ECN 104

Digital Logic Design

Spring 2023

Tanmoy Pramanik

http://ece.iitr.ac.in/tanmoy_pramanik

Acknowledgment: Content mostly taken from many textbooks

Need for error detection & correction

- Physical world is non-ideal; there are always chances of error that can flip data bits
  - During storage in memory, some bit may flip due to noise
  - During read/write there may be errors – bit may be written incorrectly or read incorrectly
  - Error may happen due to the transmission/reception of bits across a network
- Without a check for the correctness of data received for processing, all systems will become unreliable.
- However, any error correction or detection will incur additional overhead, in systems with a high degree of reliability, error correction/detection may be avoided.
- Also, if an application can be tolerant to a small number of errors, then such techniques may be avoided.

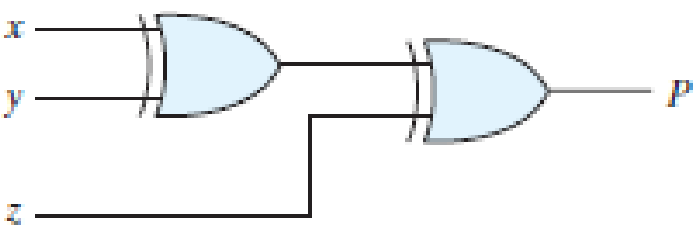
Simplest error detection scheme – Parity bit

- Generate and store the parity bit along with data

Table 3.3
Even-Parity-Generator Truth Table

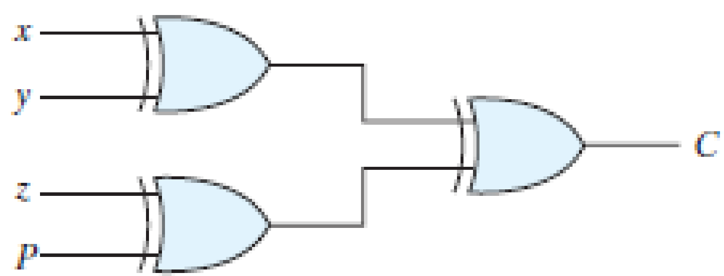
Three-Bit Message			Parity Bit
x	y	z	P
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Generator



(a) 3-bit even parity generator

Checker (C=1 if there is an error)



(b) 4-bit even parity checker

- Fails for more than 1-bit flip
- There is no scope for correction

Hamming Code

- More complex error correction techniques create multiple parity check bits that are stored along with the data.

- Each check bit is for a group of data bits. From such check bits, errors can be detected and corrected to a limited extent.

- Most common technique is known as "Hamming Code" named after R.W. Hamming

Hamming Code - example

- Add k parity bit to the n bit data word
- Store (n + k) bits instead of n bits, bit positions are numbered in sequence from 1 to (n + k)
- Positions numbered as power of 2 are reserved for parity bits.

Example: 8-bit data (or code) word, 11000100, add 4 parity bits as follows

Bit position:	1	2	3	4	5	6	7	8	9	10	11	12
	P_1	P_2	1	P_4	1	0	0	P_8	0	1	0	0

Parity bits are P1, P2, P4 & P8 in positions (1,2,4,8) respectively.

Hamming Code – generate the check bits

Example: 8-bit data word, 11000100, add 4 parity bits as follows

Bit position:	1	2	3	4	5	6	7	8	9	10	11	12
	P_1	P_2	1	P_4	1	0	0	P_8	0	1	0	0

$$P_1 = \text{XOR of bits } (3, 5, 7, 9, 11) = 1 \oplus 1 \oplus 0 \oplus 0 \oplus 0 = 0$$

$$P_2 = \text{XOR of bits } (3, 5, 7, 10, 11) = 1 \oplus 0 \oplus 0 \oplus 1 \oplus 0 = 0 \longleftarrow 3,6,7,10,11$$

$$P_4 = \text{XOR of bits } (5, 6, 7, 12) = 1 \oplus 0 \oplus 0 \oplus 0 = 1$$

$$P_8 = \text{XOR of bits } (9, 10, 11, 12) = 0 \oplus 1 \oplus 0 \oplus 0 = 1$$

So the stored data in memory would be:

	0	0	1	1	1	0	0	1	0	1	0	0
Bit position:	1	2	3	4	5	6	7	8	9	10	11	12

Hamming Code – verify the data

Example: 8-bit data word, 11000100, after storing becomes,

	0	0	1	1	1	0	0	1	0	1	0	0
Bit position:	1	2	3	4	5	6	7	8	9	10	11	12

When the data is read back from memory, following checks are done:

$$C_1 = \text{XOR of bits } (1, 3, 5, 7, 9, 11)$$

$$C_2 = \text{XOR of bits } (2, 3, 6, 7, 10, 11)$$

$$C_4 = \text{XOR of bits } (4, 5, 6, 7, 12)$$

$$C_8 = \text{XOR of bits } (8, 9, 10, 11, 12)$$

Make $C = C_8C_4C_2C_1$, the any non-zero C will indicate error and the error-bit position is given by the number C.

Hamming Code – correcting errors

Bit position:	1	2	3	4	5	6	7	8	9	10	11	12	
	0	0	1	1	1	0	0	1	0	1	0	0	No error
	1	0	1	1	1	0	0	1	0	1	0	0	Error in bit 1
	0	0	1	1	0	0	0	1	0	1	0	0	Error in bit 5

C_1 = XOR of bits (1, 3, 5, 7, 9, 11)

C_2 = XOR of bits (2, 3, 6, 7, 10, 11)

C_4 = XOR of bits (4, 5, 6, 7, 12)

C_8 = XOR of bits (8, 9, 10, 11, 12)

	C_8	C_4	C_2	C_1
For no error:	0	0	0	0
With error in bit 1:	0	0	0	1
With error in bit 5:	0	1	0	1

Deciding the number of check bits

- For k check bits, the number C will range from 0 to $2^k - 1$.
- Out of a total 2^k values, C = 0 is usually reserved to indicate no error.
- Remaining $2^k - 1$ values should be able to uniquely indicate bit positions within a $(n+k)$ bit data word.
- Choose k such that, $2^k - 1 \geq n + k \Rightarrow 2^k - 1 - k \geq n$

Table 7.2
Range of Data Bits for k Check Bits

Number of Check Bits, *k*	Range of Data Bits, *n*
3	2–4
4	5–11
5	12–26
6	27–57
7	58–120

- The grouping is decided by which binary bits contribute to 1 at a particular position

Single Error Correction Double Error Detection (SECDED)

- Add a parity bit on top of the 12-bit code: $001110010100P_{13}$
- Once the code is fetched, check C and the parity P over all 13 bits.

If $C = 0$ and $P = 0$, no error occurred.
If $C \neq 0$ and $P = 1$, a single error occurred that can be corrected.
If $C \neq 0$ and $P = 0$, a double error occurred that is detected, but that cannot be corrected.
If $C = 0$ and $P = 1$, an error occurred in the P_{13} bit.

(Note that C is the 4-bit number as before and P is just 1-bit)

There are similar schemes DECTED, TECQED etc.

Hamming Distance

- The minimum number of bits changed between two code words in a set is defined as the Hamming distance of that set
- Let C_i and C_j be any two code words in a particular encoding scheme. Hamming distance d_{ij} is the number of bits changed from C_i to C_j.

$$\begin{matrix} C_i & = & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ C_j & = & 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{matrix} \quad \Rightarrow d_{ij} = 3$$

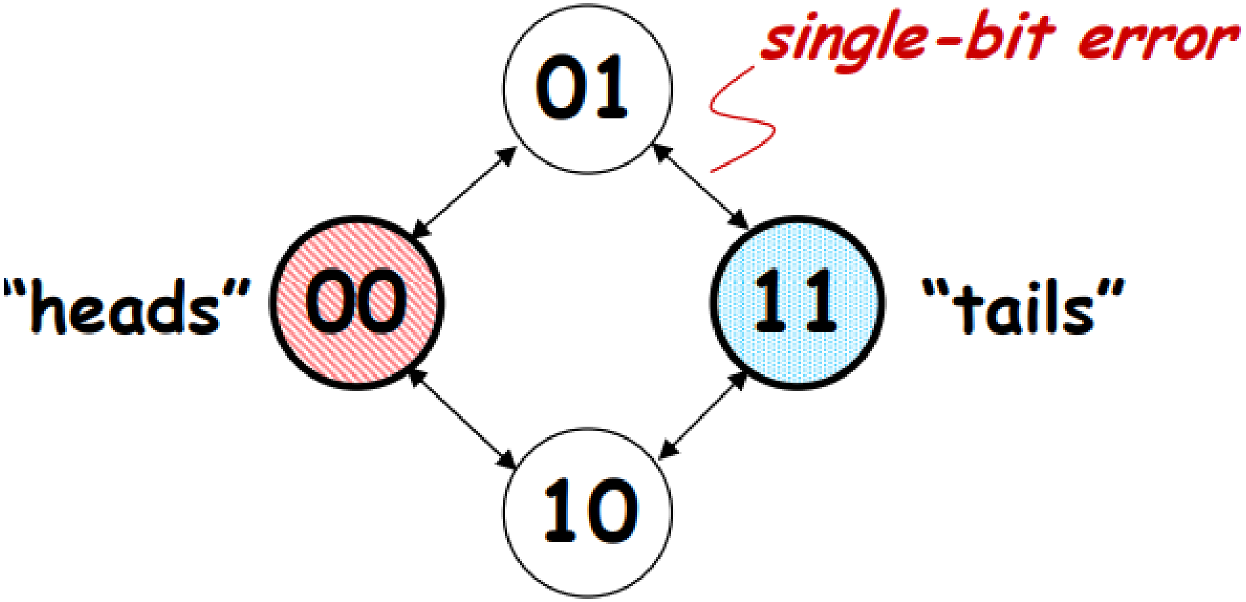
- *Hamming distance* for the whole encoding scheme is gives as:

$$d = \min(d_{ij}) \;\; for\ all\ possible\ (i, j), but\ with\ i \neq j$$

Hamming Distance for ECC

- Hamming noted the following observations from the analysis of code distances,
  - A minimum distance of at least two or higher is needed to detect single errors.
  - Since the number of errors E <=(d_min-1)/2, a minimum distance of three is needed for single error correction.
  - Greater distances will provide detection and/or correction of more number of errors.
- Verify with our simple parity-bit-based error detection

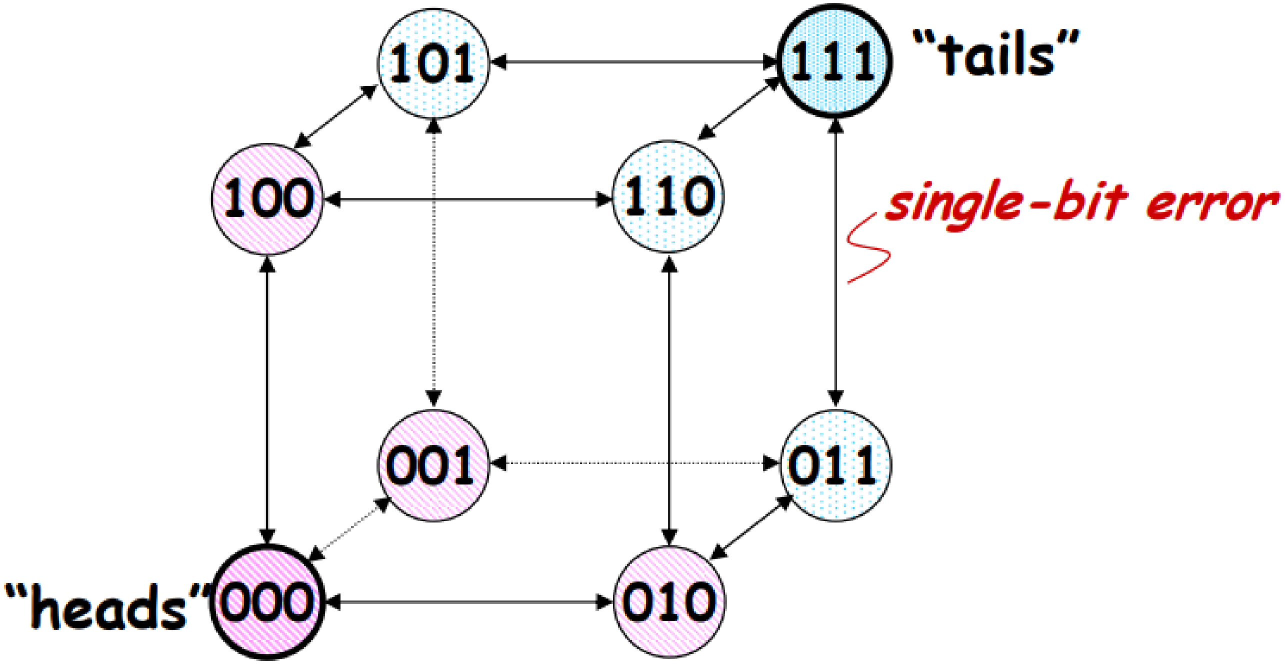
Simple parity bit scheme



If D is the minimum Hamming distance between code words, we can detect up to (D-1)-bit errors

http://web.mit.edu/6.02/www/f2006/handouts/bits_ecc.pdf

Hamming Code Scheme



If D is the minimum Hamming distance between code words, we can correct up to $\left\lfloor \frac{D-1}{2} \right\rfloor$ - bit errors

http://web.mit.edu/6.02/www/f2006/handouts/bits_ecc.pdf