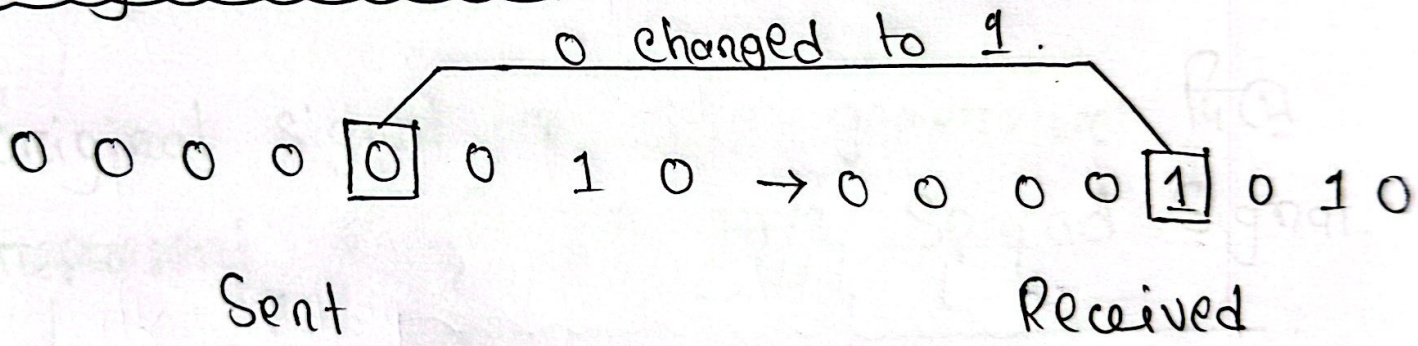
Error Detection and Correction

Chapter: 10

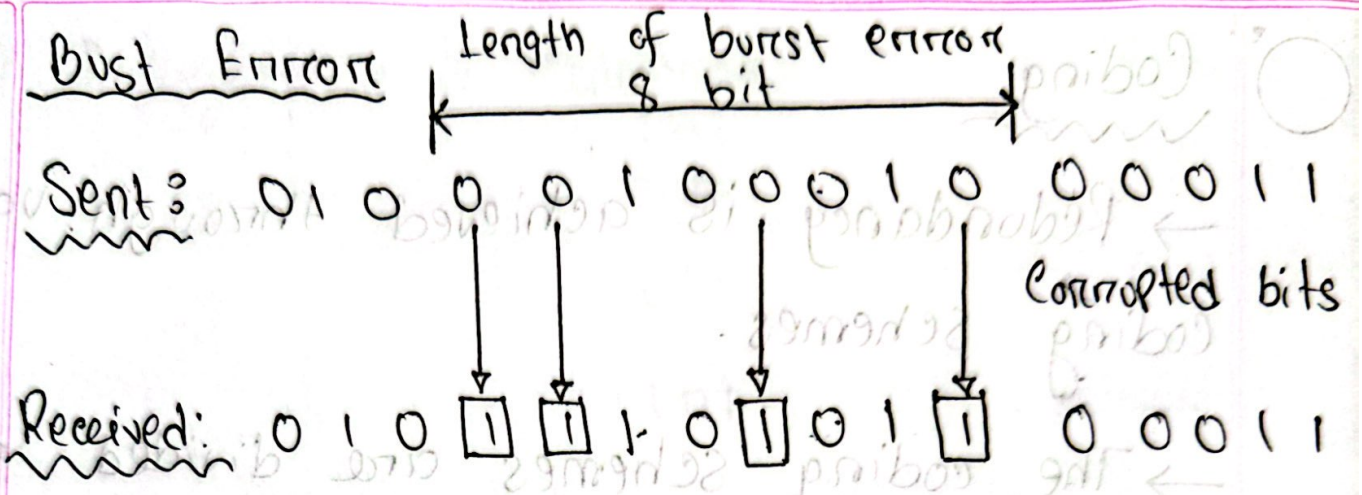
→ Data can be corrupted during transmission. Some applications require that errors be detected and corrected.

Single bit Error



⇒ A burst error means that 2 or more bits in the data unit changed during the sending and the receiving time.

Bust Error



Redundancy

→ the central concept in error detection and correction.

→ The redundant bits are added by the sender and removed by the receiver

→ Their presence helps the receiver to detect and correct corrupted bits.

Coding

→ Redundancy is achieved through various coding schemes.

→ The coding schemes are divided into two broad categories

→ Block coding

→ Convolution coding. (এটি পড়া লাগবে না)

Block Coding

In block coding, we divide our message into blocks, each of k bits, called datawords. We add r redundant bits to each block to make the length ($n = k + r$). The resulting n-bit blocks are called codewords.

Datawords (K)	Codewords (N)
00	000
	011
01	101
10	
11	110

Let, us assume that $K = 2$ and $N = 3$.

→ Assume the sender encodes the dataword 01 as 011 and sends it to the receiver. Consider the following case:

i) The receiver receives 011. It is a valid codeword. The receiver extracts the dataword 01 from it.

ii) The codeword is corrupted during transmission, and 111 is received. This is

not a valid codeword and is discarded.

iii) The codeword is corrupted during transmission, and 000 is received. This is a valid codeword. The receiver incorrectly extracts the dataword 00.

→ An error-detecting code can detect only the types of errors for which it is designed; other types of error may remain undetected.

XORing of two single bits or two words

$0 + 0 = 0$ } $0 + 1 = 1$ }

$1 + 1 = 0$ } $1 + 0 = 1$ }

Two bits are different, the result is 1.

two bits are same, the result is 0.

$$\begin{array}{r} 1\ 0\ 1\ 1\ 0 \\ +\ 1\ 1\ 1\ 0\ 0 \\ \hline 0\ 1\ 0\ 1\ 0 \end{array}$$

Result of XORing two patterns.

Hamming distance 2.

⇒ The Hamming distance between two words is the number of differences between corresponding bits.

(25) Hamming distance

i) The Hamming distance d(000, 011) is

```
  0 0 0
+ 0 1 1
-------
  0 ! !
```

→ distance is 2.

ii) The Hamming distance d(10101, 11110)

```
  1 0 1 0 1
+ 1 1 1 1 0
-----------
  0 ! 0 ! !
```

→ distance is 3.

Ex: 10.6 $d(00000, 01011)$

$$\begin{array}{r} 0\ 0\ 0\ 0\ 0 \\ \oplus\ 0\ 1\ 0\ 1\ 1 \\ \hline 0\ 1\ 0\ 1\ 1 \end{array} \longrightarrow \text{distance} = 3$$

$d(00000, 10101)$

$$\begin{array}{r} 0\ 0\ 0\ 0\ 0 \\ \oplus\ 1\ 0\ 1\ 0\ 1 \\ \hline 1\ 0\ 1\ 0\ 1 \end{array} \longrightarrow \text{distance} = 3$$

$d(00000, 11110)$

$$\begin{array}{r} 0\ 0\ 0\ 0\ 0 \\ \oplus\ 1\ 1\ 1\ 1\ 0 \\ \hline 1\ 1\ 1\ 1\ 0 \end{array} \longrightarrow \text{distance} = 4$$

$d(01011, 10101)$

$$\begin{array}{r} 0\ 1\ 0\ 1\ 1 \\ \oplus\ 1\ 0\ 1\ 0\ 1 \\ \hline 1\ 1\ 1\ 1\ 0 \end{array} \longrightarrow \text{distance} = 4$$

d(01011, 11110)

```
  01011
⊕ 11110
-------
  10101  → distance 3
```

d(10101, 11110)

```
  10101
⊕ 11110
-------
  01011  → distance 3.
```

The dmin in this case is 3

→ gurantee the detection of up to s errors, dmin = s+1

→ gurantee the correction of up to t errors, dmin = 2t+1.

Cyclic Codes (CRC)

Cyclic codes are special linear block codes with one extra property. In a cyclic code, if a codeword is cyclically shifted (rotated), the result is another codeword.

$a_0\ a_1\ a_2\ a_3\ a_4$
1 0 1 1 0

left shift → $b_0\ b_1\ b_2\ b_3\ b_4$
0 1 1 0 1

$b_0 = a_1$
$b_1 = a_2$
$b_2 = a_3$
$b_3 = a_4$
$b_4 = a_0$

⇒ We need Cyclic Redundancy Check (CRC)

Cyclic Redundancy Check (CRC)

K = 4

Dataword = a_3 a_2 a_1 a_0 (1 1 1) → dividend

we add ↑

Divisor = d_3 d_2 d_1 d_0

Codeword = **Remainder + Dataword** = New message sent to receiver

N bits

(N = 7)

Divisor Size = n − k + 1

Dividend Size = n − k → Codeword − Dataword

{ Encoder and Decoder ସ୍ଥ Divisor same ଥାଏ always. }.

Figure 10.15

Division in CRC encoder

Dataword 1001

Divisor 1011

```
          1010
      ------------
1011 ) 1001000
       1011
       ----
        0100
        0000
        ----
        1000
        1011
        ----
        0110
        0000
        ----
         110  → reminder
```

add করলাম (Divisor-1) bit. Normally dividend = n-k. N জানি না।

leftmost 0 ← 0100

তাই Divide করতে হবে 0000 দিয়ে

leftmost সরছে 0 ← 0110

Dividing by 0000

We are doing XOR operation

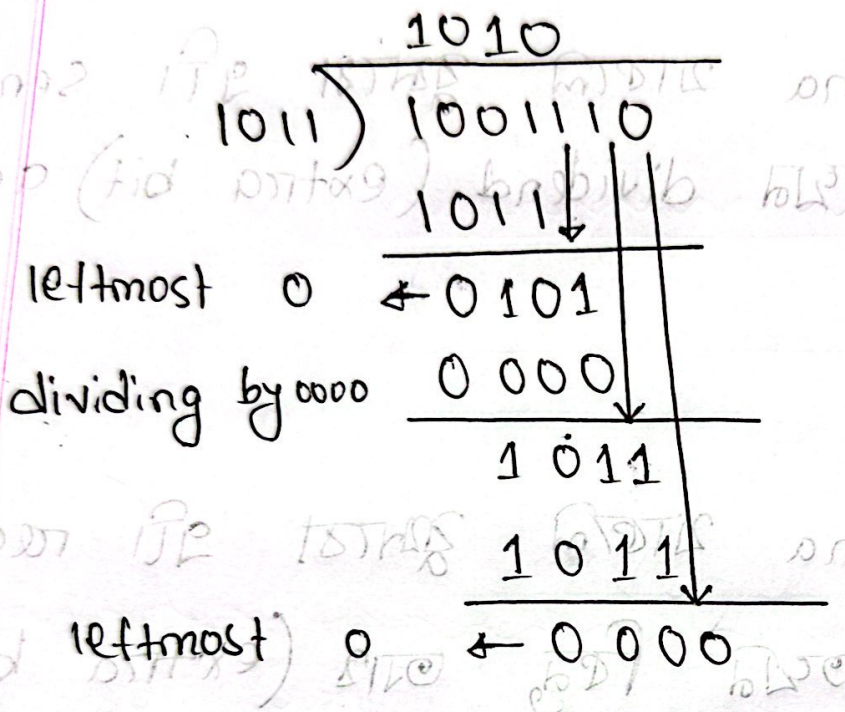
∴ Codeword = Dataword + Reminder

= 1001 + 110

= 1001110 → এটা প্রথমে receiver এ পাঠাবে

Codeword = 1001110

Divisor = 1011.



000 → যদি 000 আসে তাহলে বুঝাবে যে কোন error হয় নাই।

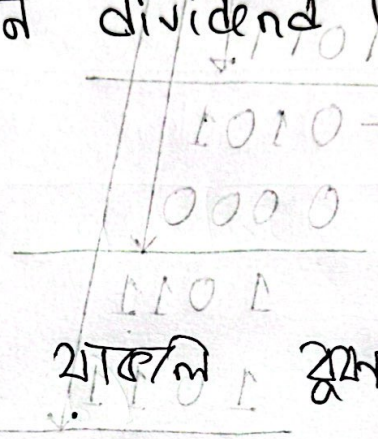
So, dataword is accepted.

↳ (1001) 110 → redundant. So, এটা consider হবেনা না।

{যদি remainder 000 না হয়, then Dataword rejected}.

আমাদের খেয়াল রাখতে হবে Question এ কি লেখা, Dataword নাকি codeword.

→ Dataword lekha থাকলে বুঝাবে এটা sender এর side এ, তখন dividend (extra bit) add করবে,



→ Codeword lekha থাকলে বুঝাবে এটা receiver এর side এ, তখন কিন্তু আর (extra bit) add করবে না, Sender এর side থেকে (Dataword + reminder) = Codeword কে consider করবে,

Figure 10.21 A polynomial to represent a binary word

* $x^6 + x + 1$.

Highest power = 6.

$\therefore x^6 + x^5 + x^4 + x^3 + x^2 + x^1 + x^0$

$\Rightarrow 1x^6 + 0x^5 + 0x^4 + 0x^3 + 0x^2 + 1x^1 + 1x^0$

$\Rightarrow 1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1 \quad 1.$

* $x^4 + x^2 + x.$

Highest power : 4

$\therefore x^4 + x^3 + x^2 + x^1 + x^0$

$\Rightarrow 1x^4 + 0x^3 + 1x^2 + 1x^1 + 0x^0$

$\Rightarrow 1 \quad 0 \quad 1 \quad 1 \quad 0$

(b) Checksum

The checksum is used in the internet by several protocols although not at the data link layer. However,

Data is a list of five 4 bit numbers

= (7, 11, 12, 0, 6)

At the end we add another 4 bit number which is the sum of 5 numbers

= 7, 11, 12, 0, 6, 36.

When the receiver receives the data. from the sum, we substract the remaining numbers consecutively from the summation and at the end if we get zero then we will know that no error occured

And, if we don't get 0, then we will get an error.

→ We can make the job of the receiver easier if we send the negative (compliment) of the sum, called the checksum. In this case, we send (7, 11, 12, 0, 6, −36). The receiver can add all the numbers received (including the checksum). If the result is 0, it assumes no error, otherwise there is an error.

Fig 10.24

Sender

7
11
12
0
6
0

Sum → 36

Wrapped Sum → 6

Checksum → 9

36 binary → 100100

4 bit

∴ 0100
⊕ 10
0110 → 6

Now, reverse
1001 → 9

8 4 2 1

Receiver

7
11
12
0
6
9

Sum → 45

Wrapped sum → 15

Checksum → 0

45 binary → 101101

⊕ 10
1111 → 15

reverse 0000 → 0

If we get 0, then we assume no error.

decimal to binary করতে হবে again.

if we get 4 bit, then we don't need to add the extra bit on perform the XOR operation.