Hamming Code (Single Bit Error Correction and Double Bit Error Detection)

The Hamming Code is an error control technique developed by R.W. Hamming for the correction of errors in data transmission. It works by adding redundant parity bits to a binary data sequence to detect and correct errors during transmission. Hamming codes are widely used in computer memory systems, communication channels, and other applications where reliable data transmission is crucial. This method is capable of correcting all single-bit errors and detecting (but not correcting) all double-bit errors.

Calculation of Redundant Bits
If m = number of data bits and r = number of redundant bits, then the formula is:
$2^r \geq m + r + 1$
This inequality ensures that the number of distinct error syndromes generated is sufficient to identify every possible single-bit error and the 'no error' condition. Example: For m = 7 data bits, r = 4 redundant bits are required because:
$2^4 = 16 \geq 7 + 4 + 1 = 12$.

Positioning of Bits
The positions of the bits in the final codeword are numbered starting from 1. Parity bits are placed at positions that are powers of two: 1, 2, 4, 8, … These positions correspond to P1, P2, P4, P8, etc. The remaining positions are filled with data bits D3, D5, D6, D7, etc. This arrangement allows each parity bit to check specific positions based on binary representation.

Example – Single Bit Error Correction
Let the data bits be: D1 = 1, D2 = 0, D3 = 1, D4 = 1.
Placed in positions: P1, P2, D1(1), P4, D2(0), D3(1), D4(1) → Initial codeword: P1, P2, 1, P4, 0, 1, 1.
Step 1: Calculate Parity Bits (Even Parity)
P1 checks positions 1, 3, 5, 7 → (P1, 1, 0, 1) → two 1’s → P1 = 0.
P2 checks positions 2, 3, 6, 7 → (P2, 1, 1, 1) → three 1’s → P2 = 1.
P4 checks positions 4, 5, 6, 7 → (P4, 0, 1, 1) → two 1’s → P4 = 0.
Final transmitted codeword: 0 1 1 0 0 1 1.

Step 2: At the Receiver – Error Detection
Suppose bit 5 changes from 0 to 1 during transmission: Received: 0 1 1 0 1 1 1.
C1 (P1 check) = odd → C1 = 1.
C2 (P2 check) = even → C2 = 0.
C4 (P4 check) = odd → C4 = 1.
Error position = C4C2C1 = 101■ = 5 → flip bit 5 to restore original data.

Example – Double Bit Error Detection
To detect double-bit errors, add an extra parity bit (Poverall) to the MSB of the codeword. Poverall covers all bits in the codeword, including the parity bits. This enables the detection of any two-bit error without confusion with single-bit error correction.
Error Detection Rules:
C = 0, P = 0 → No Error.
C ≠ 0, P = 1 → Single-bit error (Correctable).
C ≠ 0, P = 0 → Double-bit error (Detected, not correctable).
C = 0, P = 1 → Error in Poverall bit.

Worked Example:
Original codeword with Poverall: 1 0 0 1 0 1 0 0 1 1 1 0 0.
If bits 3 and 6 are corrupted, the receiver computes C ≠ 0 and P = 0, indicating a double-bit error.

Why Hamming Code Works
Hamming code uses the positions of bits and the binary representation of these positions to determine which bits each parity bit will check. Each parity bit checks a unique combination of data bits, so that the binary syndrome generated at the receiver directly maps to the position of the erroneous bit. In the case of double-bit errors, the syndrome may point to an incorrect position, but the Poverall bit differentiates between single and double-bit errors.

Summary:
Hamming Code corrects all single-bit errors and detects double-bit errors by adding redundant parity bits at positions that are powers of two. With an additional overall parity bit, it can detect double-bit errors with certainty. It is a powerful and efficient method for error control in systems where single-bit errors are common and cost-effective detection is needed.