CS010 404

Signals and Communication Systems

Module 5 – Error Correction , Coding

Dept. of Computer Science and Engineering

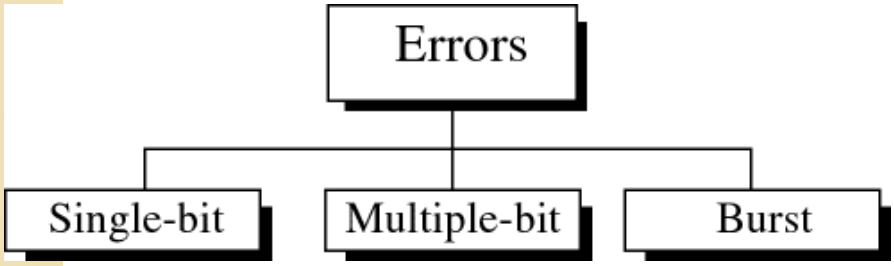
Module 5: Syllabus

Error Correction and Detection:-
Line Coding Schemes
Block Coding
Convolution Coding
Hamming Codes
Transmission Codes:-
Different Character Codes- ASCII, EBCDIC, Baudot Code, Bar Coding, Parity Coding

2

Errors in Data Transmission

Errors occur when a bit is altered between transmission and reception



3

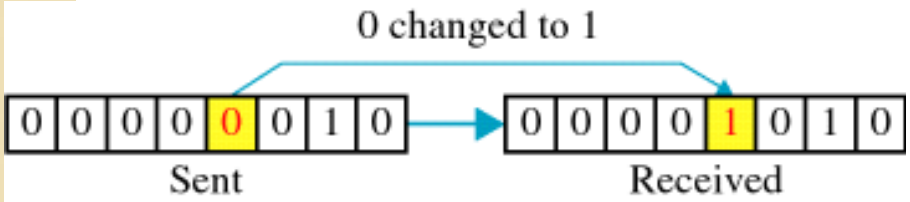
Unidirectional Errors

- Errors in block of data which only cause 0→1 or 1→0, but not both
  - Any number of bits in error in one direction
- Example
  - For the data 111000
  - Unidirectional errors could cause
    - 001000, 000000, 101000 (only 1→0 errors)
  - Non-unidirectional errors
    - 101001, 011001, 011011 (both 1→0 and 0→1)

4

Single-bit error

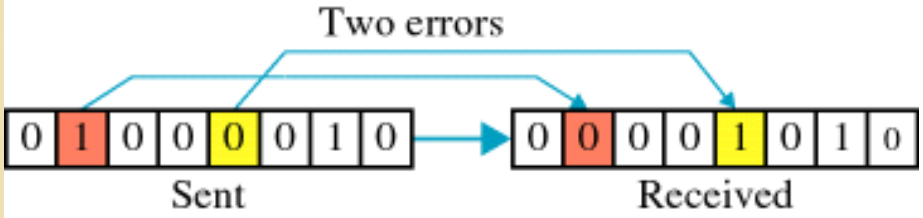
Single bit errors
- only one bit altered
- caused by white noise



5

Multiple-bit error

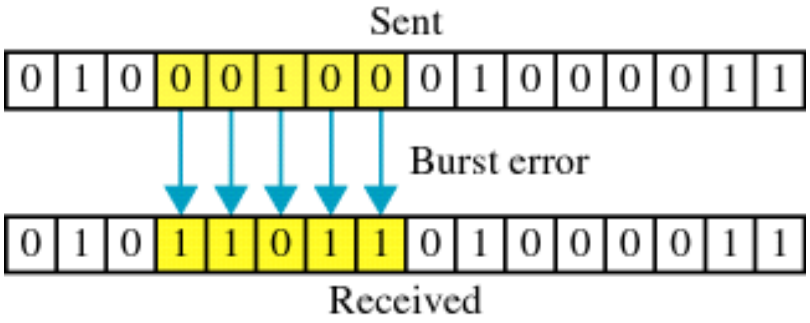
- Multiple bit errors
  - Multiple bits are altered
  - Not necessarily contiguous
  - Caused by white noise



6

Burst error

- Burst errors
  - Contiguous sequence of *B* bits in which first last and any number of intermediate bits in error
  - Caused by impulse noise or by fading in wireless
  - Effect is greater at higher data rates



7

Burst error - Problem

What is the maximum effect of a 2 millisecond burst of noise on data transmitted at the following rates ?

a. 2000 bps
b. 14 Kbps
c. 200 Kbps
d. 300 Mbps

8

Error Detection

- Transmission will have errors
- Detect using error-detecting code
- Code is added by transmitter
- Code is recalculated and checked by receiver
- Still chance of undetected error
- Simple error detection - Parity
  - Parity bit set so character has even (even parity) or odd (odd parity) number of ones
  - Even number of bit errors goes undetected

9

Error Control Techniques

- Forward Error Correction (FEC)
  - Coding designed so that errors can be corrected at the receiver
  - Appropriate for delay sensitive and one-way transmission (e.g., broadcast TV) of data
  - Two main types, namely block codes and convolutional codes.

10

Block Codes - I

- Data is grouped into blocks of length *k* bits (dataword)
- Each dataword is coded into blocks of length *n* bits (codeword), where in general *n*>*k*
- For each block add (n-k) redundant bits to produce the codeword
- This is known as an (*n*,*k*) block code
- The redundancy introduced by the code is quantified by the code rate,
  - Code rate = *k*/*n*
  - i.e., the higher the redundancy, the lower the code rate

11

Block Codes - II

- Can detect errors that cause codeword to become non-codeword
- Cannot detect errors that cause codeword to become another codeword

12

Block Code - Example

- Dataword length k = 4
- Codeword length n = 7
- This is a (7,4) block code with code rate=4/7
- For example, d = (1101), c = (1101001)

13

Simple Block Code - Parity Codes

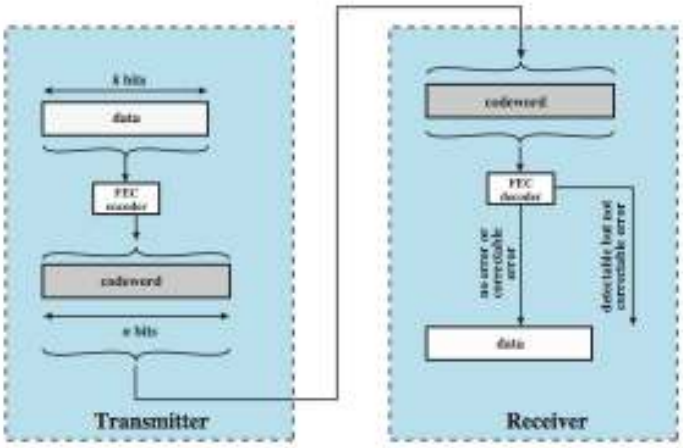
- Even parity

(i) d=(10110) so,
c=(101101)

(ii) d=(11011) so,
c=(110110)

14

Error Detection Process



15

Checksum

- Used to detect errors in data transmission on communication networks
- Basic idea is to add up the block of data being transmitted and transmit this sum (check sum) as well
- Receiver adds up the data it received and compares it with the checksum it received
- If the two do not match - an error is indicated
- All checksum schemes allow error detection but not error location - entire block of data must be retransmitted if an error is detected

16

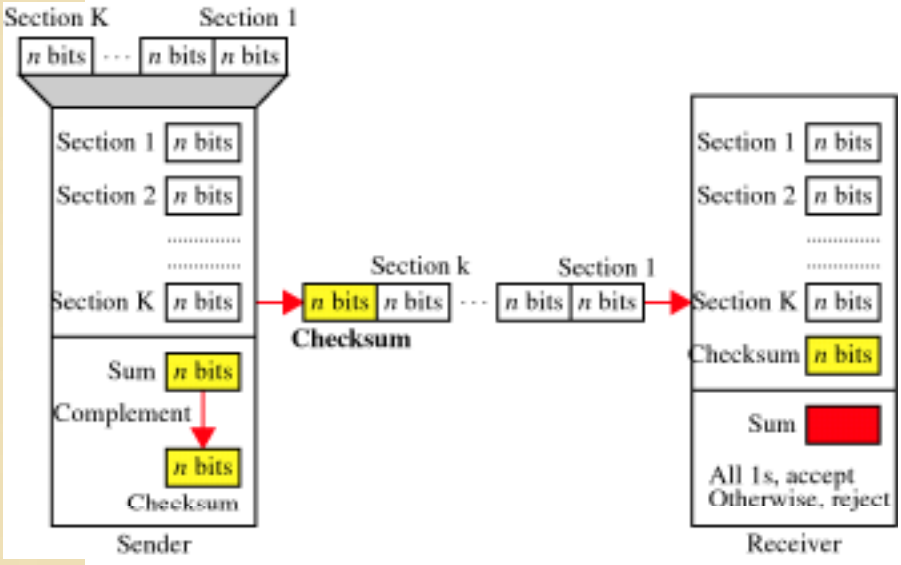
Versions of Checksum

(a) Single-precision	(b) Double-precision	(c) Residue
0000	0000	0000
0101	0101	0101
1111	1111	1111
0010	0010	0010
0110	**00010110**	**0111**

- **In general - single-precision checksum catches fewer errors than double-precision, since it only keeps the rightmost d bits of the sum**
- **Residue checksum takes into account the carry out bit as an end-around carry – somewhat more reliable**

17

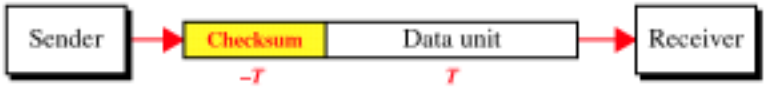
Checksum



18

Data Unit and Checksum

The receiver adds the data unit and the checksum field. If the result is all 1s, the data unit is accepted; otherwise it is discarded.



Problem

A sender needs to send the four data items 0x3456, 0xABCC, 0x02BC, and 0xEEEE. Answer the following

a. Find the checksum at the sender site
b. Find the checksum at the receiver site of there is no error
c. Find the checksum at the receiver site if the second data item is changed to 0xABCE during transmission
d. Find the checksum at the receiver site if the second data item is changed to 0xABCE and the third data item is changed to 0x02BA during transmission.

19

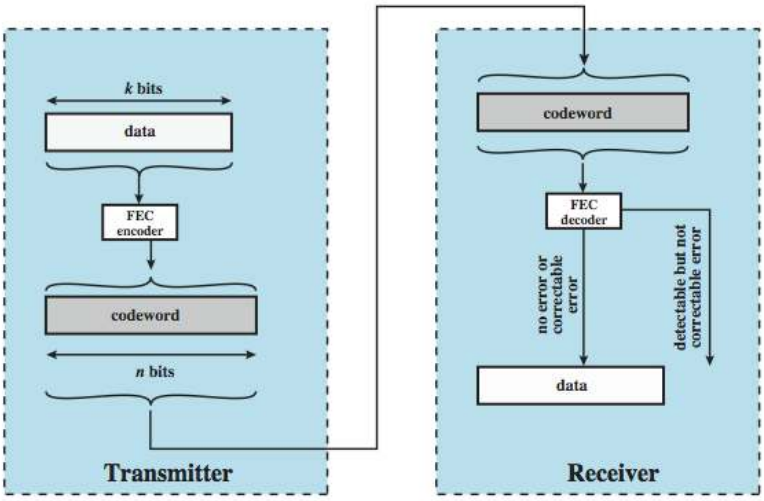
Cases in which checksum fails

At least three types of error cannot be detected by the current checksum calculation:

1. If two data items are swapped during transmission, the sum and the checksum values will not change.
2. If the value of one data item is increased (intentionally or maliciously) and the value of another one is decreased (intentionally or maliciously) the same amount, the sum and the checksum cannot detect these changes.
3. If one or more data items is changed in such a way that the change is a multiple of $2^{16} - 1$, the sum or the checksum cannot detect the changes.

20

Error Correction Process



21

Error Correction schemes

ARQ - Automatic-Repeat Request scheme

- Upon detection of error, the receiver requests a repeat transmission of the corrupted codeword

FEC - Forward Error-Correction

- Channel encoder accepts information in successive k-bit blocks and for each block, it adds (n-k) redundant bits to produce an encoded block of n-bits called a codeword
- Channel decoder uses the redundancy to decide which message bits were actually transmitted.

22

Automatic Repeat Request (ARQ)

- Collective name for error control mechanism based on repeat transmission, including:

1. stop and wait
2. go back N
3. selective reject (selective retransmission)

23

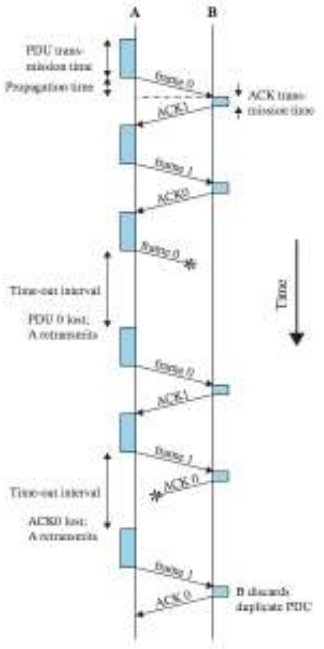
Stop and Wait

- Source transmits single frame
- Wait for ACK
- If received frame damaged, discard it
  - Transmitter has timeout
  - If no ACK within timeout, retransmit
- If ACK damaged, transmitter will not recognize it
  - transmitter will retransmit
  - receive gets two copies of frame
  - use alternate numbering and ACK0 / ACK1

24

Stop and Wait

- See example with both types of errors
- Pros and cons
  - simple
  - inefficient



25

Go Back N

- Based on sliding window
- If no error, ACK as usual
- Use window to control number of outstanding frames
- If error, reply with rejection
  - discard that frame and all future frames until error frame received correctly
  - transmitter must go back and retransmit that frame and all subsequent frames

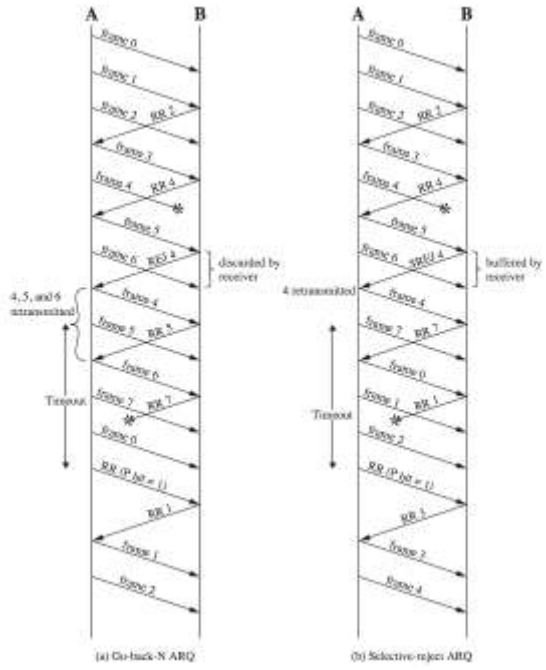
26

Selective Reject

- Also called selective retransmission
- Only rejected frames are retransmitted
- Subsequent frames are accepted by the receiver and buffered
- Minimizes retransmission
- Receiver must maintain large enough buffer
- More complex logic in transmitter
- Hence less widely used
- Useful for satellite links with long propagation delays

27

Go Back N vs Selective Reject



28

Error Correction

- ARQ correction of detected errors require data block to be retransmitted
- Not appropriate for wireless applications
  - Bit error rate is high causing lots of retransmissions
  - When propagation delay long (satellite) compared with frame transmission time, resulting in retransmission of frame in error plus many subsequent frames
- Instead need to correct errors on basis of bits received
- FEC provides this

29

How Error Correction Works

- Adds redundancy to transmitted message
- Can deduce original despite some errors
- eg. block error correction code
  - map *k* bit input onto an *n* bit codeword
  - each distinctly different
  - if get error assume codeword sent was closest to that received
- Reduces effective data rate
- For math, see Stallings chapter 6

30

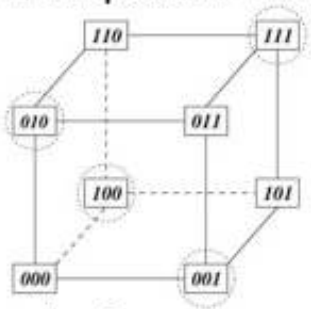
Information Coding

- A data word with k bits is encoded into a codeword with n bits - n>k
- Not all combinations are valid codewords
- To extract original data - n bits must be decoded
- If the n bits do not constitute a valid codeword an error is detected
- For certain encoding schemes - some types of errors can also be corrected
- Key parameters: number of erroneous bits that can be detected as erroneous and number of erroneous bits that can be corrected
- Overhead:
- * additional bits required
- * time to encode and decode

31

Hamming Distance

- The Hamming distance between two codewords - the number of bit positions in which the two words differ



- Two words in this figure are connected by an edge if their Hamming distance is 1

32

Hamming Distance

- The Hamming distance between two words (of the same size) is the number of differences between the corresponding bits.
- The Hamming distance can easily be found if we apply the XOR operation on the two words and count the number of 1s in the result.
- The minimum Hamming distance is the smallest Hamming distance between all possible pairs in a set of words.

33

Detection Vs Correction capability

- **To detect up to n bit errors, the code distance should be at least n+1**

Example - The code with four codewords - *{ 001,010,100,111}* –
has a distance of 2
This code can detect any single bit error

- **To correct up to n bit errors, the code distance should be at least 2n+1**

Example - The code with two codewords - {000,111} –
has a distance of 3
This code can correct any single bit error
And detect any double bit error

34

Hamming Distance – Example 1

Find the minimum Hamming distance of the coding scheme in Table

Datawords	Codewords
00	000
01	011
10	101
11	110

Solution

We first find all Hamming distances.

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

The d_{min} in this case is 2.

10.35

Hamming Distance – Example 2

Find the minimum Hamming distance of the coding scheme in Table

Dataword	Codeword
00	00000
01	01011
10	10101
11	11110

Solution

We first find all the Hamming distances.

$d(00000, 01011) = 3$ $d(00000, 10101) = 3$ $d(00000, 11110) = 4$
$d(01011, 10101) = 4$ $d(01011, 11110) = 3$ $d(10101, 11110) = 3$

The d_{min} in this case is 3.

10.36

Minimum Distance And Error correction

If d_{min} is the minimum Hamming distance between 2 codewords

- The maximum number of detectable errors is

$$s = d_{min} - 1$$

- The maximum number of correctable errors is given by,

$$t = \left\lfloor \frac{d_{min} - 1}{2} \right\rfloor$$

37

Hamming Distance – Problem

A code scheme has a Hamming distance $d_{min} = 4$. What is the error detection and correction capability of this scheme?

Solution

This code guarantees the detection of up to three errors (s = 3), but it can correct up to one error.

In other words, if this code is used for error correction, part of its capability is wasted. Error correction codes need to have an odd minimum distance (3, 5, 7, . . .).

10.38

Error detection - correction

- Error detection
  - Check if any error has occurred
  - Don't care the number of errors
  - Don't care the positions of errors
- Error correction
  - Need to know the number of errors
  - Need to know the positions of errors
  - More difficult

Error Correction by 2-D parity

2-dimensional Parity:

"Data is arranged in 2-dimensional array and parity bit is added for each row and column"

P_V

0 1 1 0 0 "Detects and *Corrects* all single bit errors:

1 0 1 0 0

1 1 1 0 1 "Detects all odd number of bit errors

0 1 1 1 1 and some even number of bit errors

0 1 0 1 0

P_H

Hamming Error Correction Code

- Named for Richard .W.Hamming – Pioneer in error correction
- Uses extra parity bits to allow the position identification of a single error

1. Mark all bit positions that are powers of 2 as parity bits. (positions 1, 2, 4, 8, 16, etc.)
2. All other bit positions are for the data to be encoded. (positions 3, 5, 6, 7, 9, 10, 11, 12, 13, 14, 15, etc.)

Bit Position:	1	2	3	4	5	6	7	8	9	10	11	12
	P1	P2	D	P4	D	D	D	P8	D	D	D	D

41

Hamming ECC

- Each parity bit calculates the parity for some of the bits in the code word. The position of the parity bit determines the sequence of bits that it checks.
- Position 1: checks bits (1,3,5,7,9,11,...) – *Alternate bits*
  Position 2: checks bits (2,3,6,7,10,11,14,15,...)
  – *Alternate 2-bits*
- Position 4: checks bits (4,5,6,7,12,13,14,15,20,21,22,23,...)
  – *Alternate 4-bits*
- Position 8: checks bits (8-15, 24-31, 40-47,...)
  – *Alternate 8-bits*

Set the parity bit to create Even parity.

42

Hamming Code – Example 1

Eg: Data word – 1 0 1 1

P1 P2 1 P4 0 1 1
1 2 3 4 5 6 7 ← Bit locations in codeword

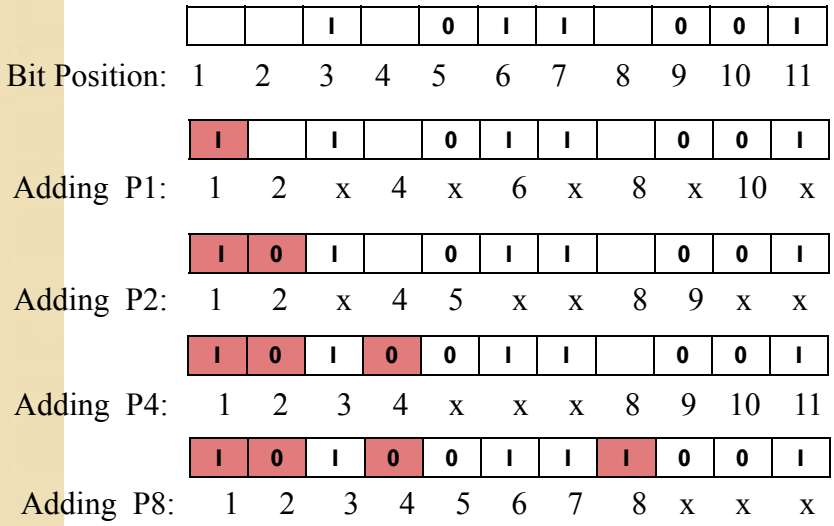
P1 – Provides even parity from a check of bit locations
1,3,5,7 (These are 101, P1 is set to 0 to achieve even parity
P2 – Checks locations 2,3,6, and 7. (111) - P2 is 1 in this case
P4 – Checks 4,5,6 and 7. (011) - P4 is 0 in this case

Hence the 7-bit Codeword is 0 1 1 0 0 1 1

43

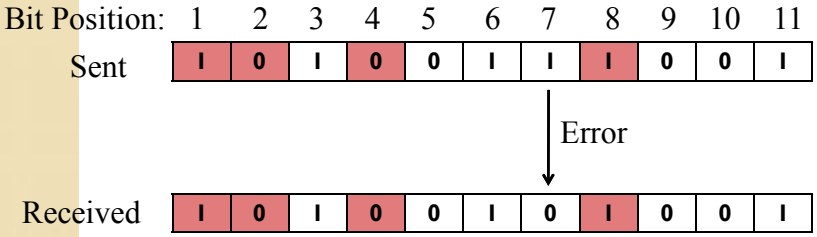
Hamming Code – Example 2

Data = 1011001



44

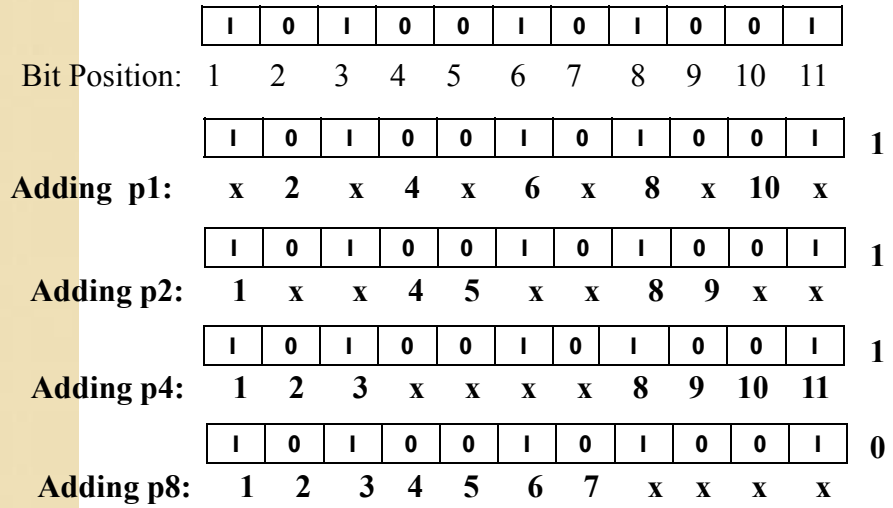
Single bit Error – Example 2 Cntd..



45

Hamming Example 2 – Syndrome Calculation

Received Code Word = 10100101001



Syndrome = 0111

46

Hamming Code – Example 3

A byte of data: 10011010

Place the data word, leaving spaces for the parity bits:

_ _ 1 _ 0 0 1 _ 1 0 1 0

1 2 3 4 5 6 7 8 9 10 11 12

Calculate the parity bits.

47

Hamming Code – Example 3 Cntd.

Position 1 checks bits 1,3,5,7,9,11: Even Parity 0
Set position 1 to a 0: **0** _ 1 _ 0 0 1 _ 1 0 1 0

Position 2 checks bits 2,3,6,7,10,11: Even Parity 1
Set position 2 to a 1: **0 1** 1 _ 0 0 1 _ 1 0 1 0

Position 4 checks bits 4,5,6,7,12: Even Parity 1
Set position 4 to a 1: **0 1** 1 **1** 0 0 1 _ 1 0 1 0

Position 8 checks bits 8,9,10,11,12: Even Parity 0
Set position 8 to a 0: **0 1** 1 **1** 0 0 1 **0** 1 0 1 0

Final code word: **01**1**1**00**1 0**1010

48

Hamming ECC - Finding and fixing a corrupted bit

Transmitted code word:
011100101010
Suppose that the word was received as 011100101110 instead.

- Error correction involves calculation of Syndrome vector
- Syndrome vector is calculated at the receiving end
- Syndrome vector indicates whether error occurred
- If so, for which codeword bit
- Null syndrome vector indicates that no error has occured

Find out the Syndrome ?

Parity bits 2 and 8 are incorrect. It is 2+8 = 10
Bit position 10 is the location of the bad bit
Bit in position 10 needs to be inverted.

49

Hamming ECC- Problems

Test if these Hamming-code words are correct. If one is incorrect, indicate the correct code word. Also, indicate what the original data was.

010101100011
111110001100
000010001010

50

Hamming Code summary

- Linear block coding
- Hamming (7,4) algorithm
  - $2^4 = 16$ datawords and $2^7 = 128$ codewords
  - 16 out of 128 codewords are used for message transfer and the rest are either used for other purposes or unused.
  - Adds three additional check bits to every four data bits of the message.
  - Hamming's (7,4) algorithm can correct any single-bit error, or detect all single-bit and two-bit errors.

51

Burst Error Correction

- Hamming Code can be used:
- Arrange N data elements (with ECC) in two dimension
- Transmit all the first bits from N elements
- Transmit all the second bits from N elements
- And so on …
- Organize them as N elements at receiver.

52

Burst Error Correction

Any burst error of length <= N is seen as a single bit error in a data element and can be corrected.

X2 X1 X0	X2 Y2 Z2	X2 X1 X0
Y2 Y1 Y0	X1 Y1 Z1	Y2 Y1 Y0
Z2 Z1 Z0	X0 Y0 Z0	Z2 Z1 Z0
Original	Transmit	Received and Organized

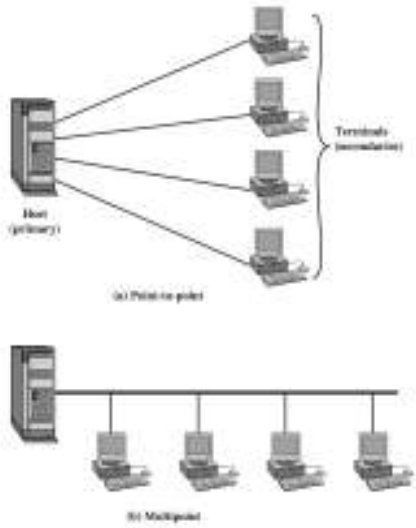
53

Line Configuration - Topology

- physical arrangement of stations on medium
  - point to point - two stations
    - such as between two routers / computers
  - multi point - multiple stations
    - traditionally mainframe computer and terminals
    - now typically a local area network (LAN)

54

Line Configuration - Topology



55

End of Module 5

56