CSED601
Dependable Computing
Lecture 8

Jong Kim

Dept. of CSE

POSTECH

Cyclic codes

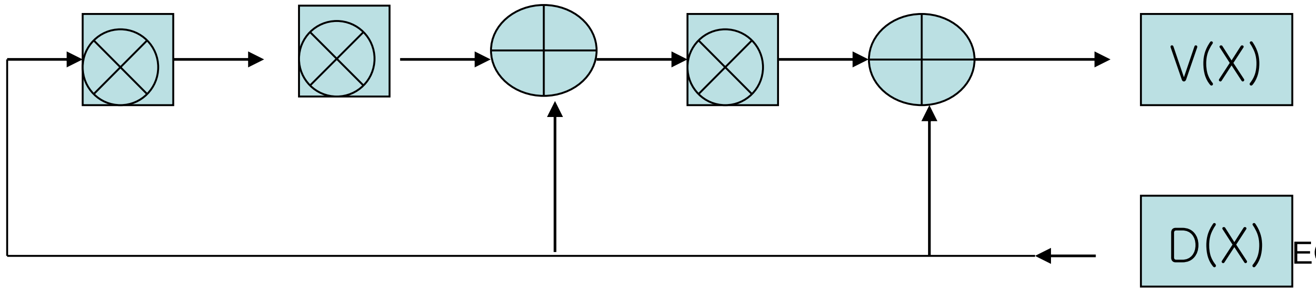
- Concept
  - Any end-around shift of a code word will produce another code word.
  - Frequently used to sequential access devices such as tapes, bubble memory, and disks.
  - Encoding operation can be implemented using simple shift registers with feedback connections.
  - Characterized by its generator polynomial G(X), which is a polynomial of degree n-k or greater.
  - The n bits are contained in the complete code word and k bits are in the original information to be encoded.
  - A cyclic code with a G(X) of degree n-k is called a (n, k) cyclic code.
  - Can detect all single errors and all multiple adjacent errors affecting fewer than (n-k) bits.
  - Very important in communication where burst error can occur.

Cyclic code

- Implementation
  - For code word $V = (v_0, v_1, \ldots, v_{n-1})$,

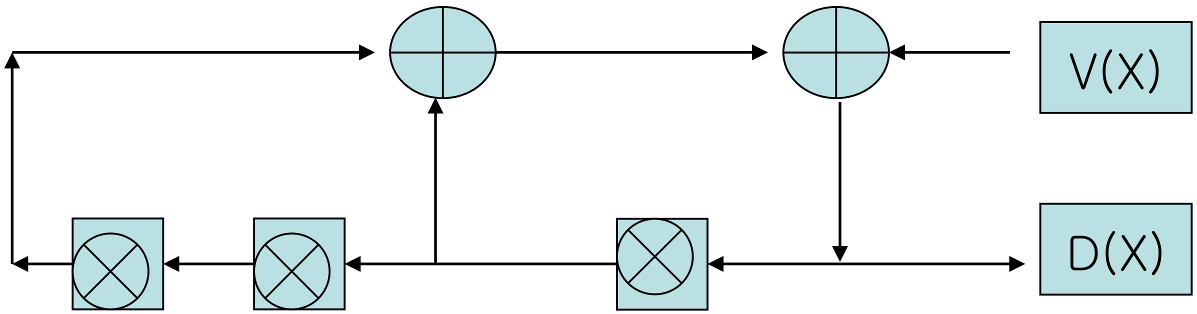
    $V(X) = v_0 + v_1X + v_2 X^2 + \ldots + v_{n-1} X^{(n-1)}$
  - Code polynomial for a non-separable cyclic code is generated by multiplying a polynomial, representing the data to be encoded by another polynomial known as the generator polynomial, G(X).
  - Any additions during multiplication is performed by using modulo-2
  - Eg.: $V(X) = D(X) * G(X)$

    $(1101) * (1101) = (1+X+X^3)(1+X+X^3) = 1+X^2+X^6 = (1010001)$
  - Use a special hardware to generate a code word



Cyclic code

- Decoding procedure
  - Received code word R = (r0, r1, …, rn-1)
  - R(X) = D(X)G(X) + S(X)
  - S(X) should be zero if R(X) is a valid code word.
  - S(X) is called the syndrome polynomial.
  - R(X) should be an exact multiple of generator polynomial G(X) ➔ Divide R(X) by G(X) and see the remainder of division is zero.
  - Disadvantage : Non-separable code



Cyclic code

- Separable cyclic code
  - How to make a separable code :

    $V(X) = R(X) + X^{\wedge(n-1)} D(X)$

    Where R(X) is the remainder polynomial obtained by dividing D(X) with G(X)
  - Eg.: G(X) = 1+X+X^3, D(X) = 1+X^3

    X^(n-k) D(X) = X^3 D(X) = X^3 + X^6

    R(X) = X^2 + X

    V(X) = X^3 D(X) + R(X) = X^6+X^3+X^2 + X

Cyclic Redundancy Check (CRC)

- Characteristics
  - Separable code
  - Used to check communication error
  - Block error checking method
- Method
  - CRC = D(X) / G(X)
  - Send the final remainder information to the other part
  - CRC-16, CRC-CCITT : fix generator polynomial

Arithmetic Codes

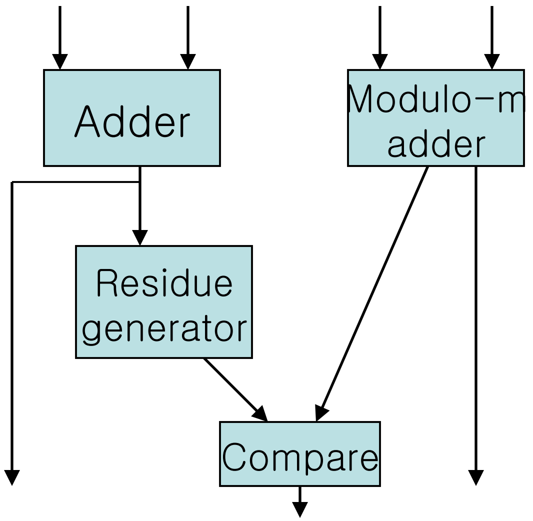
- Concept
  - Some arithmetic code are invariant to a set of arithmetic operations: A(b*c) = A(b) * A(c)
  - Used to check arithmetic operations such as addition, multiplication, and division.
- Operation
  - The data presented to the arithmetic operation is encoded before the operation is performed.
  - After completing the arithmetic operation, the result code word must be checked to make sure that they are valid.
- Methods
  - AN codes
  - Residue codes
  - Inverse-Residue codes
  - Residue Number System (RNS)

AN Codes

- Concept
  - Multiplying each data word N by some constant A.
  - AN codes are invariant to addition and subtraction, but not to multiplication and division : $AN_1+AN_2 = A(N_1+N_2)$
- Operations
  - Check whether the result is divisible by A.
  - The magnitude of the constant A determines the number of extra bits required to represent the code words and the error detection capability.
  - A should not be power of 2
    - If A=2^a, it is not capable of detecting single bit error.
  - Eg.: 3N codes need n+2 bits
  - 3N codes can be checked by using a simple combinational ckt.

Residue Codes

- Concept
  - Separable arithmetic code created by appending the residue of a number to the number ➔ D | R
- Operation
  - N = I m + r ➔ N / m = I + r / m where r is residue and m is check base.
  - The number of bits for residue depends on modulus
    - Eg.: n+2 bits for modulus of 3
  - Advantage:
    - Invariant to the operations of addition
    - Residue can be handled separately
  - Low cost residue code when m = 2^b – 1
    - Number of extra bits = b
    - Easy encoding process
      - Divide data bits into k groups of b bits
      - Add in modulo-b addition



Inverse Residue Codes

- Concept
  - Similar to residue codes
  - Inverse residue code Q=m-r
  - Have better fault detection capability for repeated-use fault
    - Repeated addition for multiplication

Residue Number System (RNS)

- Concept
  - Numbers are represented by a set of residues
  - Does not produce a separable code
- Operation
  - Define a relatively prime moduli
    - P = [p1,p2,…,pk] where M = p1*p2*…*pk
    - 0 <= N <= M where N is the number to be represented
    - Find residue for each modulus
    - Represent N = (r1,r2, …., rk) where r1 = N modulo p1, ….
    - Eg.: 32 in [3,4,5] ➔ (2,0,2)

Residue Number System (RNS)

- Operation
  - Advantage
    - Carry free number system
    - Arithmetic can be performed on the individual digits of numbers
    - Can add speedly
    - Have error detection capability
    - Add redundant modulus for error detection
      - If the number is between 0 and M, then valid code word
      - If the number is beyond M, then invalid code word
  - Disadvantage
    - Conversion to normal number system is not easy.

Berger Code

- Concept
  - Make a valid code word by appending a special set of bits
  - Check bits are created based on the number of 1's in the original information.
- Operation
  - Code length n = I + k
    - I is the number of information bits
    - k = log (I+1) and n = I + k
  - Eg.: D =(0111010), I = 7, k = 3

    1's in I = 4 ➔ 100 ➔ Complement 011

    Resulting code word = (0111010 011)
  - The overhead depends on the number of information bits
  - Separable and detect all multiple and unidirectional errors
  - Berger code use the fewest number of check bits of the available separable codes.

Horizontal and Vertical Parity

- Concept
  - Use both horizontal and vertical parity
  - Can detect and correct one single bit error
  - For multiple errors, can detect if it shows an odd sequence.
  - Used for block of data

Hamming Error Correcting Code

- Background
  - ECC is commonly used in memory design.
  - Memory is relatively inexpensive:10~40% redundancy
  - Efficient in terms of the time required to perform the correction process
  - Error correction circuit is readily available on inexpensive chips
  - Memory error is 60~70% of the faults in a system
- Concept
  - Use c parity check bits to protect k bits of information (single-bit correction code)
    - $2^c \geq c + k + 1$ total length $n = c + k$
    - K check codes to indicate an error in one of k bits
    - C check codes to indicate an error in one of c bits
    - At least one check code to indicate no-error

Hamming Error Correcting Code

- Example
  - N= (d3, d2, d1, d0) (c2, c1, c0) where (c0 = d3 Xor d1 Xor d0) (c1 = d3 Xor d2 Xor d0) (c2 = d3 Xor d2 Xor d1)
  - Add one additional parity check bit to differentiate single/ multiple errors.
- Code word generation scheme
  - Let V = code word, D = data word
  - V = [v1,v2,…,vn] = [d1,d2,…,dk] [g11…. g1n]

    […………..]

    [gk1……gkn]

    = D x [G]

  where G is called Generator Matrix and code word is called (n,k) code
  - To be a separable code, G should be G = [I | P] where I is k X k identity matrix and P is k X [n-k] Parity generator matrix.
  - The result would be [k bits | n-k bits]

Hamming Error Correcting Code

- Code word checking scheme
  - Let H be parity checking matrix
  - $V \times H^{\wedge T} = 0$ where $V = D \times [G] = D \times [\ I \mid P\ ]$

    ➔ $D \times G \times H^{\wedge T} = 0$ hence $G \times H^{\wedge T} = 0$
  - Then how to find H from the given G?
  - G and H should be orthogonal to produce 0
  - $G = [\ I_k \mid P\ ]$ and $H = [P^{\wedge T} \mid I_{n-k}\ ]$
  - Eg.: G = [1000 101] then H = [1110 100]

    [0100 111] [0111 010]

    [0010 110] [1101 001]

    [0001 011]

Hamming Error Correcting Code

- Code word checking scheme
  - Let R be the transmitted information
  - H x R^T = S (syndrome)
  - S = 0 if there is no error.
  - Let r = f + ei where f is the code word and ei is the error at ith bit.
  - Then, H r = H f + H ei since H f = 0
  - H r = H ei ← error condition
  - If S matches with the ith column of H matrix, then ith bit has an error.

Code Selection Issues

- Checking issues
  - Other forms of redundancy exist?
  - Check the amount of redundancy required
- Code selection criteria
  - Whether or not the code needs to be separable
  - Error detection / error correction is required
  - Number of bit errors that need to be detected or corrected

Error Model

- Symmetric error : 0 -> 1 and 1 -> 0 errors are equally likely.
- Asymmetric error : A given word or operational unit has 0 -> 1 or 1 -> 0 errors not equally likely.
- Unidirectional error : When all components affected by a multiple error change their values in one direction from, say, 0 to 1 or vice versa.
- B-adjacent error: An error that affects only component values within a byte of b-bit width is classified as a single b-adjacent error.