Error Correction

Semiconductor memory errors 2 types.

1. Hard Failure
   - Permanent defect — Gate became stuck at 0 or 1.
2. Soft Error
   - Random, non-destructive — power supply problems or alpha particles.
   - No permanent damage to memory
- Detected using Hamming error correcting code

Suppose we want to store M-Bit word of data.
a calculation is performed using a function.
This calculation yields a code. Length of the code is K bits.
Now save the Data and the code.

It increases required Storage space to M+K bits

When we read M-Bit word of Data, the Code is recalculated and Compared with the code that was saved.

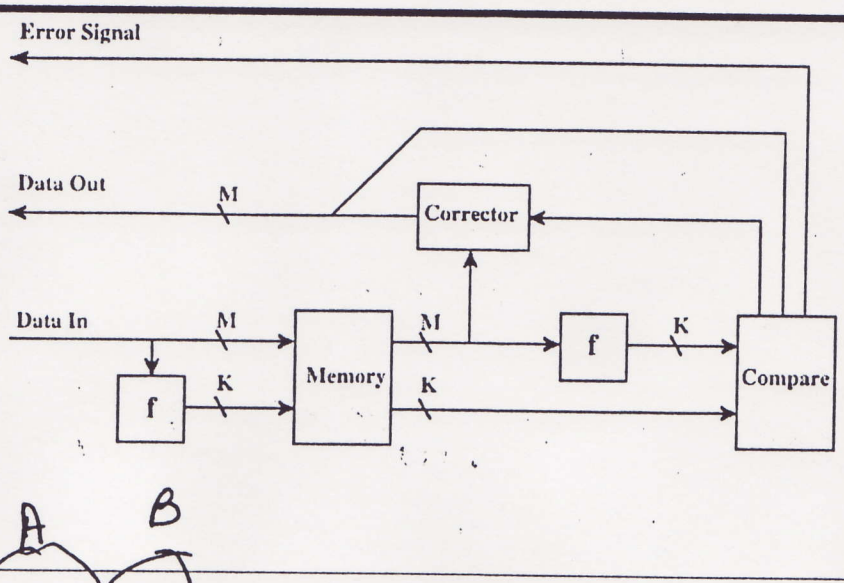
Possible results:
No errors are detected.
Error is detected .. the error can be corrected.
Error is detected .. can't be corrected.

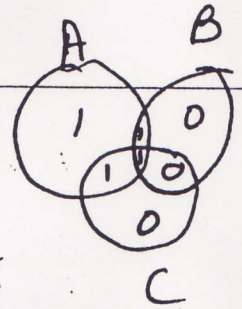
Simplest error Correcting Code is the Hamming Code.
3 Venn diagram with 3 circles.

4 Bit word.
Send 1110

7 compartments.
Place the 4 bits into inner compartments.
Remaining compartments are for parity bits.
Parity is chosen that total number of 1s are even.
Look — A only has 3 ones - so add one. Put 0
B has 2 — no need to add. Put 1
C has 2 — no need to add. Put 0

Error Correcting Code Function





parity

bits needed $2^K - 1 \geq 3 + K$ $2^3 - 1$

This data A bit requires a Code word of 3 bits + parity bit

only single bit errors can be corrected.

If an error changes one of the data bits. It is easily detected see this example

Not OK A is now even ← But has a 1 parity

B is OK

C is now odd but has no parity bit it is zero

So the problem should have occurred where it is Cannot A∩C

Change that and try it!

We can use Hamming algorithm to construct error detecting codes for any memory word.

K parity bits are added to m-bit word.

New word is then m+k bits.

We can mix them data & parity. Numbered from 1 to 21 not 0 to 20

All bits that are powers of 2 are parity bits, rest are data bits.

for a 16 bit word, 5 parity bits are added → 1, 2, 4, 8, 16

The rest are data bits.

Each parity bit checks specific positions.

Parity Bit 1 checks bits 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21
2 checks 2, 3, 6, 7, 10, 11, 14, 15, 18, 19.
4 4, 5, 6, 7, 12, 13, 14, 15, 20, 21
8 8, 9, 10, 11, 12, 13, 14, 15
16 16, 17, 18, 19, 20, 21.

Bit 5 is checked by bits 4 & 1 because 4 + 1 = 5
Bit 6 is checked by bits 4 & 2
Bit 15 is checked by bits 8, 4, 2 & 1 and so on.

Let us by figure out the code word for a 16 bit number
data is 1111000010101110
Mark parity bits by Box. Then Put the data

□	□	1	□	1	1	1	□	0	0	0	0	1	0	1	□	0	1	1	1	0
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21

Now Place the parity after counting one for each parity bit checks.
Example Parity bit 1 checks 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21
There are 5 is ones. So put a zero.

0	0	0	0	1
1	2	4	8	16

Suppose an error was introduced and the new Code word is

0	0	1	0	0	1	1	0	0	0	0	0	1	0	1	1	0	1	1	1	0
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21

Parity bit 1 Contains five 1s. So parity is wrong. ✓
2 Contains six 1's. Correct.
4 Contains five 1's incorrect
8 Contains two 1's Correct
16 Contains four 1's Correct

Error must be in one bit Common to 1 and 4 parity bit lists.
5, 7, 13, 15, 21
Now take out the bits common in other lists. 7, 15 are common with p.2 list.
Leaves 5, 13, 21