L4: Building Direct Link Networks II

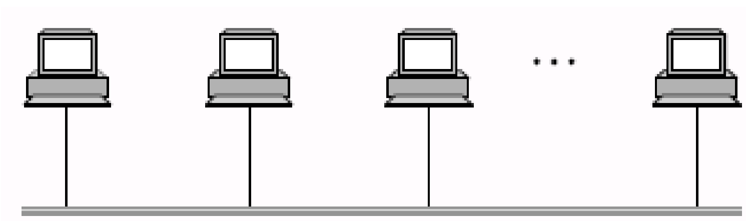
Hui Chen, Ph.D.
Dept. of Engineering & Computer Science
Virginia State University
Petersburg, VA 23806

Acknowledgements

- Some pictures used in this presentation were obtained from the Internet
- The instructor used the following references
  - Larry L. Peterson and Bruce S. Davie, Computer Networks: A Systems Approach, 5th Edition, Elsevier, 2011
  - Andrew S. Tanenbaum, Computer Networks, 5th Edition, Prentice-Hall, 2010
  - James F. Kurose and Keith W. Ross, Computer Networking: A Top-Down Approach, 5th Ed., Addison Wesley, 2009
  - Larry L. Peterson's (http://www.cs.princeton.edu/~llp/) Computer Networks class web site

Direct Link Networks

- Types of Networks
  - Point-to-point
  - Multiple access



- Encoding
  - Encoding bits onto transmission medium
- Framing
  - Delineating sequence of bits into messages
- **Error detection**
  - **Detecting errors and acting on them**
- Reliable delivery
  - Making links appear reliable despite errors
- Media access control
  - Mediating access to shared link

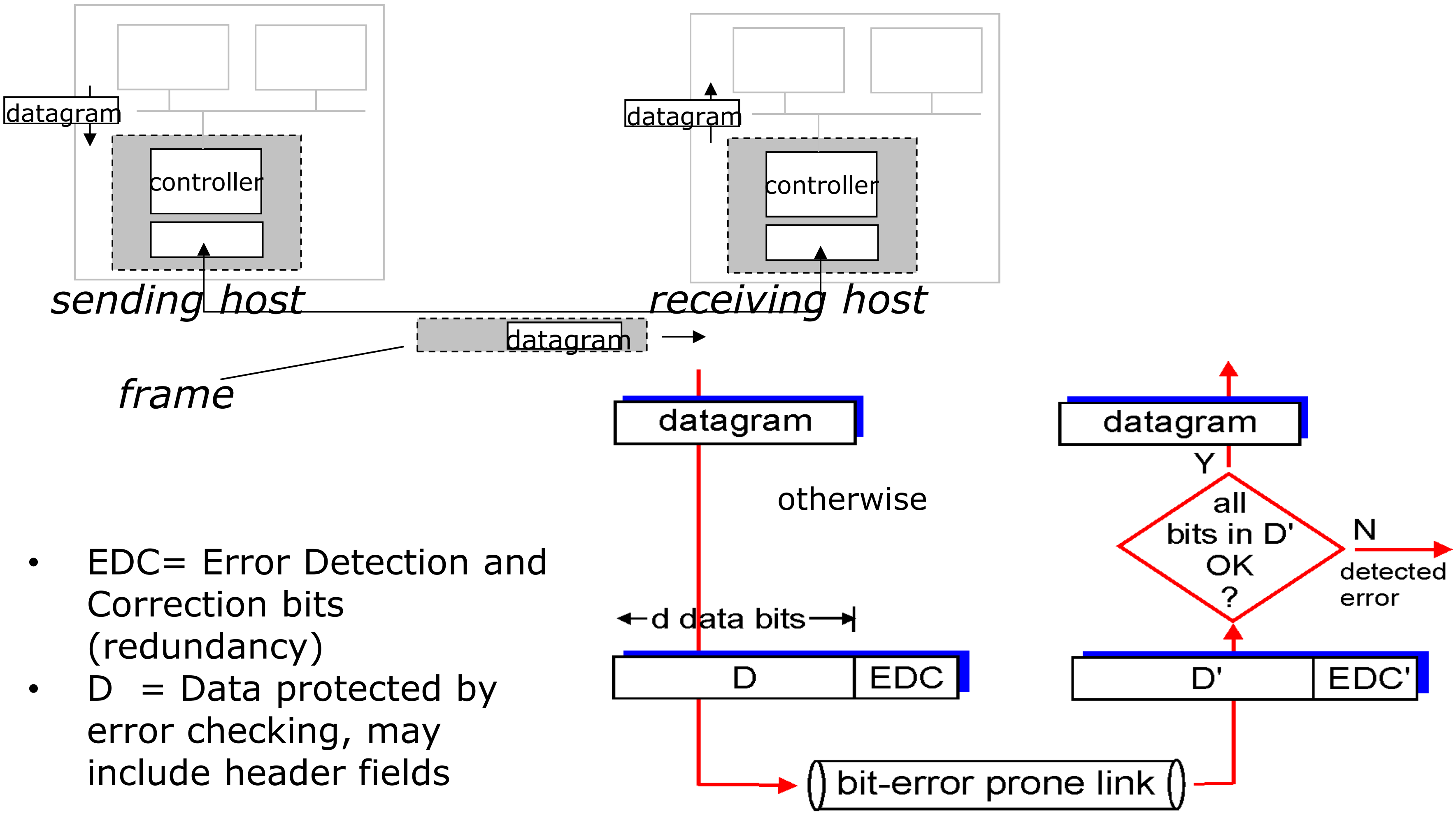
Things Can Go Wrong …

- How does a receiver know that a frame contains error?

Error Detection

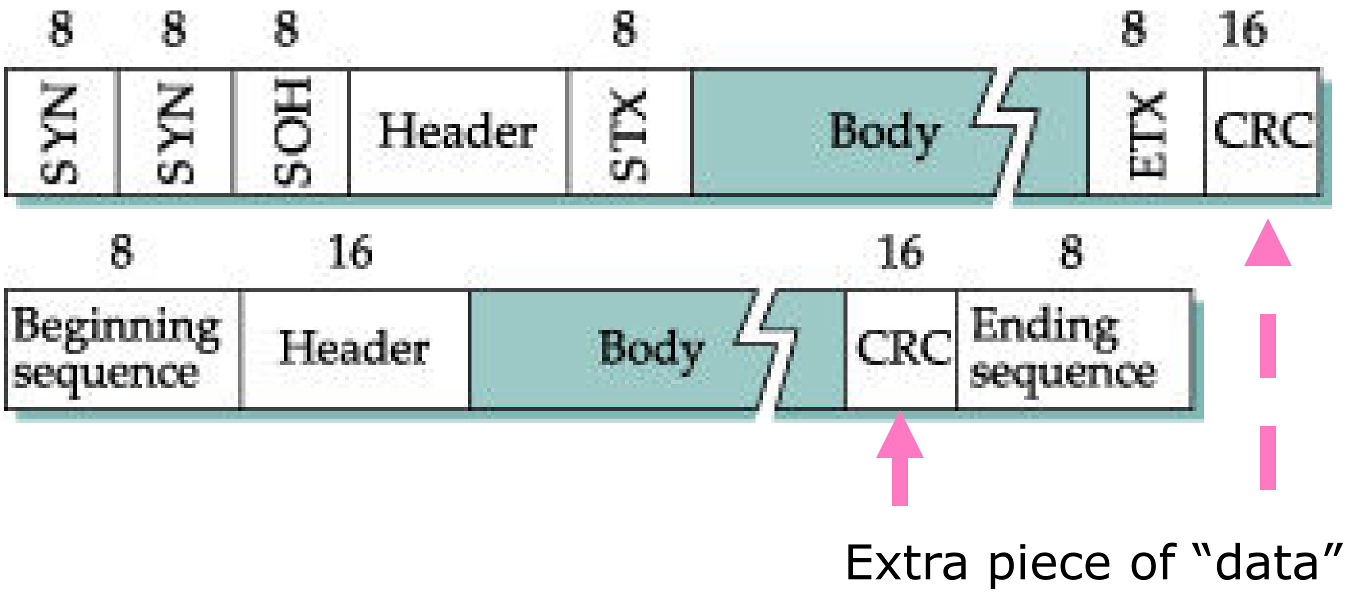
- Detect that the received contains error
- How?

Error Detection



- EDC= Error Detection and Correction bits (redundancy)
- D = Data protected by error checking, may include header fields

Additional Data for Error Detection



Error Detection Code

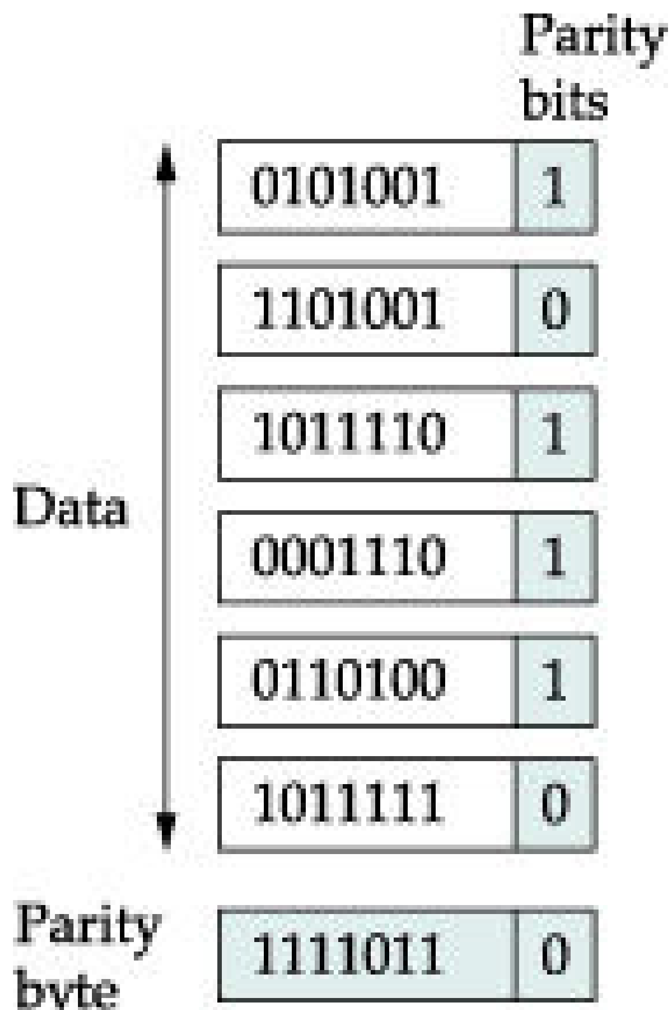
- Two Examples
  - Two-dimensional parity
  - Cyclic redundancy code

Parity Check

- Append a parity bit to each character
- Even parity
  - Set the parity bit as either 0 or 1 such that the number of 1's in the character is EVEN
- Odd parity
  - Set the parity bit as either 0 or 1 such that the number of 1's in the character is ODD

Two-Dimensional Parity

- Assume event parity is used
- Parity carried out on both directions
- Each byte has a parity bit
  - Even number of 1's: 1 → parity bit
- Each frame has a parity byte
  - Event number of 1's: 1 → corresponding bit in parity byte



Exercise L4-1

- Q1: Sending the following message over a link

  H E L O

  determine its two-dimensional parity bits and byte.

  Assume using the ASCII code (**not** the Extended ASCII).
- Q2: In above case, show an example of received “frame” (i.e., data // parity bits and byte) that has detectable error. Include both data bits and parity bits and byte.
- Q3: Show an example of received “frame” (i.e., data // parity bits and byte) that has non-detectable error.

How Good is Two-Dimensional Parity?

- What types of errors does it catch?
  - Any 1-bit error? 2-bit error? 3-bit error? 4-bit error? …
- How much extra data are needed to detect errors?
- How efficient is the algorithms to compute the EDC and detect errors?

Cyclic Redundant Check (1)

- Error checking code
  - Add k bits of redundant data to an n-bit message
- Quality of the error detection code
  - Low redundancy: $k << n$
  - High probability of detecting errors
  - Can be implemented efficiently
- Polynomial Code: Cyclic Redundant Check (CRC)
- Sender sends message M to receiver
  - Generate a bit string P: M // E
  - How does sender generate E?
  - How does receiver verifies if error?

Cyclic Redundant Check (2)

- Represent *n*-bit string as *n*-1 degree polynomial
  - Bit position as power of each term
  - Digital signal: coefficients are either *0* or *1*
  - Bit string: *11011* as $M(x) = 1\,x^4 + 1\,x^3 + 0\,x^2 + 1\,x^1 + 1x^0 = x^4 + x^3 + x + 1$
- Sender and receiver agrees on a divisor polynomial $C(x)$
  - Digital signal: coefficients are either *0* or *1*
  - Degree of $C(x)$: k
  - Example: $C(x) = x^3 + x^2 + 1$ and $k = 3$

Cyclic Redundant Check (3)

- Algorithm generating M//E
  - Left shift *M* by *k* bits
    - Example
      - 11011 becomes 11011000
      - New polynomial: $T(x) = M(x)x^k$
  - Get remainder of T(x)/C(x)
    - Example: $(x^4 + x^3 + x + 1)x^3 / (x^3 + x^2 + 1) \rightarrow$
      - Result must be 0 or 1: modular 2 arithmetic → "-" = XOR
      - Quotient: $X^4 + 1$
      - Remainder: $R(x) = x^2+1$
  - Subtract R(x) from T(x)
    - Example
      - $(x^4 + x^3 + x + 1)x^3 - (x^2+1) = x^7+x^6+x^4+x^3+x^2+1$
  - The result is M//E
- Send the result to receiver

Previous Example Using Shift and XOR

- Message: 11011000
- Divisor: 1101

```
  11011000
  1101
⊕ ------------
  00001000
      1101
⊕ ------------
      0101
  11011000
⊕ ------------
  11011101
```

Obtain remainder

Obtain M//E

Cyclic Redundant Check (4)

- Algorithm verifying received message
  - Message represented as polynomial T(x)
  - Calculate remainder of T(x) / C(x)
  - If the remainder is not 0, an error
  - Otherwise, *no errors detected*

Cyclic Redundant Check (5)

- Quality of CRC
  - Algorithm efficiency
    - Shift and XOR
  - Redundancy
    - Depends on C(x)
  - Error detection probability
    - Depends on C(x)
- Common CRC Polynomials
  - CRC-8: 1 0000 0111
  - CRC-10: 110 0011 0011
  - CRC-32: used in Ethernet

Exercise L4-2

- Q1: Sending the following data (two bytes in hexadecimal numbers) over a link

  24 A1

  determine the "frame" (data // CRC) to be transmit using CRC-8 (divisor = x^8+x^2+x+1)
- Q2: In above case, show an example of received frame (data // CRC) that contains a detectable error.
- Q3: Show an example of received frame that has non-detectable error.

Summary

- A frame can be corrupted
  - Error detection
- Error detection not 100% reliable! protocol may miss some errors, but rarely
  - larger EDC field yields better detection and correction
- FYI: error handling in general
- Q: How to make the link appear to be reliable despite errors?
  - Reliable transmission

Error Handling: Geometrical Perspective

- This discussion is informational
- Q: why can an Error Detection Code (EDC) detect a certain number of bit errors, and why can not the EDC detect some other number of bit errors?
  - Recall discussion on two dimensional parity code
    - 1-bit error, 2-bit error, 3-bit error, 4, 5, 6 ???
    - answered on case-by-case basis
- Q: Is there systematic way to deal with this problem?

Error Handling (1)

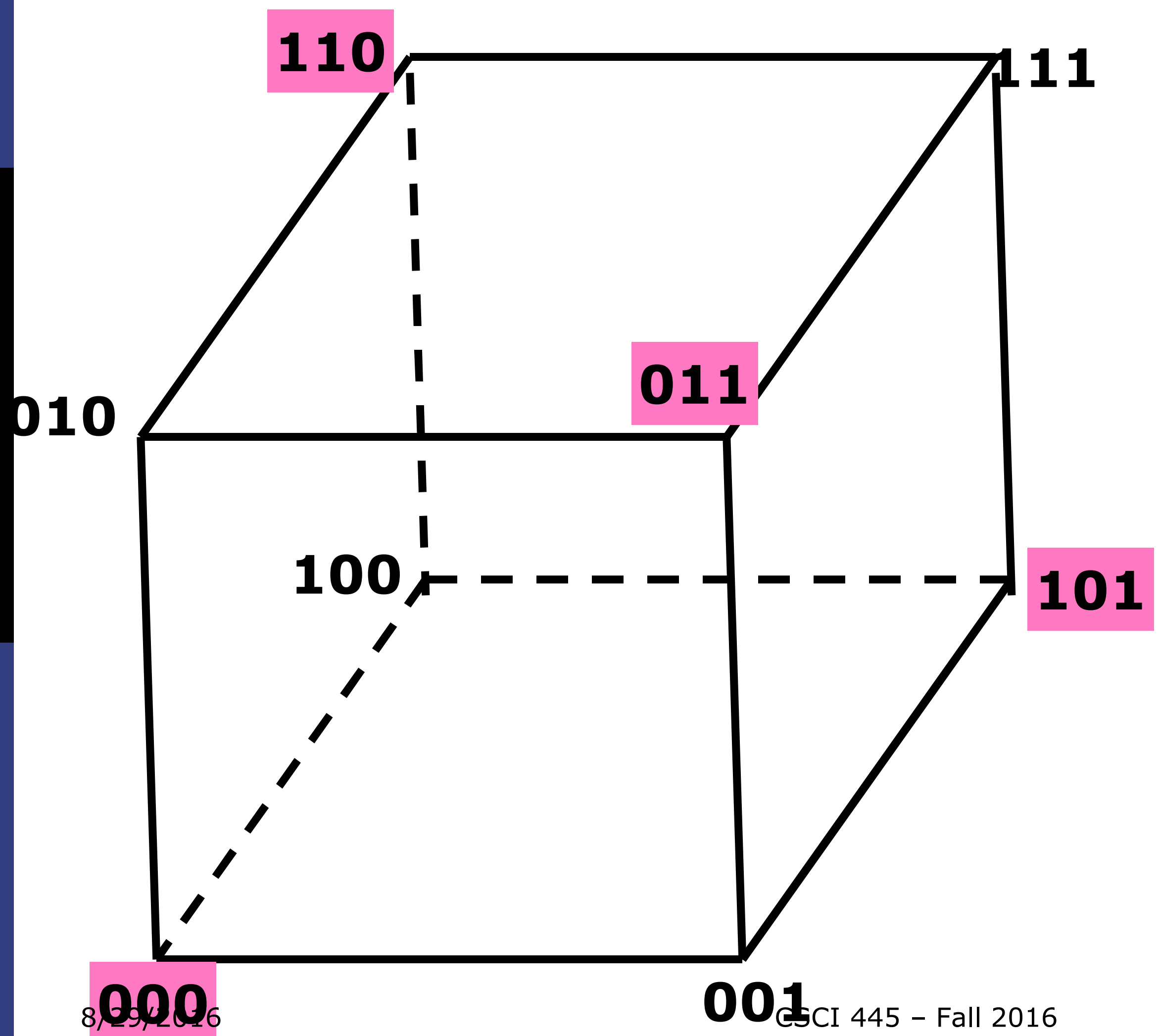
- Error handling
  - Unit of data sent: code words
  - Original data mapped to sequence of code words
  - Send the code words
  - Receiver recovers original data from the received code words
  - Original message m bits → m + k = n bits message → n bit code word
  - What are the lengths of the error detection codes studied?

Error Handling (2)

- Hamming distance
  - # of bit positions in which two code words differ
  - h(10001001, 10110001) = 3
- M → M//K: m → m + k
  - # of total possible bit strings: $2^{(n+k)}$
  - k << (m + k)
- Example code words
  - Message size 2: m = 2
  - 1 bit parity bit: k = 1
  - $2^{(m+k)} = 2^3 = 8$
  - Possible code words: 000, 011, 101, 110
    - # = 4
    - Minimum distance of any pair = 2

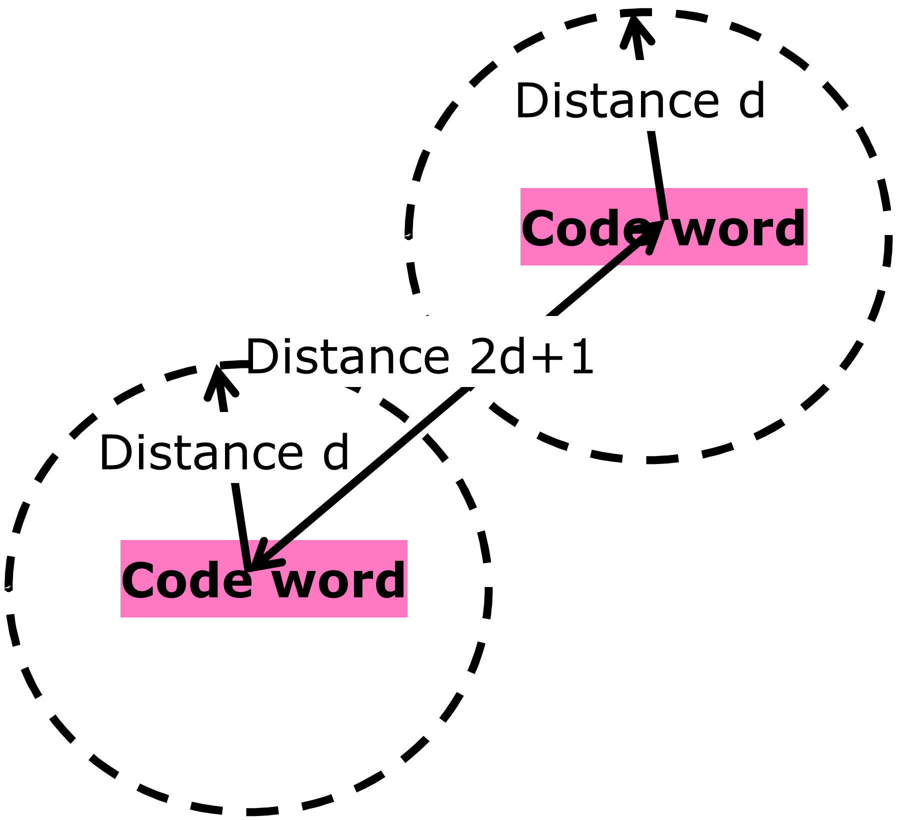
```
 10001001
 10110001
----------
 00111000
```

Error Handling (3)



- Detect 1 bit errors
- Cannot detect any 2-bit errors
- Distance of the code is 2
- d+1 distance code words
  - No d bit difference leads to a valid code
  - detect d errors

Error Handling (4)



- Correct d errors, need distance 2d + 1 code words
  - After d errors, the closest code word remains the correct one.
  - Code words 5 = 2x2+1
    - 00000 00000
    - 00000 11111
    - 11111 00000
    - 11111 11111
    - Correct at most 2 errors

Error Handling (5)

- Observation
  - 2d + 1 distance code → correct d errors
  - 2d + 1 distance code → detect 2d errors
- Error correction codes generally more redundant
- Error correction or error detection?
  - Error detection example: *m + k* with error rate *r*
    - N (m + k) + r N (m + k) with error correction
  - Error correction example: m + K with error rate *r* and K >> k
    - N (m + K)
  - N (m + k) + r N (m + k) - N (m + K) = N k + r N (m + k) - NK = N (r + rm + rk) – N K = N (r + rm + rk – K)
  - r + rm + rk – K > 0? r + rm + rk – K < 0?

Two-Dimensional Parity Code as Error Correction Code (1)

```
              0 0 0
              0 1 0
Parity byte → 0 0 0
                  ↑
              Parity bit
```

- Assuming even parity, is there any bit error?
- Assuming 1 bit error, where is the error?

Two-Dimensional Parity Code as Error Correction Code (2)

0 0 0
1 1 0
Parity byte → 0 0 0
↑
Parity bit

- Assuming even parity, is there any bit error?
- Assuming 2 bit error, where are the errors?

Two-Dimensional Parity Code as Error Correction Code (3)

```
0 0 0          0 0 0
1 1 0          0 0 0
0 0 0          0 0 0

                        ?

               1 1 0
               1 1 0
               0 0 0
```

Two-Dimensional Parity Code as Error Correction Code (4)

- How many bit errors can two-dimensional parity code correct?
  - 1-bit error?
  - 2-bit error?
  - ……
- Flip 1 bit → 3 bits are flipped
  - Minimum distance is $3 = 2 \times 1 + 1$
  - Then?

Summary

- A frame can be corrupted
  - Error detection and correction
- Error detection not 100% reliable! protocol may miss some errors, but rarely
  - larger EDC field yields better detection and correction
- Q: How to make the link appear to be reliable despite errors?
  - Reliable transmission