Error Detection

- Add redundant bits
  - simple case
    - two copies of data
    - receiver compares copies ‘equal’ then no error.
    - probability of same bits corrupted low.
  - Add k bits << n bits (n is message length)
  - Example: 12,000 bits (1500 byte) cost 32 bit CRC.
- Why redundant bits?
  - Redundant bits are used by receiver to detect errors

Error Detection: 2-d parity

- Two dimensional (2-d) parity

0	1	0	1	0	0	1	1
1	1	0	1	0	0	1	0
1	0	1	1	1	1	0	1
0	0	0	1	1	1	0	1
1	0	1	1	1	1	1	0
1	0	0	1	1	1	1	1

parity bits

parity byte

Error Detection: 2-d parity

- Add 1 bit to a seven bit code
  - catches all 1 - 2 - and 3 & 4 bit errors along a row
  - extra byte are redundant information.
  - does not add information.
- Additionally parity byte enables detection of errors along a column

Error Detection: Check Sum

- Algorithm based on addition of all the codes used to encode the data.
- send Check Sum
- receiver also computes Check Sum
- Internet Check Sum Algorithm:
  - Example: 16 bit integers –treat data as 16 bit integers
  - Add using 16 bit one's complement.
  - take one's complement of result

Frame Error: A probabilistic Estimate

- Let probability that 1 bit is in error be p
  - Probability that no bit is in error in a 10000 bit packet is:
    - $(1-p)^{10000}$
  - Probability that 1 bit is in error
    - $10^4 p(1-p)^{99999}$
  - Probability that at least 1 bit is in error
    - $1-(1-p)^{10000}$

Error Detection: CRC

- CRC (Cyclic Redundancy Check)
  - goal to maximise the probability of detecting an error
  - nth degree polynomial
  - value of each bit is a coefficient
    - Example: **10011100**
    - $\mathbf{M(x) = x^7 + x^4 + x^3 + x^2}$
  - sender and receiver exchange polynomials

Error Detection: CRC

- Agreed upon polynomial C(x), degree k
- Message exchanged:
- M(x) + k bits = P(x)
- Make P(x) exactly divisible by C(x).
- If no errors at receiver
- P(x) / C(x) – zero remainder => no errors
- B(x) of degree > C(x) => B(x) divisible by C(x)
- B(x) of degree = C(x) => B(x) divisible once by C(x)
- B(x) – C(x) = remainder
- subtract C(x) from B(x)
  - EXOR on matching pair of coefficients.

CRC Algorithm

- Step1: Compute M(x) * x^k
  - equivalent to adding **k** zeros
  - example: M(x) = **1000, C(x) of degree 2**
  - $\mathbf{x^3 * x^2 = x^5}$ = **T(x) (10000)**
- Step2: Divide T(x) by C(x)
- Step3: Find remainder T(x) / C(x) = R(x)
- Step4: subtract T(x) – R(x) = D(x)
  - **D(x) is exactly divisible by C(x)**
- Step5: Transmit D(x)

CRC - An example

- Example:
  - – $M(x) = 101010$
  - – $C(x) = x^3 + x^1$ (1010)
  - - Message transmitted is:
    - **101010100 is transmitted**
    - **101010100 is exactly divisible by 1010**

```
         10001
      ___________
1010 | 101010000
       1010
       ----
          1000
          1010
          ----
          00100  - Remainder
```

101010000 – Message padded with 3 zeros

000000100 -- Remainder

101010100 – Message xored with remainder

CRC Standards

- **CRC - 8 : x8+ x2 + x1 + 1**
- **CRC - 10 : x10 + x9 + x5 + x4+ x1 + 1**
- **CRC – 12: x12 + x11 + x3 + x2+ 1**
- **CRC – 16: x16 + x12 + x5 + 1**
- **CRC – CCITT: x16+ x12 + x5 + 1**
- **CRC – 32: x32 + x26 + x23+ x22+ x16 + x12 + x11 + x10 + x8+ x7 + x5 + x4 + x2 + x + 1**

Characteristics of CRC

- detect all single bit errors as long as x^k & x^0 have non zero coefficients.
- detect double bit errors as long as C(x) has at least three terms.
- any odd number of errors as long as C(x) has a factor (x+1)
- any burst error of length < k bits can also be detected.

Error Detection and Correction

- Code m + r
  - m bit message, r check bits
- Hamming distance of code:
  - Minimum distance between any two code words in a code
- To detect d errors d+1 code
- To correct d errors 2d+1 code

Error Correction

- **Example:**
- **0000000000**
- **0000011111**
- **1111100000**
- **1111111111**

  } **code**
- **Hamming distance = 5**
- **Example:**
- **If 0000000111 received**
- **- has to be 0000011111**
- **provided double bit errors.**