Error Detection / Correction

Error Detection / Correction

- Why might we need Error detection/correction?

- Even & Odd Parity
  - Error detection

- Hamming code
  - Used for error detection & error correction

Parity bits

- **ASCII – 7 bit code (hex 00 to 7F)**

  Could use "8th" bit for parity bit:

  X1011010

  - **Even parity: make total number of "1" bits is even**

    01011010

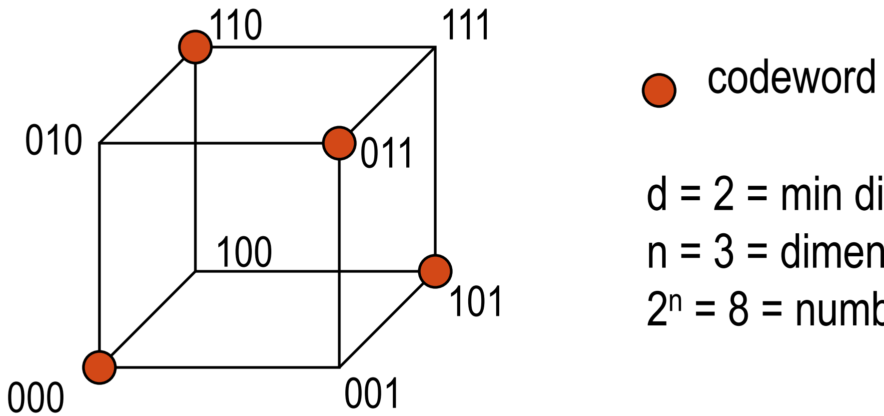
  - **Odd parity: make total number of "1" bits odd**

    11011010

If a parity bit is added to a bit stream, then there is a basis to check for bit(s) being corrupted.

Hypercube Interpretation

- Consider codewords as vertices on a hypercube.



d = 2 = min distance
n = 3 = dimensionality
2^n = 8 = number of nodes

The distance between nodes on the hypercube is the Hamming distance D. The minimum distance is d.

001 is equidistance from 000, 011 and 101.

For s-bit error detection $d \geq s + 1$

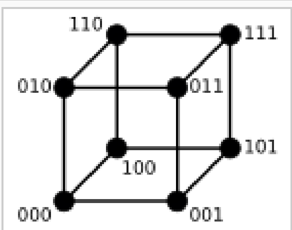
For s-bit error correction $d \geq 2s + 1$

Hamming Distance

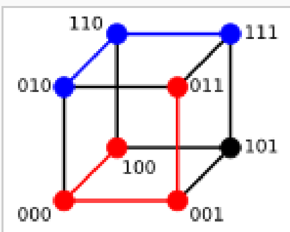
- The Hamming distance is the number of bits that have to be changed to get from one bit pattern to another.

Example: 10010101 & 10011001 have a hamming distance of 2

- For any coding whose members have a Hamming distance of two, any one bit error can be detected. Why?



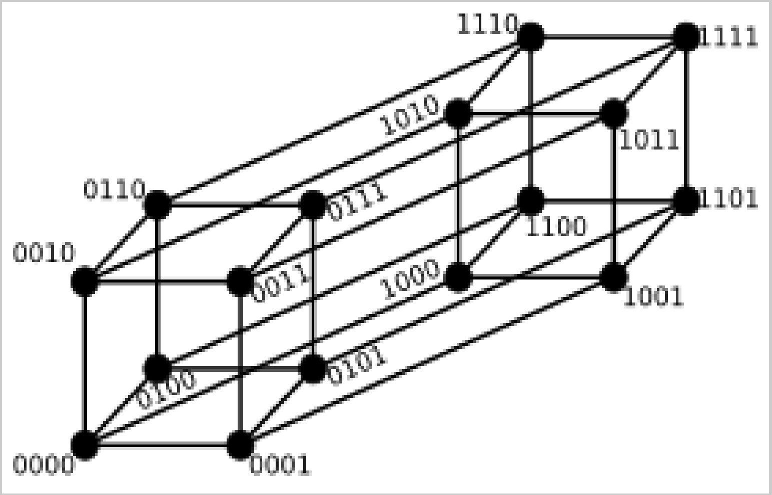
3-bit binary cube for finding hamming distance



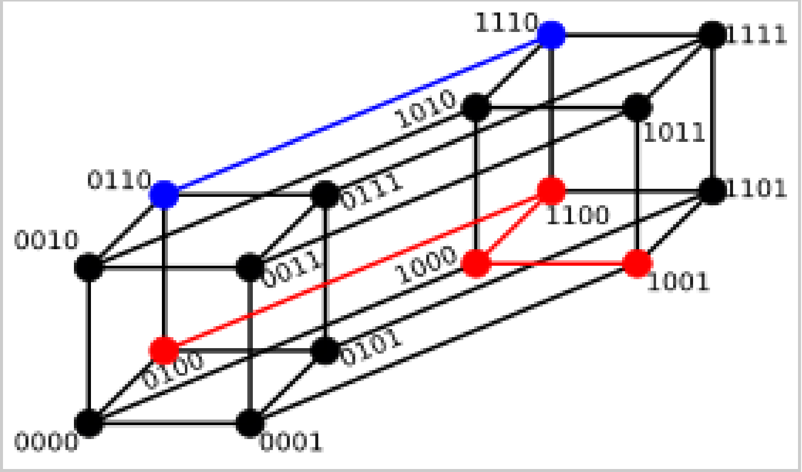
Two example distances: 100->011 has distance 3 (red path); 010->111 has distance 2 (blue path)

Hamming Distance

- For any coding whose members have a Hamming distance of three, any one bit error can be detected and corrected. Why?
- And any two bit error can be detected. Why?

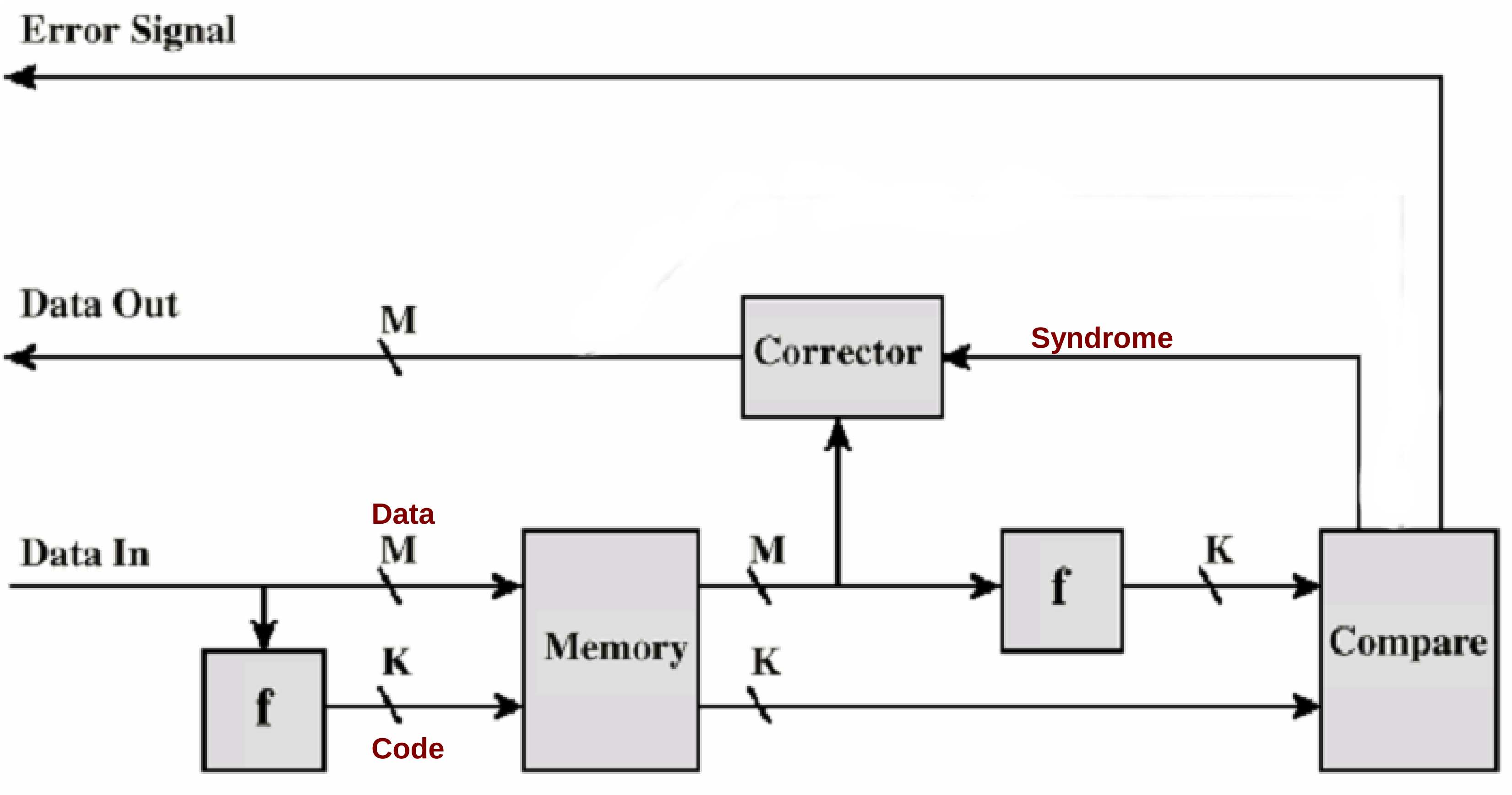


4-bit binary hypercube for finding hamming distance



Two example distances: 0100->1001 has distance 3 (red path); 0110->1110 has distance 1 (blue path)

Error Correcting Code Function



The output of the "Compare" to the "Corrector" is termed the "syndrome", and is K bits long

Hamming Code Syndrome

- If we compare the read K bits compared with the write K bits, using an EXOR function, the result is called the "syndrome".

- If the syndrome is all zeros, there were no errors.

- If there is a 1 bit <u>somewhere</u>, we know it represents an error.

Hamming Code Design – determining K

To store an M bit word with detection/correction takes M+K bit words

If K =1, we can detect single bit errors but not correct them

If $2^K - 1 \geq M + K$, we can detect, identify, and correct all single bit errors, i.e. the syndrome contains the information to correct any single bit error

Example: For M = 8:

and K = 3: $2^3 - 1 = 7 < 8 + 3$ (doesn't work)

and K = 4: $2^4 - 1 = 15 > 8 + 4$ (works!)

Therefore, we must choose K =4,

i.e., the minimum size of the syndrome is 4

Increased word length for error correcting

	Single-Error Correction		Single-Error Correction/ Double-Error Detection	
Data Bits	Check Bits	% Increase	Check Bits	% Increase
8	4	50	5	62.5
16	5	31.25	6	37.5
32	6	18.75	7	21.875
64	7	10.94	8	12.5
128	8	6.25	9	7.03
256	9	3.52	10	3.91

Hamming code

- 01001101 ---- Data
- ??0?100?1101 ---- ? For 1,2,4,8
- P1= ??0?100?1101
  - P1=?01010
  - Even ? 0
- P2= ??0?100?1101
  - P2=?00010
  - Even ? 1
- P4= ??0?100?1101
  - P4=?1001
  - Even ? 0
- p8= ??0?100?1101
  - P8=?1101
  - Even ? 1

Detecting and correcting

- 01001101 ---- Data
- Transmitted data with hamming code: 010010011101 ---
- Suppose error in bit 9--- 010010010101
- P1= 010010010101
  - P1=001000
  - Even ? 1
- P2= 010010010101
  - P2=100010
  - Even ? 0
- P4= 010010010101
  - P4=01001
  - Even ? 0
- p8= 010010010101
  - P8=10101
  - Even ? 1
- correcting----- 1+0+0+8=9