Hamming (7, 4) Code

Pranay Khosla (2K19/EP/073)
Sakshi Arora (2K19/EP/084)

EC-262 Communication Systems
MTE Project

01

ERRORS IN COMMUNICATION

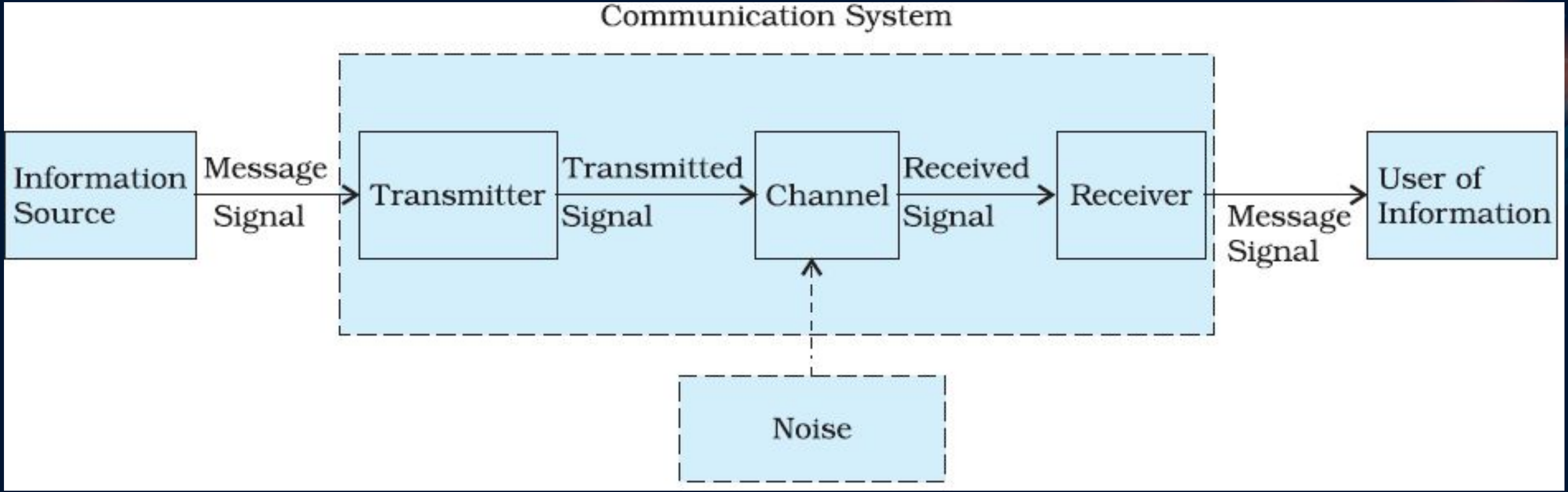
02

ERROR CORRECTION & DETECTION

03

IMPLEMENTATION OF HAMMING (7, 4) CODE

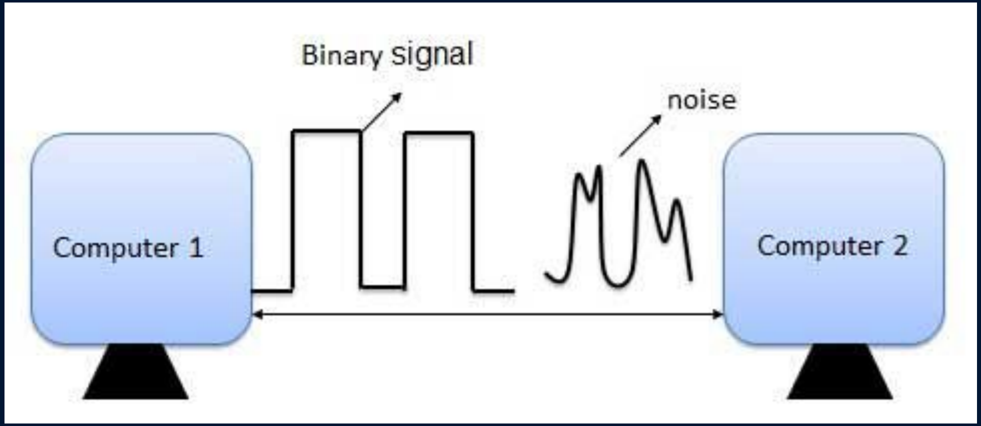
COMMUNICATION SYSTEM BLOCK DIAGRAM



Source: Class 12 NCERT Physics Chapter 15, CBSE

ERROR IN COMMUNICATION

- Error:- a condition when the output and input information do not match.
- During transmission, noise may introduce errors in the binary bits while travelling; a 0 bit may change to 1 or a 1 bit may change to 0.



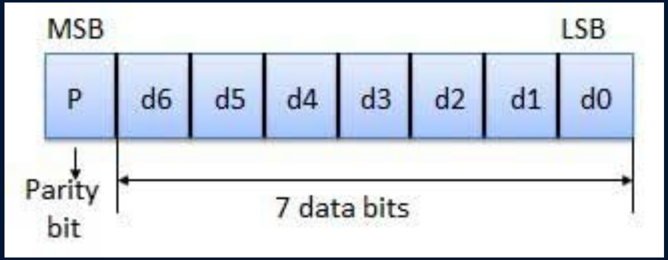
ERROR DETECTION

To avoid this error, we use error-detecting codes:- additional data added to the digital message to detect if an error occurred during transmission.

Parity generation:- one of the most widely used error detection techniques. Noise may change 0 bit to 1 bit or 1 bit to 0 bit.

A Parity Bit is added to the word containing data to make number of 1s either even or odd. The message containing both data bits and parity bit is transmitted. At the receiving end, the number of 1s is counted and if it does not match with the transmitted one, there is an error.

Using this technique, an even number of errors might go undetected.



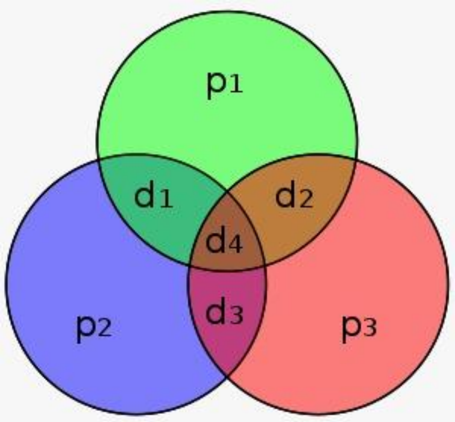
HAMMING CODE

- Hamming codes are a family of linear error-correcting codes.
- **Goal:** to create a set of parity bits that overlap so that a single-bit error in a data bit or a parity bit can be detected and corrected.
- Can detect up to two-bit errors or correct one-bit errors without detection of uncorrected errors.
- Are perfect codes: achieve highest possible code rate.
- With m parity bits, bits from 1 up to $2^m - 1$ can be covered. After discounting the parity bits, $2^m - m - 1$ bits remain for use as data. Code rate = $(2^m - m - 1)/(2^m - 1)$
- As m varies, we get some possible Hamming codes:
    - Hamming (7, 4) [m=3]
    - Hamming (15, 11) [m=4]

THE HAMMING (7, 4) CODE

Hamming (7, 4) codes encode 4 bits of data into 7 bit blocks. The extra 3 bits are the redundant parity bits we talked about. For 4-bit data: d1, d2, d3, d4, parity bits are:

$$p_1 = d_2 + d_3 + d_4$$
$$p_2 = d_1 + d_3 + d_4$$
$$p_3 = d_1 + d_2 + d_4$$



Source: Hamming code, Wikipedia

THE HAMMING (7, 4) CODE

So, the Hamming code becomes: [p1, p2, p3, d1, d2, d3, d4], at the receiver end, parity bits are again constructed, and checked with the received bits to detect, and correct errors. Decoding the message, we get:

No error	->	[0 0 0]
Error in p1	->	[1 0 0]
Error in p2	->	[0 1 0]
Error in p3	->	[0 0 1]
Error in d1	->	[0 1 1]
Error in d2	->	[1 0 1]
Error in d3	->	[1 1 0]
Error in d4	->	[1 1 1]

Now that we know where the error is, decoding is just flipping the error bit, and removing the parity bits.

WORKING OF OUR CODE

CS — octave-gui — 100×24

~/Desktop/Projects/CS — octave-gui

```
octave:19> hamming

 Enter message                : Winter is Coming

 Generating Hamming Codes

 Introducing Error in a random bit

 Message with error           : Winter as Coming

 Correcting error

 Data with error correction : Winter is Coming


octave:20>
```

CONCLUSION

Errors creep in due to noise in channel

Errors in Communication System

Used to detect and correct errors

Hamming (7, 4) Code

Used Octave to implement Hamming (7, 4) Code to detect and correct error in a string

Octave implementation

THANK YOU