $G(x)$ 'e tam bölünme olasılığı düşüktür
- Gevresel olaylar gibi faktörlerde bitlerde toplu bozulmalar olabilir. Bu senaryoya burst error adı verilir.

ddd... ~~xx~~ --- ~~xx~~ ddd

ERROR CORRECTION / HATA DÜZELTME

→ Alıcı, bu yöntemle veriyi belirli bir bozulma ölçüsune kadar düzeltebilir. (Veri yeniden gönderilmeden)
 Bu yöntem işlem malzeti doğurur ve zaman yeniden veri göndermek daha mantıklı olabilir. Sadece iletim yolu çok pahalıysa ya da büyük gecikme yasası yorsa ~~hata~~ hata düzeltme kullanılır.

1. Hamming Code

↳ Alıcıya ulaşan ve belirli bir sinyeye karşı düşer şekilde kod, yolda bir ölçüde bozulmuş bile olsa asıl kod elde edilebilir.

Kod sözcüğünün yolda en çok j bitinin bozulması durumunda hata düzeltme mümkünse koda j bit bağımlılığı olan kod denir.

Hamming(7,4) Kodunda 4 adet bilgi biti, 3 adet sırname biti kullanılarak 7 bitlik kod sözcükleri elde edilir.

Kod Sözcüğü : $I_1 I_2 I_3 I_u C_1 C_2 C_3$

$C_1 \rightarrow I_1 I_2 I_u$ 'e ilişkin çift eşlik biti

$C_2 \rightarrow I_1 I_3 I_u$ 'e ilişkin çift eşlik biti

$C_3 \rightarrow I_2 I_3 I_u$ 'e ilişkin çift eşlik biti

1 bitlik hata düzeltilebilir
2 bitlik hata düzeltilebilir

$$C_1 = I_1 \oplus I_2 \oplus I_u$$

$$C_2 = I_1 \oplus I_3 \oplus I_u$$

$$C_3 = I_2 \oplus I_3 \oplus I_u$$

$I_1 I_2 I_3 I_u C_1 C_2 C_3$

1 ⊕ 1 ⊕ 1 ⊕ 1 ⊕ 1

~~M=U~~
 $M=U$ olsun

$$n = \binom{u}{2} + \binom{u}{3} + \binom{u}{4} \Rightarrow 6 + u + 1 = 11 \quad \text{tane bilgi biti kullanılır}$$

$$C_1 = I_1 \oplus I_2 \oplus I_3 \oplus I_7 \oplus I_8 \oplus I_9 \oplus I_{11}$$

$$C_2 = I_1 \oplus I_u \oplus I_5 \oplus I_7 \oplus I_8 \oplus I_{10} \oplus I_{11}$$

$$C_3 = I_2 \oplus I_u \oplus I_6 \oplus I_7 \oplus I_9 \oplus I_{10} \oplus I_{11}$$

$$C_4 = I_3 \oplus I_5 \oplus I_6 \oplus I_8 \oplus I_9 \oplus I_{10} \oplus I_{11}$$

İletişim Başarı Hesabı

Bir iletişim kanalının sahip olduğu band genişliğinin ne kadarının gerçek veri aktarımında kullanıldığını gösteren

Kanalın birliği bit aktarımıyla gerçek performansı ifade edilir.

R : Bit aktarım hızı (bps)

E : Kodlama verimi (Code Rate)

B_i : Kodlanan bir karakterin kod sözcüğündeki bilgi biti sayısı

B_t : " " " " " " fazlalık bit sayısı

BER : Bir bitin yolda bozulma olasılığı

P_{no} : Bloğun alıcısına hatasız ulaşma olasılığı

N : Blok içindeki toplam bit sayısı

Kodlama verimi $E = B_i / B_t$

Hatasız ulaşma olasılığı $P_{no} : (1 - BER)^N$ (hatasız hatlarda $P_{no} \approx 1$ olur)

Bilgi biti aktarım hızı $TRIB = E \cdot R \cdot P_{no}$

$TRIB = (B_i / B_t) \cdot R \cdot (1 - BER)^N$