Error Detection and Correction

ICT 2156

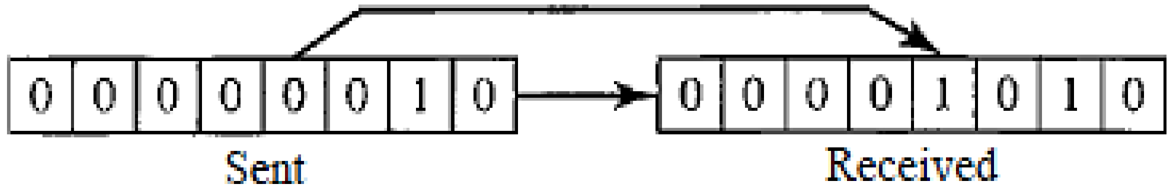
IPSITA UPASANA

Error

- What is an Error?
- Types of Errors
    - Single-bit errors
    - Burst errors

Types of Errors

- Single-Bit Error:
  - only 1 bit of a **given data unit**.



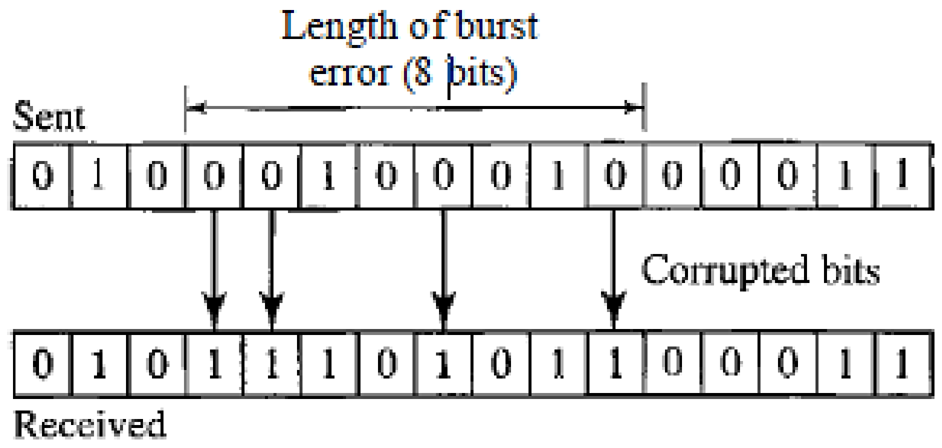
ASCII Start of text

ASCII Line Feed

  - Caused by white noise.

Types of Errors

- Burst Error:
  - 2 or more bits of a given data unit.
  - Number of bits affected depends on the data rate and duration of noise.
  - Caused by impulse noise and fading.



Which is least likely to occur in serial data transmission? Why?

Detection versus Correction

- Redundancy
- Detection
- Correction
    - Number of bits affected.
    - Location of corrupted bits.
    - **Number of errors and size of message.**

- **Methods of error correction:** Forward Error Correction and Retransmission.

Why do we need Error Detecting Techniques?

Error Probabilities

- P_b : Bit Error Rate
- P_1 : Frame with e=0
- P_2 : Frame with undetected errors (>=1), with error-detecting algorithms.
- P_3 : Frame with detected bit errors (>=1) and 0 undetected bit errors, with error-detecting algorithms.
- Assumption: P_b is constant and independent for each bit.

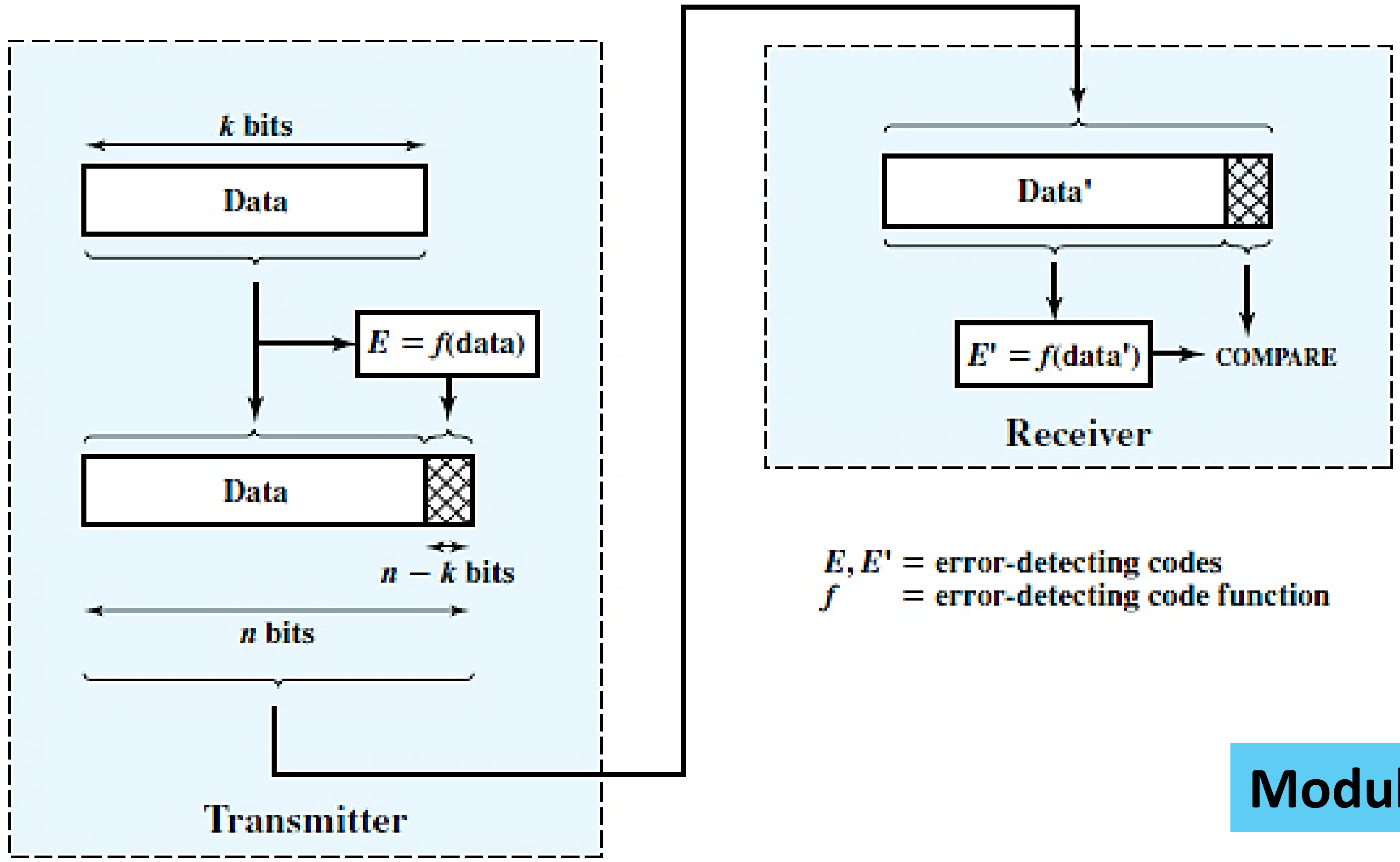
- If there are no means to detect errors, $P_3 = 0$

$P_1 = (1-P_b)^F$ and $P_2 = 1-P_1$ Results?

Why do we need Error Detecting Techniques?

- Requirements
  - $BER < 10^{-6}$
  - One frame with an undetected bit error should occur per day on a continuously used 64-kbps channel.
- F=1000 bits.
- How much data transmitted in a day?
- Desired Frame Error Rate?
- If $BER = 10^{-6}$, P_1 =? and P_2 =?

Error Detection Process (Coding)



E, *E'* = error-detecting codes
f = error-detecting code function

Modulo 2 Arithmetic

Cyclic Redundancy Check (CRC)

TRANSMITTER

k bits of data/message.

Transmitter generates (*n*-*k***)** bit sequence to append to the message.

Frame Check Sequence (FCS)

What is the frame size now?

RECEIVER

Receiver divides the incoming frame by a predetermined divisor, ***P***.

No remainder ➔No error.

Cyclic Redundancy Check (CRC)

- Modulo 2 Arithmetic
- Polynomials
- Digital Logic

Modulo 2 Arithmetic

- Uses binary addition with no carries: which is just the exclusive-OR (XOR) operation.
- Binary subtraction with no carries is also interpreted as the XOR operation.
- Example

Cyclic Redundancy Check (CRC)

T = n-bit frame to be transmitted

D = k-bit block of data, or message, the first k bits of T

F = $(n - k)$-bit FCS, the last $(n - k)$ bits of T

P = pattern of $n - k + 1$ bits; this is the predetermined divisor

- For no error: *T*/*P* should have no remainder.
- How is P chosen?

  Depending on the type of error expected.
- If E: error pattern with 1s in positions where errors occur, then $\mathbf{T_r = T \oplus E}$
- What if T_r is divisible by P?

CRC

- Exact bit pattern for P depends on the type of errors expected. Both the high- and low-order bits of P must be 1.
- If there is an error the receiver will fail to detect the error if and only if is divisible by P, which is equivalent to E divisible by P.

CRC modulo 2

- Message, D: 1010 0011 11 (k = 10 bits)
- P: 110101
- FCS, R =?

CRC Polynomials

- Use dummy variable X, with binary coefficients.

$$\frac{X^{n-k}D(X)}{P(X)} = Q(X) + \frac{R(X)}{P(X)}$$
$$T(X) = X^{n-k}D(X) + R(X)$$

- Message, D: 1010 0011 11 (k = 10 bits)
- P: 110101

CRC: Errors

- An error $E(X)$ will only be undetectable if it is divisible by $P(X)$.
- The following errors are not divisible by a suitably chosen $P(X)$ and hence are detectable:
    - All single-bit errors, if $P(X)$ has more than one nonzero term
    - All double-bit errors, as long as $P(X)$ is a special type of polynomial, called a **primitive polynomial**, with maximum exponent L, and the frame length is less than $2^L - 1$.
    - Any odd number of errors, as long as $P(X)$ contains a factor $(X+1)$.
    - Any burst error for which the length of the burst is less than or equal to (n-k)
    - A fraction of error bursts of length (n-k+1); the fraction equals $1- 2^{-(n-k-1)}$
    - A fraction of error bursts of length greater than (n-k+1); the fraction equals $1- 2^{-(n-k)}$

CRC

- If all error patterns are considered equally likely, then for a burst error of length r+1 the probability of an undetected error is $1/2^{r-1}$
- For a longer burst, the probability is $1/2^{r}$, where r is the length of the FCS.
- Four versions of P(X) are widely used:

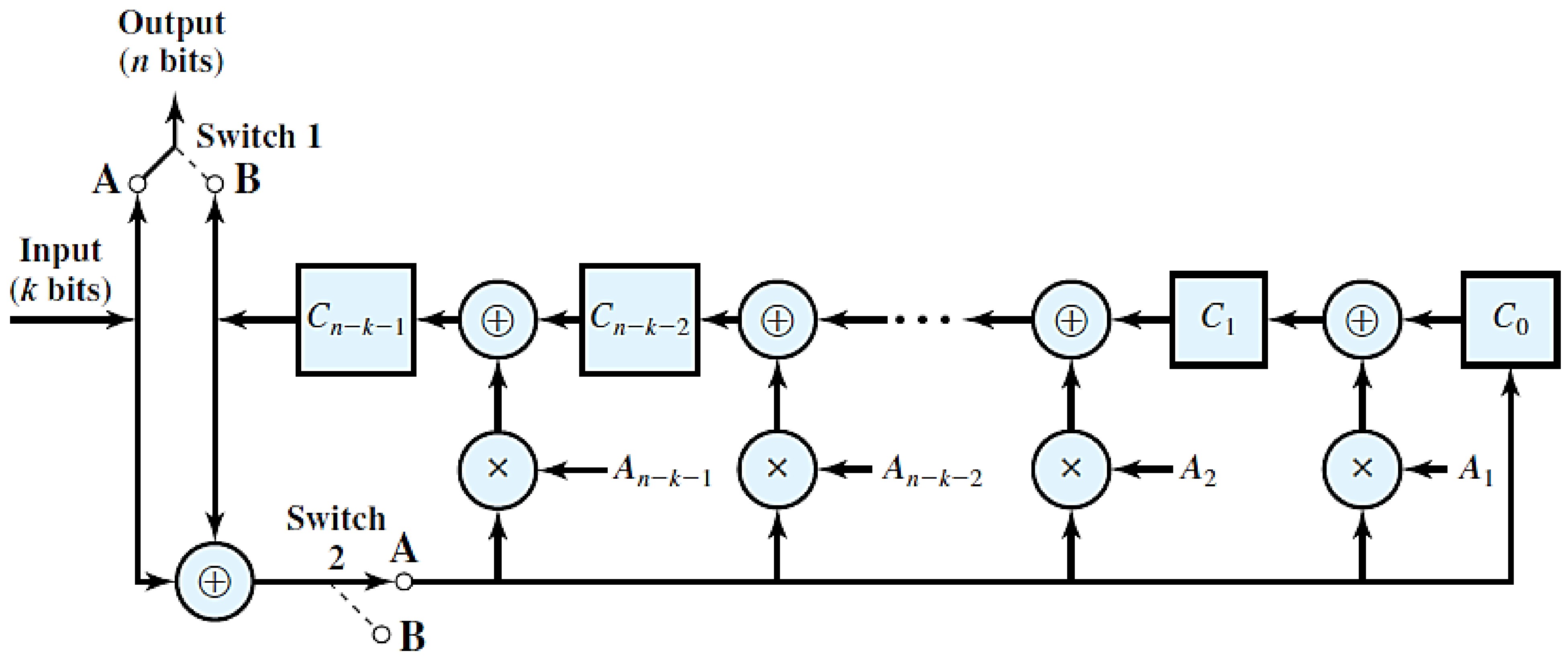
$$\begin{aligned}
\text{CRC-12} &= X^{12} + X^{11} + X^{3} + X^{2} + X + 1 \\
\text{CRC-16} &= X^{16} + X^{15} + X^{2} + 1 \\
\text{CRC-CCITT} &= X^{16} + X^{12} + X^{5} + 1 \\
\text{CRC-32} &= X^{32} + X^{26} + X^{23} + X^{22} + X^{16} + X^{12} + X^{11} \\
&\quad + X^{10} + X^{8} + X^{7} + X^{5} + X^{4} + X^{2} + X + 1
\end{aligned}$$

- The CRC-12 system is used for transmission of streams of 6-bit characters and generates a 12-bit FCS. Both CRC-16 and CRC-CCITT result in a 16-bit FCS.

CRC Digital Logic

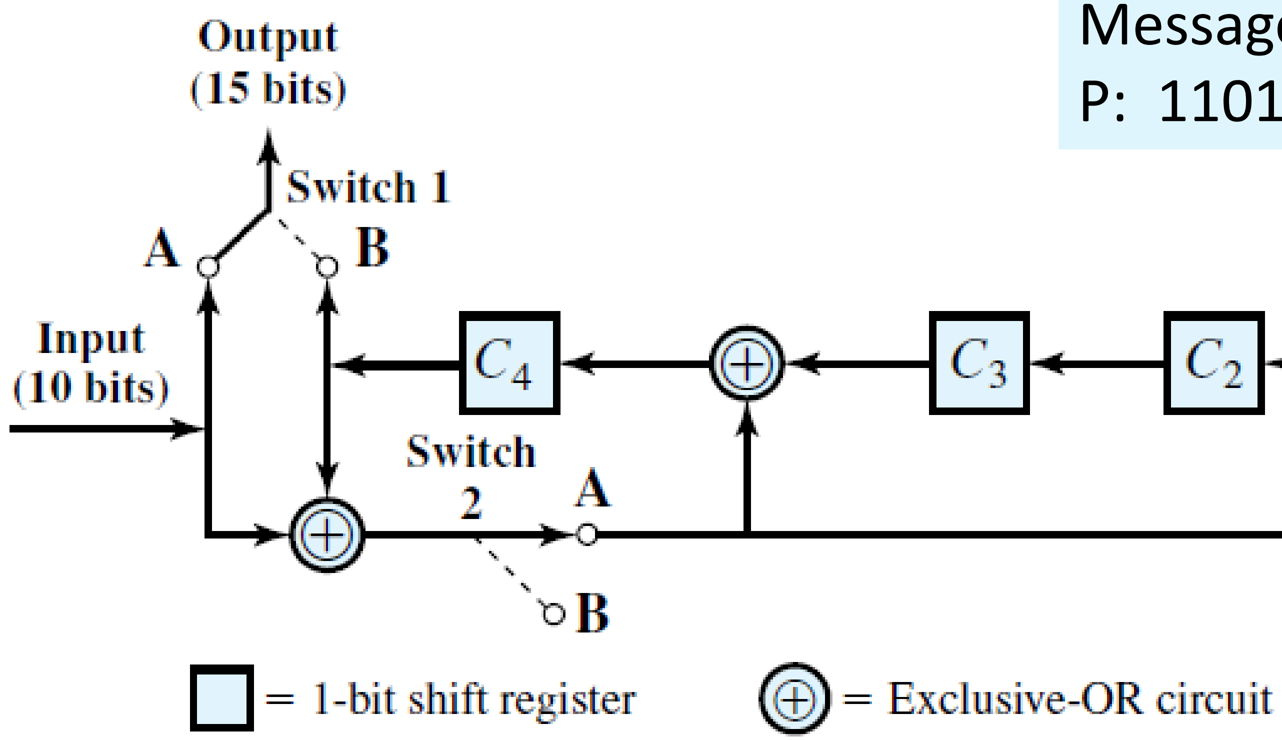
- Consists of XOR gates and shift registers (1-bit storage devices).
- Circuit Implementation:
    1. The register contains (n-k) bits, equal to the length of the FCS.
    2. There are up to (n-k) XOR gates.
    3. The presence or absence of a gate corresponds to the presence or absence of a term in the divisor polynomial, P(X), excluding the terms 1 and X^{n-k}.

CRC Hardware Implementation: Architecture



CRC Hardware Implementation

Message, D: 1010 0011 11 (k = 10 bits)
P: 110101



Step	C4= C4 XOR I XOR C3	C3=C2	C2= C4 XOR I XOR C1	C1=C0	C4= C4 XOR I	Input, I
Initial	0	0	0	0	0	1
1	1	0	1	0	1	0
2	1	1	1	1	1	1
3	1	1	1	1	0	0
4	0	1	0	0	1	0
5	1	0	0	1	0	0
6	1	0	0	0	1	1
7	0	0	0	1	0	1
8	1	0	0	0	1	0
9	1	0	1	1	1	1
10	0	1	1	1	0	FCS

CRC Polynomial

$$
\begin{array}{rl}
 & X^9 + X^8 + X^6 + X^4 + X^2 + X \quad \leftarrow Q(X) \\
P(X) \rightarrow X^5 + X^4 + X^2 + 1 \,\big/ & \overline{X^{14} \qquad X^{12} \qquad\qquad X^8 + X^7 + \qquad X^5} \quad \leftarrow X^5D(X) \\
 & \underline{X^{14} + X^{13} + \qquad X^{11} + \qquad X^9} \\
 & \qquad X^{13} + X^{12} + X^{11} + \qquad X^9 + X^8 \\
 & \qquad \underline{X^{13} + X^{12} + \qquad X^{10} + \qquad X^8} \\
 & \qquad\qquad X^{11} + X^{10} + X^9 + \qquad X^7 \\
 & \qquad\qquad \underline{X^{11} + X^{10} + \qquad X^8 + \qquad X^6} \\
 & \qquad\qquad\qquad X^9 + X^8 + X^7 + X^6 + X^5 \\
 & \qquad\qquad\qquad \underline{X^9 + X^8 + \qquad X^6 + \qquad X^4} \\
 & \qquad\qquad\qquad\qquad X^7 + \qquad X^5 + X^4 \\
 & \qquad\qquad\qquad\qquad \underline{X^7 + X^6 + \qquad X^4 + \qquad X^2} \\
 & \qquad\qquad\qquad\qquad\qquad X^6 + X^5 + \qquad X^2 \\
 & \qquad\qquad\qquad\qquad\qquad \underline{X^6 + X^5 + \qquad X^3 + \qquad X} \\
 & \qquad\qquad\qquad\qquad\qquad\qquad X^3 + X^2 + X \leftarrow R(X)
\end{array}
$$

Parity Check : Error-Detecting Code

- Even Parity (Synchronous Transmission)
- Odd Parity (Asynchronous Transmission)
- Single-bit **error-detecting** code.

- Example: Considering Even Parity, find the code words for the following data words.

0001, 0010, 0011, 0100, 0101, 0110, 0111

Any Pattern/formula?

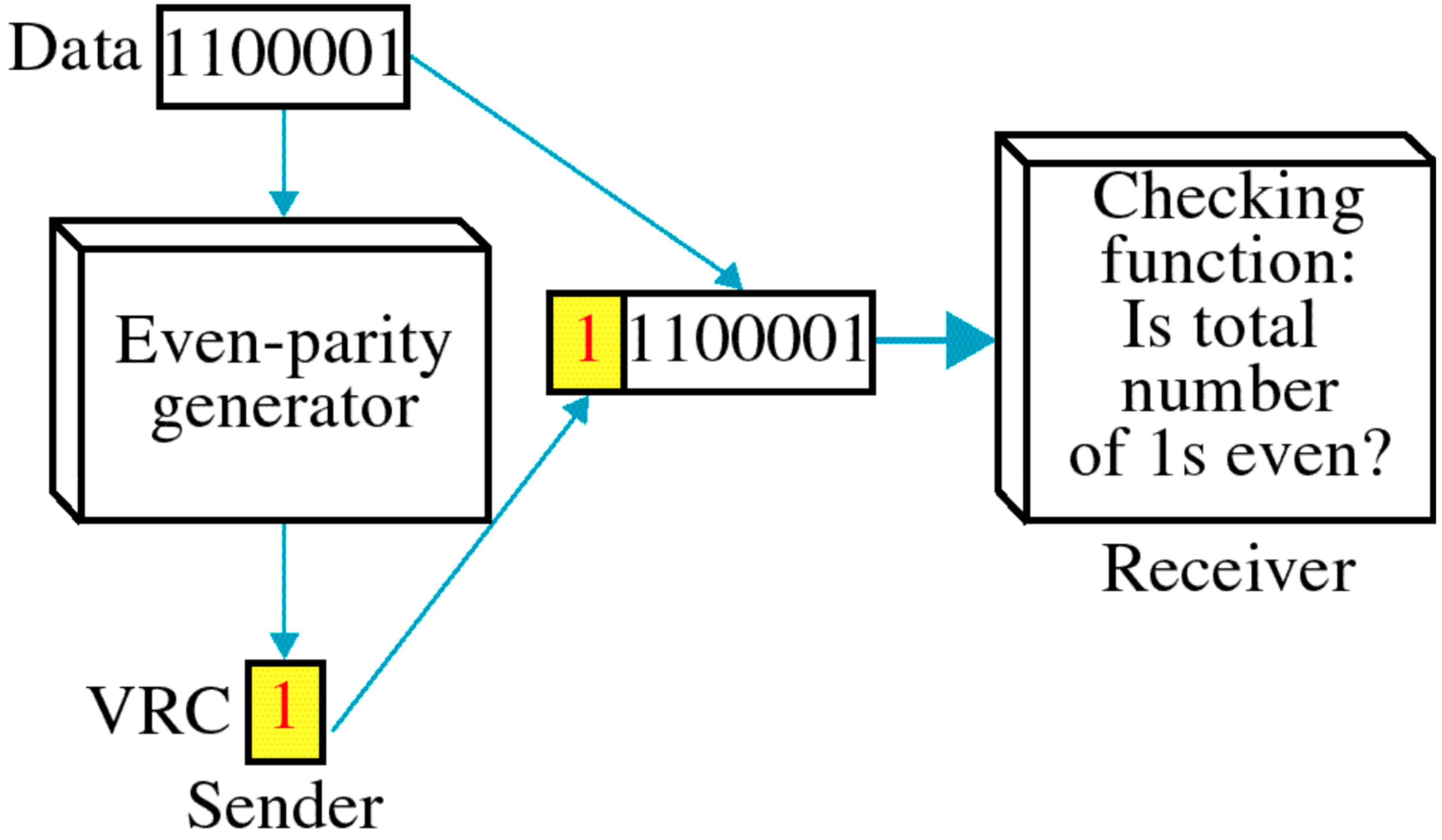
How is Syndrome calculated?

Parity

- Dataword = 1011, Codeword =?
- What happens, if the receiver receives:
- 10111
- 10011
- 10110
- 00110
- 01011
- Conclusion?

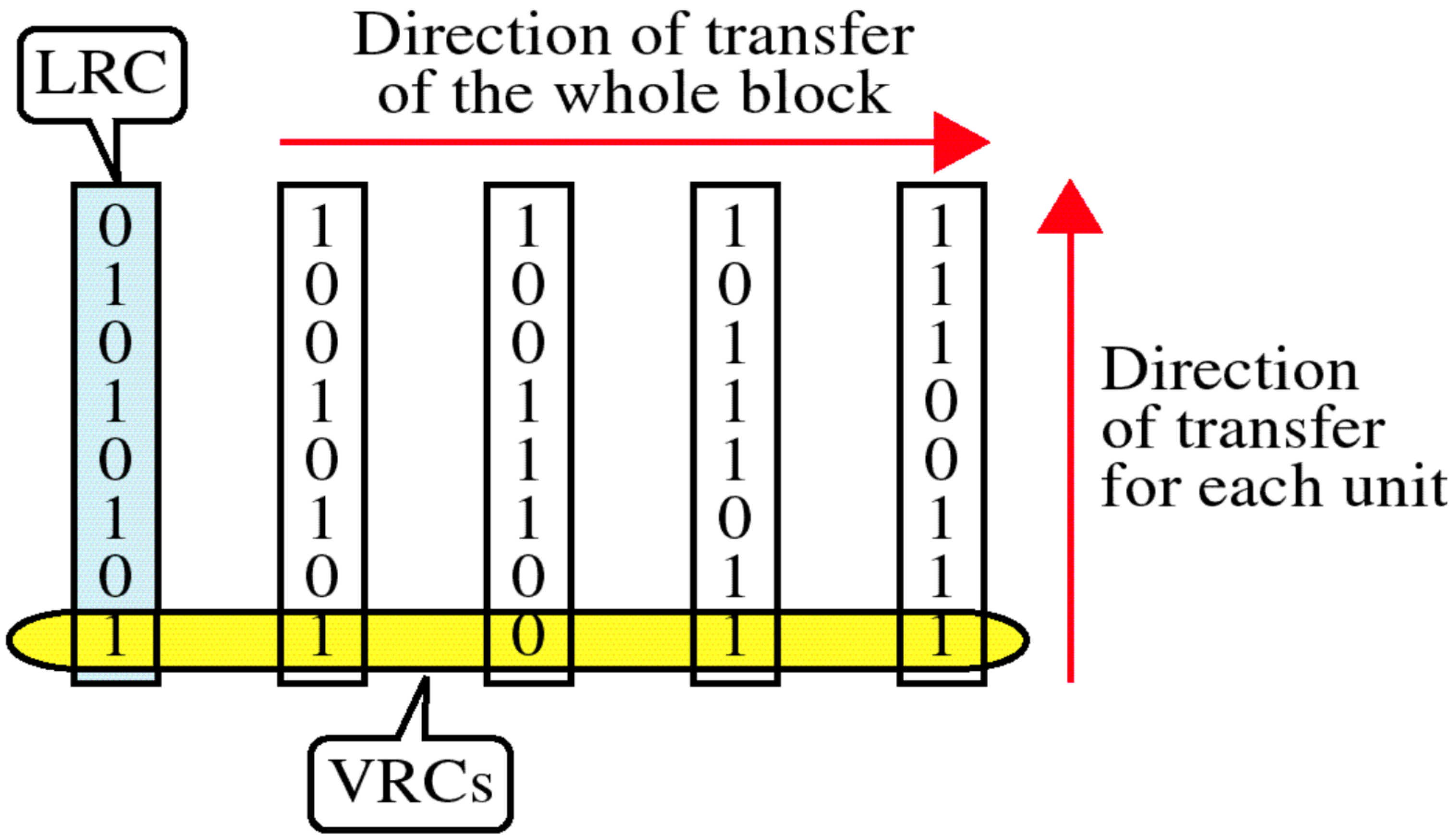
A simple parity-check code can detect an odd number of errors.

Vertical Redundancy Check (VRC)

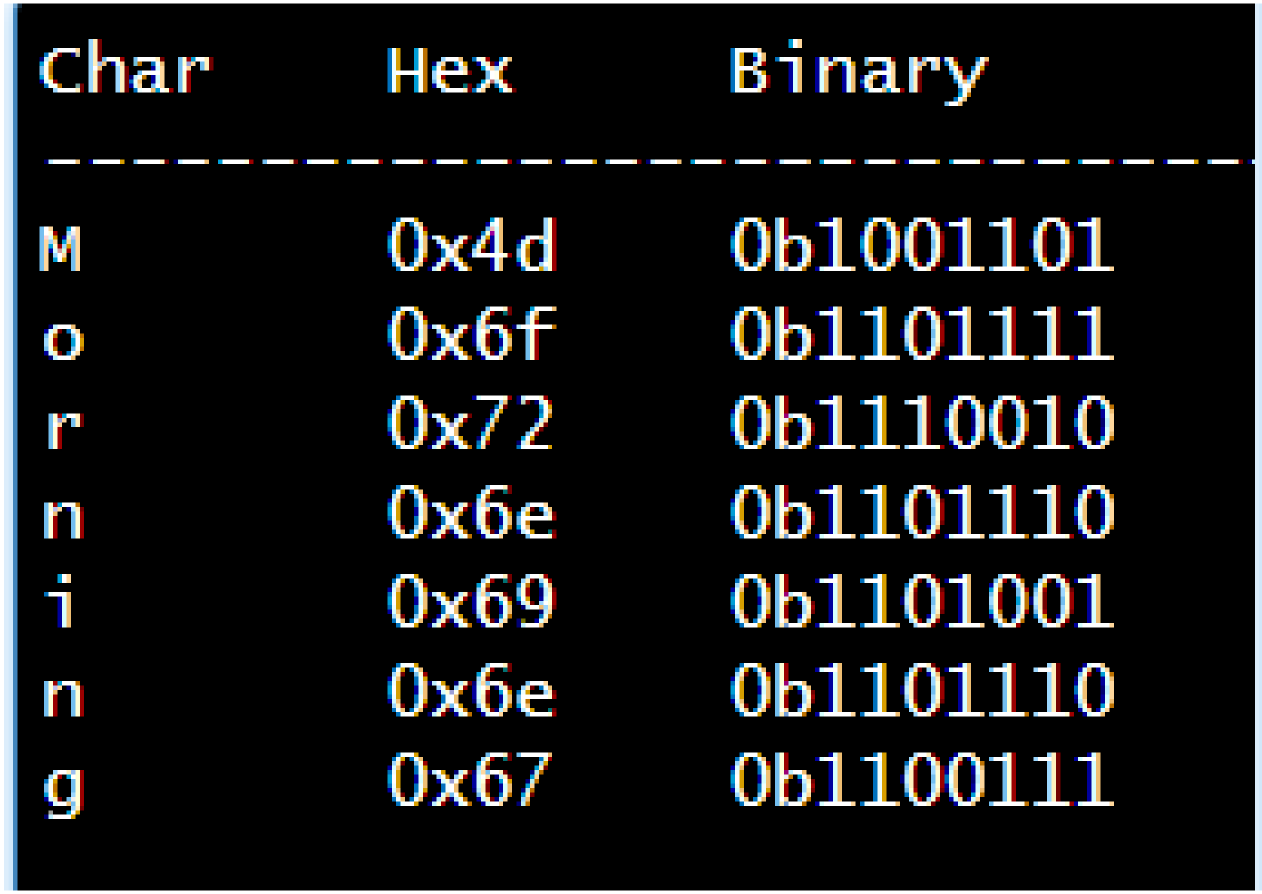


Longitudinal Redundancy Check (LRC)

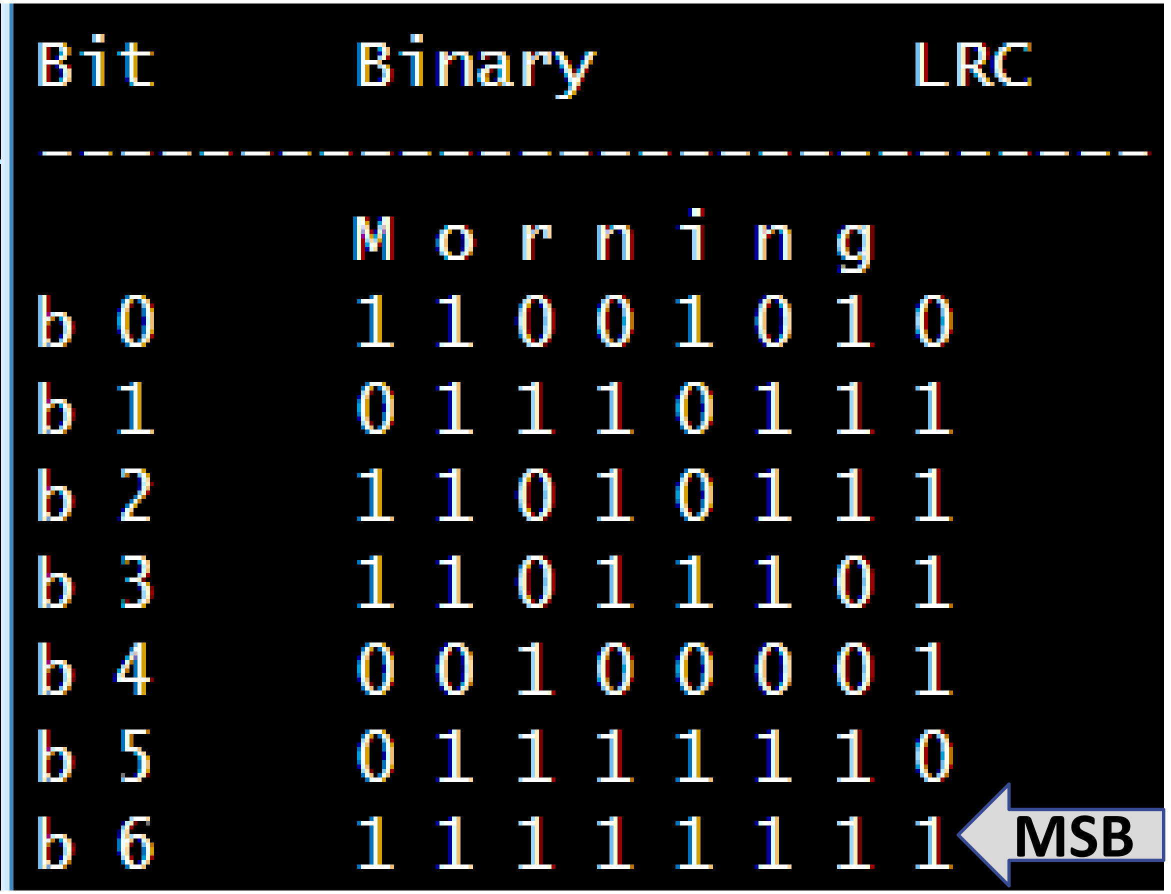
Longitudinal Redundancy Check (LRC) and VRC



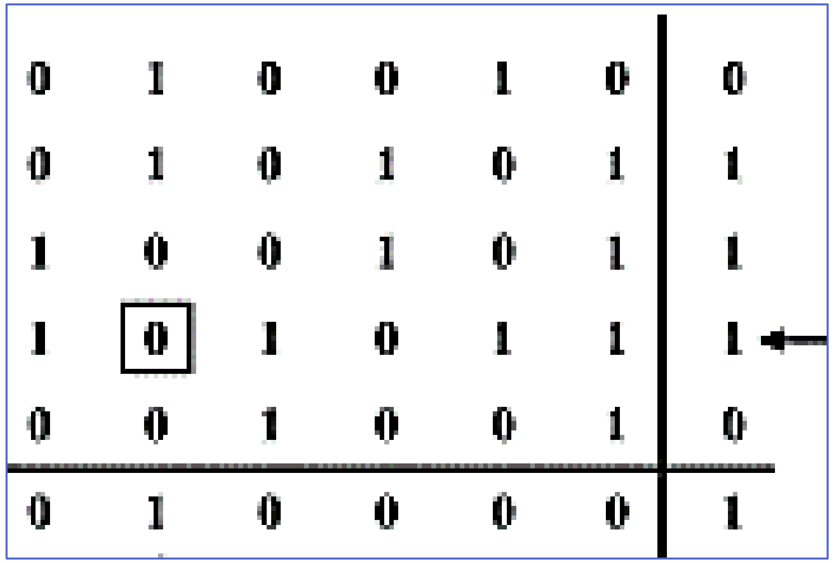
Two-Dimensional Parity

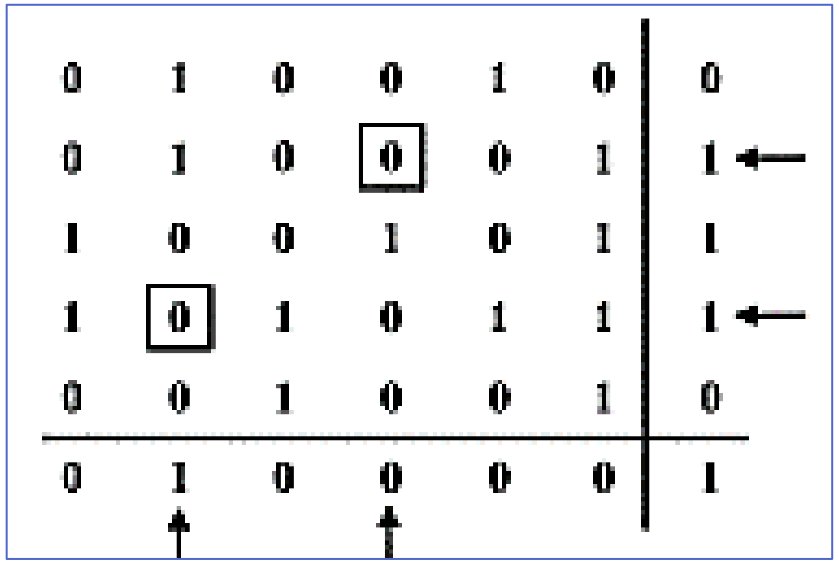


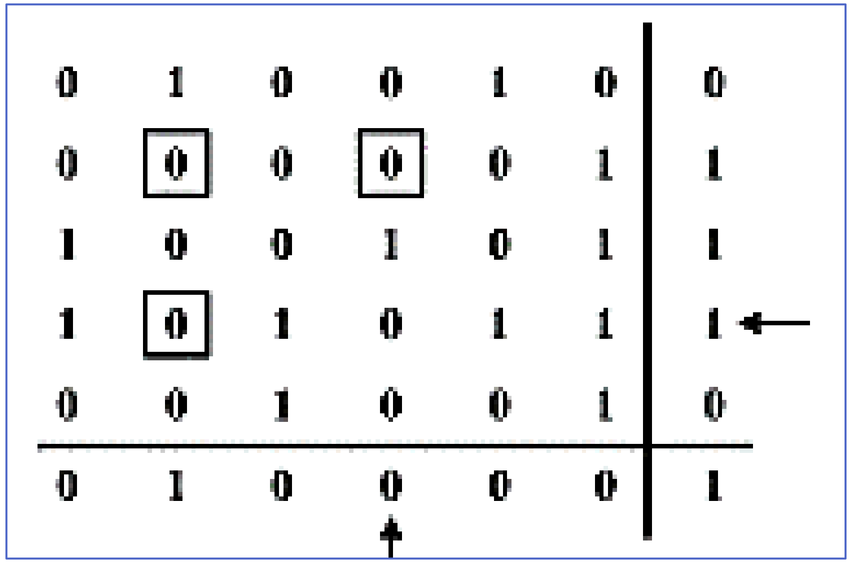
```
Char     Hex      Binary
-----------------------------------
M        0x4d     0b1001101
o        0x6f     0b1101111
r        0x72     0b1110010
n        0x6e     0b1101110
i        0x69     0b1101001
n        0x6e     0b1101110
g        0x67     0b1100111
```

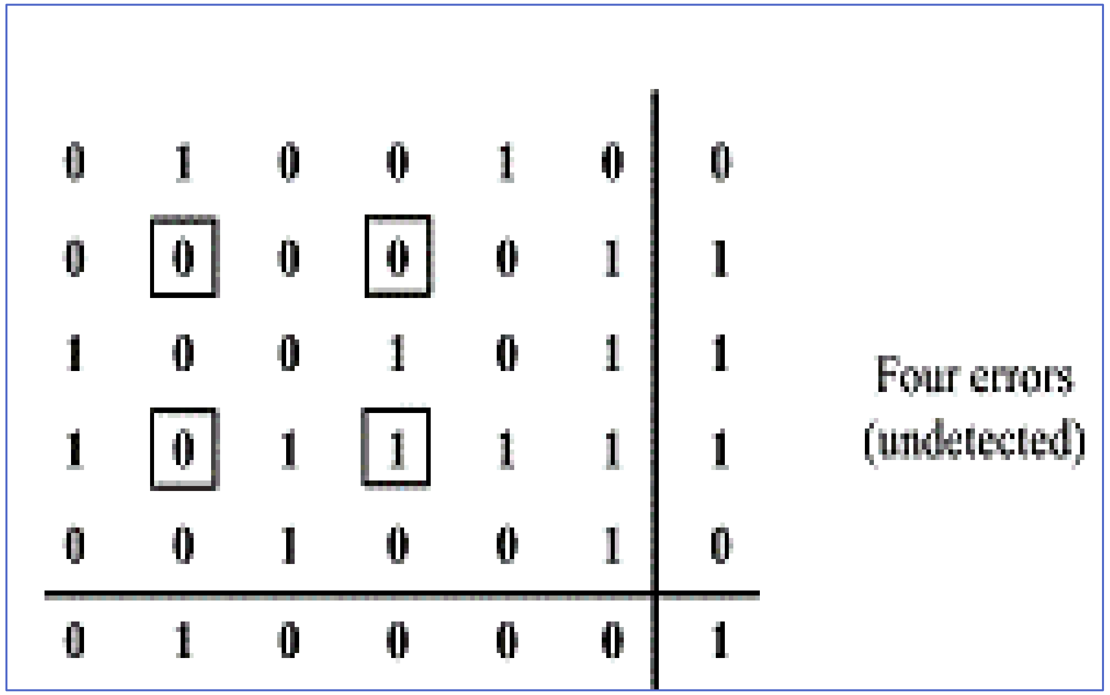


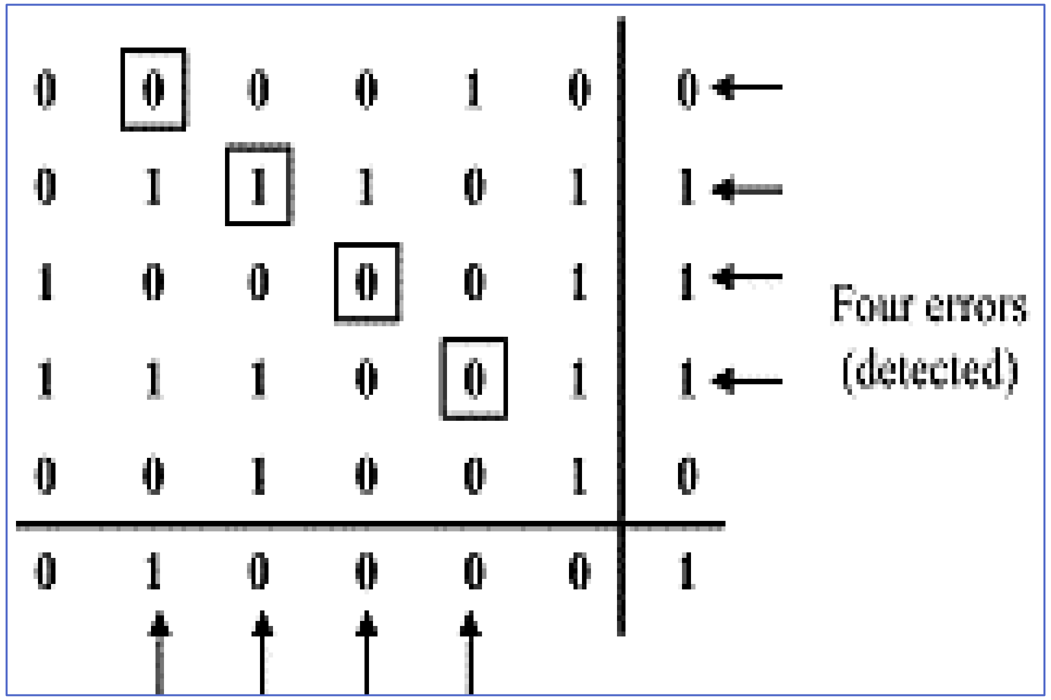
```
Bit     Binary          LRC
-----------------------------
        M o r n i n g
b 0     1 1 0 0 1 0 1   0
b 1     0 1 1 1 0 1 1   1
b 2     1 1 0 1 0 1 1   1
b 3     1 1 0 1 1 1 0   1
b 4     0 0 1 0 0 0 0   1
b 5     0 1 1 1 1 1 1   0
b 6     1 1 1 1 1 1 1   1
```





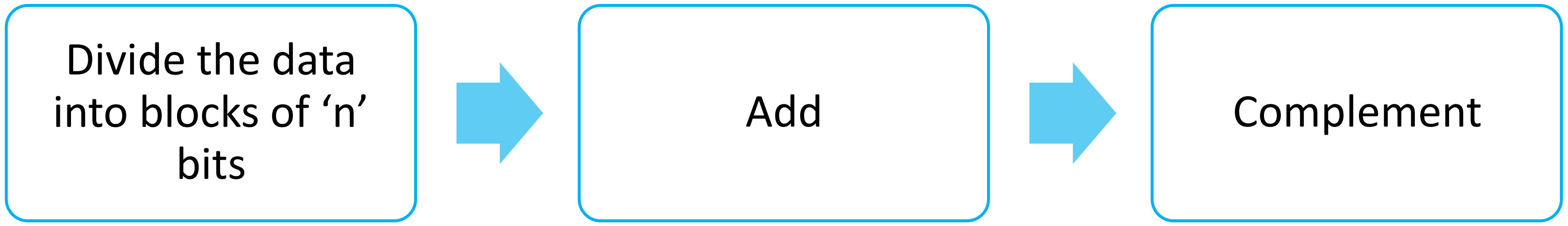



Four errors
(undetected)

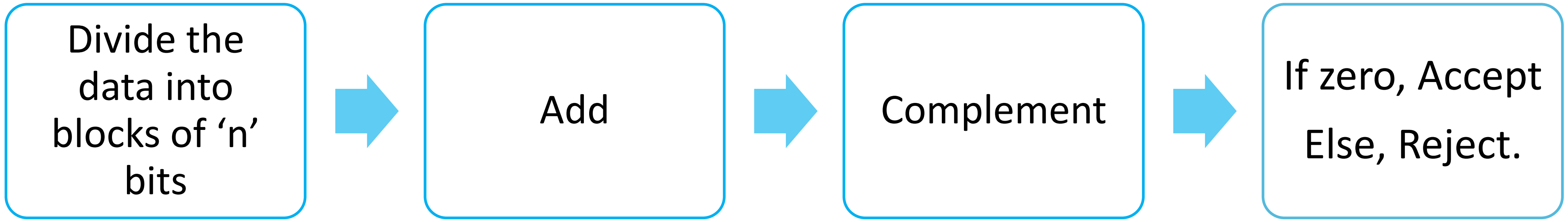

Four errors
(detected)

Checksum

- Error detection method used by most TCP/IP protocols.
- Protects against corruption of data during transmission.
- Calculated at sender site, verified at receiver site.
- **Sender site:**



- **Receiver site:**



Question (Tutorial 6)

- Calculate the checksum of “CHECKSUM”.

```
CH                  0100 0011 0100 1000
EC                  0100 0101 0100 0011
KS                  0100 1011 0101 0011
UM                  0101 0101 0100 1101
---------------------------------------------------
                  1 0010 1001 0010 1011
                                      1
----------------------------------------------------
SUM:                0010 1001 0010 1100
CHECKSUM:           1101 0110 1101 0011
```

Tutorial 6

1. A wants to send a message "**HI**" to B. The divisor used is 11001. What is the codeword that is sent to B? The Hexadecimal equivalent of H is 0x48 (110000) and I is 0x49 (110001). (Use polynomials)
2. "1100 0011 0001 1101" is received by B. B checks for error using hardware implementation. What is the conclusion? (Divisor=11001)
3. What does "**cyclic**" signify in CRC?
4. CRC versus Checksum versus Parity check?
5. VRC versus LRC?
6. Calculate (and verify) the checksum of "CHECKSUM".

Checksum

- What if the sender decides not to compute the checksum?
- What if the sum is all 1s?
- What if the sum is all 0s?
- T+(-T) = -0, -(-0) =0,

where T is the sum of all ‘k’ blocks (each of ‘n’ bits), calculated by the sender.

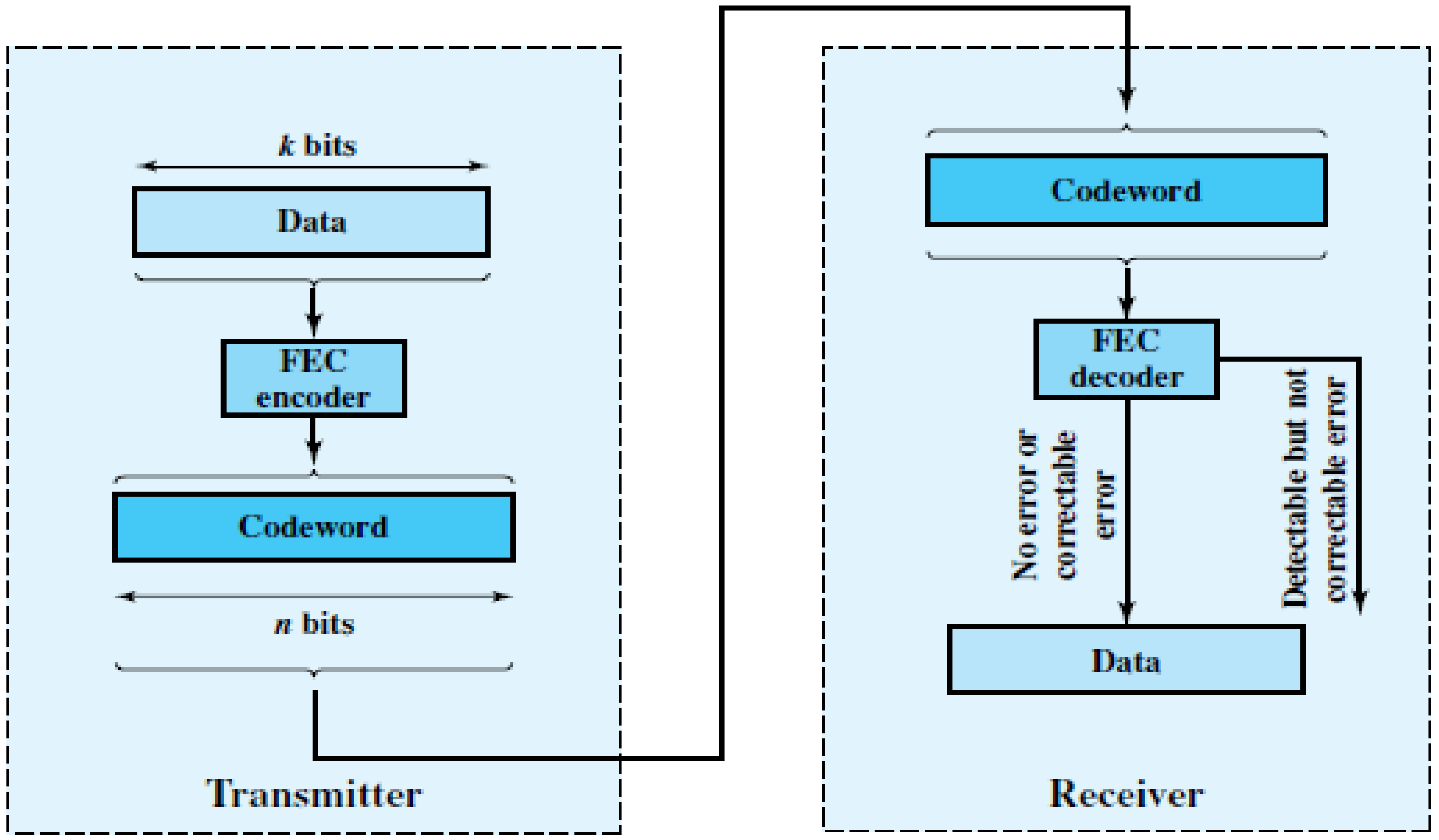
Error Correction

- Techniques?
- Retransmission is inadequate for wireless applications:
  - Quite high BER => many retransmissions
  - T_p is quite high as compared to transmission time. Only one erroneous frame or subsequent frames?

Error Correction Process

Possible Outcomes:

1. No bit errors
2. Detect but not correct
3. Detect and Correct
4. Not detected.



Hamming Code

- Given n=5, and k=2, how many codewords are possible?
- How many of these are valid codewords?

Hamming Distance

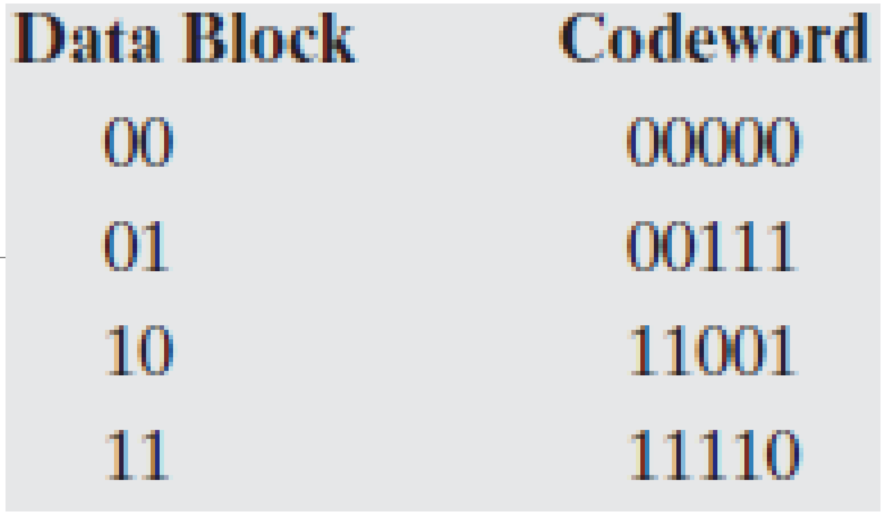
- The Hamming distance between two words is the number of differences between corresponding bits.

  v1 = 011011, **v**2 = 110001. **d(v1,v2)** =3

- Minimum Hamming Distance, $\mathbf{d}_{\mathbf{min}}$
  - smallest Hamming distance between all possible pairs in a set of words.

- Given: binary numbers from 000 to 111. What is the minimum hamming distance?

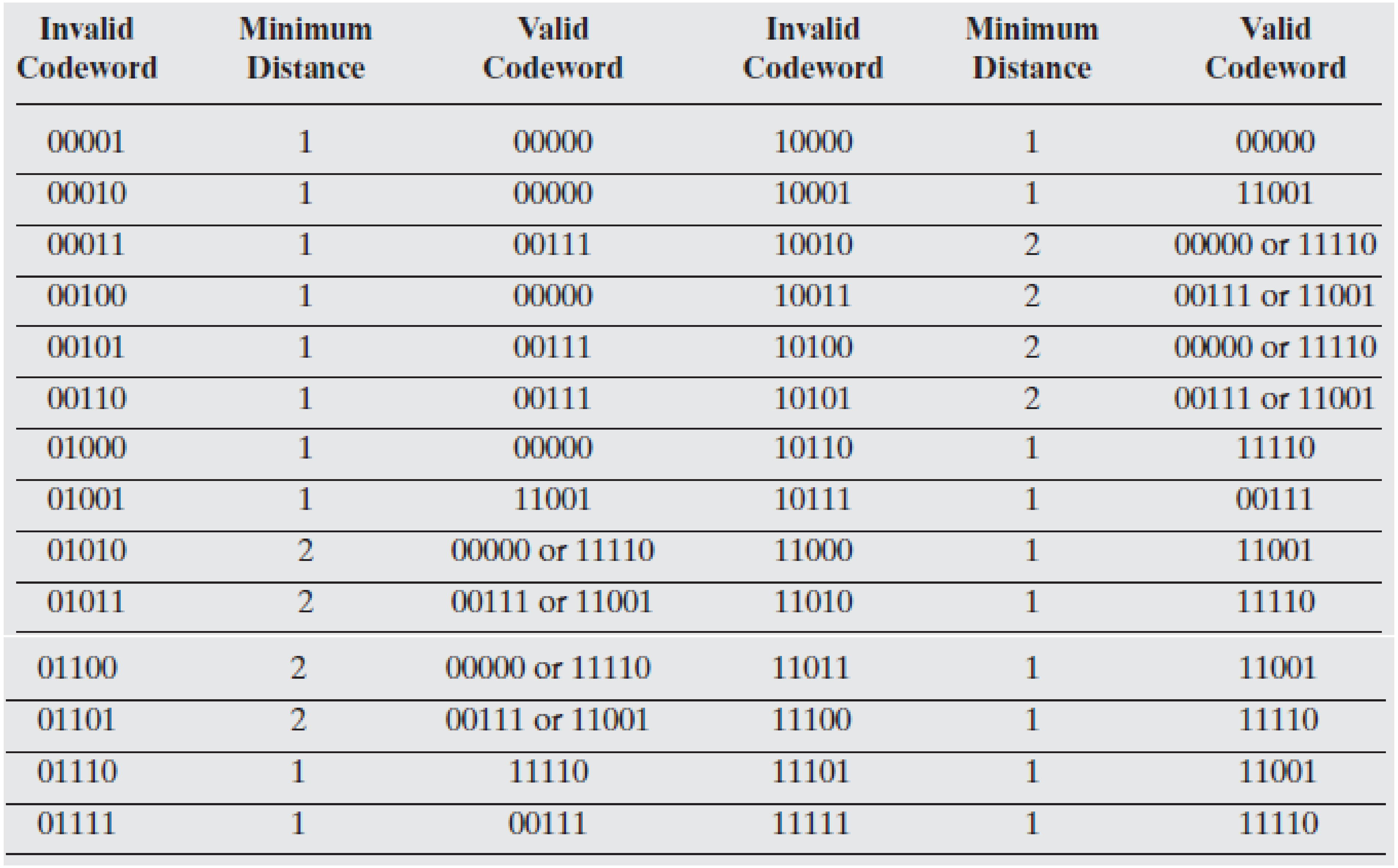
Hamming Code



Data Block	Codeword
00	00000
01	00111
10	11001
11	11110

- Find d_{min}?
- What if 00001 is received?
- What if 00110 is received?
- What if 01010 is received?
- How many errors it can detect, guaranteed?
- How many errors it can correct , guaranteed?

If an invalid codeword is received, then the valid codeword that is closest to it (minimum distance) is selected. This will only work if there is a unique valid codeword at a minimum distance from each invalid codeword.



Invalid Codeword	Minimum Distance	Valid Codeword	Invalid Codeword	Minimum Distance	Valid Codeword
00001	1	00000	10000	1	00000
00010	1	00000	10001	1	11001
00011	1	00111	10010	2	00000 or 11110
00100	1	00000	10011	2	00111 or 11001
00101	1	00111	10100	2	00000 or 11110
00110	1	00111	10101	2	00111 or 11001
01000	1	00000	10110	1	11110
01001	1	11001	10111	1	00111
01010	2	00000 or 11110	11000	1	11001
01011	2	00111 or 11001	11010	1	11110
01100	2	00000 or 11110	11011	1	11001
01101	2	00111 or 11001	11100	1	11110
01110	1	11110	11101	1	11001
01111	1	00111	11111	1	11110

If D_{min} (valid codewords)= 3, result?

Hamming Code

	't' bits error detected if:	**'t' bits error corrected if:**
Error Detection Only	$t = d_{min} - 1$	---
Error Detection and Correction ($d_{min} \geq 2t + 1$)	$t = d_{min} - 1$	$t = \text{floor}((d_{min} - 1)/2)$
Error Detection and Correction ($d_{min} \geq 2t$)	't' bits detected	≤ (t-1) bits corrected

Redundancy of the code = (n-k)/k

Code Rate = k/n $\Longrightarrow$ The code rate is a measure of **how much additional bandwidth** is required to carry data at the **same data rate** as without the code.

If the data rate input to the encoder is 1 Mbps, then to keep up, what should be the data rate of the output from the encoder?

Block Code Design Considerations

1. The largest possible value of $d_{min,}$ for given values of n and k.

2. The code should be relatively easy to encode and decode, requiring minimal memory and processing time.

3. The number of extra bits (n-k) should be small, to reduce bandwidth.

4. The number of extra bits (n-k) should be large, to reduce error rate.

How to design a code: C(n,k), d_{min} ?

- If k=2, what should be the size of n, or how many redundant bits should be added?

$$\boxed{2^{n-k} \geq (n-k) + k + 1}$$

Design a Hamming Code (5,2)(For Even Parity)

Data	**D5**	**P4 (2^2)**	**D3**	**P2 (2^1)**	**P1 (2^0)**
00:	0	0	0	0	0
01:	0	0	1	1	1
10:	1	1	0	0	1
11:	1	1	1	1	0

P1 = P1 ⊕ D3 ⊕ D5 ⊕ D7

P2 = P2 ⊕ D3 ⊕ D6 ⊕ D7

P4 = P4 ⊕ D5 ⊕ D6 ⊕ D7

	2^2	**2^1**	**2^0**
0	0	0	0
1	0	0	**1**
2	0	**1**	0
3	0	**1**	**1**
4	**1**	0	0
5	**1**	0	**1**
6	**1**	**1**	0
7	**1**	**1**	**1**

Question

Alice wants to send a pin (in binary) to Bob. The pin is 1010. Bob receives the message as 1010000. (Use Hamming Code)

(a) Find out if Bob has received the correct codeword.

(b) If not, where is the error?

(Hint: Calculate P_1, P_2, P_4. The position of error is $(P_4\,P_2\,P_1)_2$.

(a) What is the correct codeword?

Checksum Tutorial Questions. (Ignore space between words.)

Q1) Alice sent a message "**HELLO BOB**" to Bob (of course). Alice has also included the checksum in addition to the message. Bob received the following message.

0100	1000	0100	0101
0100	1100	0100	1100
0100	1111	0100	0010
0100	1111	0100	0010
1100	1100	1100	1001

(a) Show how Bob verifies the message.

(b) How does Bob know whether there is an error in the message or not?

(c) Can Bob find out where the error is? Justify.

Checksum Tutorial Questions . (Ignore space between words.)

Q2)Bob replied to Alice by sending the message “**HEY ALICE**”. However, Bob used 8-bit blocks to calculate the checksum.

(a) Show the process of calculation of checksum in this case.

(b) What is the checksum?

Summary

- Types of Error
- Detection versus Correction
- Error Probabilities
- Why do we need error detecting techniques?
- Error Detection Process: Dataword to Codeword
- CRC: Binary, Polynomials, Logic Gates (Hardware Implementation)
- Checksum (One’s Complement Arithmetic)
- Parity: Even, Odd, 1-D, 2-D (VRC, LRC)

VRC

1	1	1	0	0	1	1	**1**
1	1	0	1	1	1	0	**1**
0	0	1	1	1	0	0	**1**
1	0	1	0	1	0	1	**0**
1	**0**	**1**	**0**	**1**	**0**	**0**	**0**

Summary

- Error Correction Techniques
- Why FEC over retransmission?
- Error Correction Process
- Hamming Distance, Minimum Hamming Distance
- Hamming Code: How to generate codeword.
- Number of bits of errors detected and Corrected wrt d_{min}.

Tutorial Questions

- Q1: Which of the following g(x) values guarantees that a single-bit error is caught?

  a. $x + 1$

  b. x^3

  c. 1

- Q2: For generator polynomial: $x^{14} + x^3 + 1$, what is the probability that the error will be detected?
- Q3: The data word to be sent is 1010. Receiver gets the codeword: 0000010. Can the error be detected? Can it be corrected? Justify the result.
- Q4: Data word is 1000 000. Find codeword, using Hamming code.
- Q5: If E= 00010100 and the transmitted word is 10111011, what is the word received by the receiver?

Books

- William Stallings, Data and Computer Communications (9e), Pearson Education Inc., Noida, 2017, **Chapter 7 (7.2, 7.3, 7.4)**
- Forouzan B., *Introduction to data communication & networking (4e),* Tata McGraw Hill, New Delhi-2014. (Error Detection and Correction Problems)
- Additional Materials from several sources (journals, articles, book chapters, etc.)