

# Circuit Simulation Project

**Name of the participant:** Bharani J

**Title of the circuit:** Design and Simulation of Hamming Code (7,4) Error Detection and Correction Circuit Using Logic Gates

## Theory/Description:

Hamming Code is a method used for error detection and correction in digital communication and data storage. It adds redundant parity bits to the original data to detect and correct single bit errors.

The Hamming (7,4) code takes 4 data bits and adds 3 parity bits, forming a 7-bit code word.

- Data bits: D1, D2, D3, D4
- Parity bits: P1, P2, P3

## Parity Bit Generation (Even Parity):

The parity bits are generated using XOR operations as follows:

- P1 is generated using bits 3, 5, and 7
- P2 is generated using bits 3, 6, and 7
- P3 is generated using bits 5, 6, and 7

## Error Detection and Correction:

At the receiver side, parity bits are recalculated and compared with the received parity bits to generate syndrome bits:

- S1 checks positions 1, 3, 5, 7
- S2 checks positions 2, 3, 6, 7
- S3 checks positions 4, 5, 6, 7

The syndrome value (S3 S2 S1) represents the binary index of the erroneous bit position. If the syndrome is non-zero, the corresponding bit is inverted to correct the error automatically.

## Circuit Diagram:

The circuit diagram is designed using XOR gates and a 3-to-8-line decoder (also designed using logic gates). The error bit will be inverted using XOR gates at the output end. Hence it produces the corrected code.

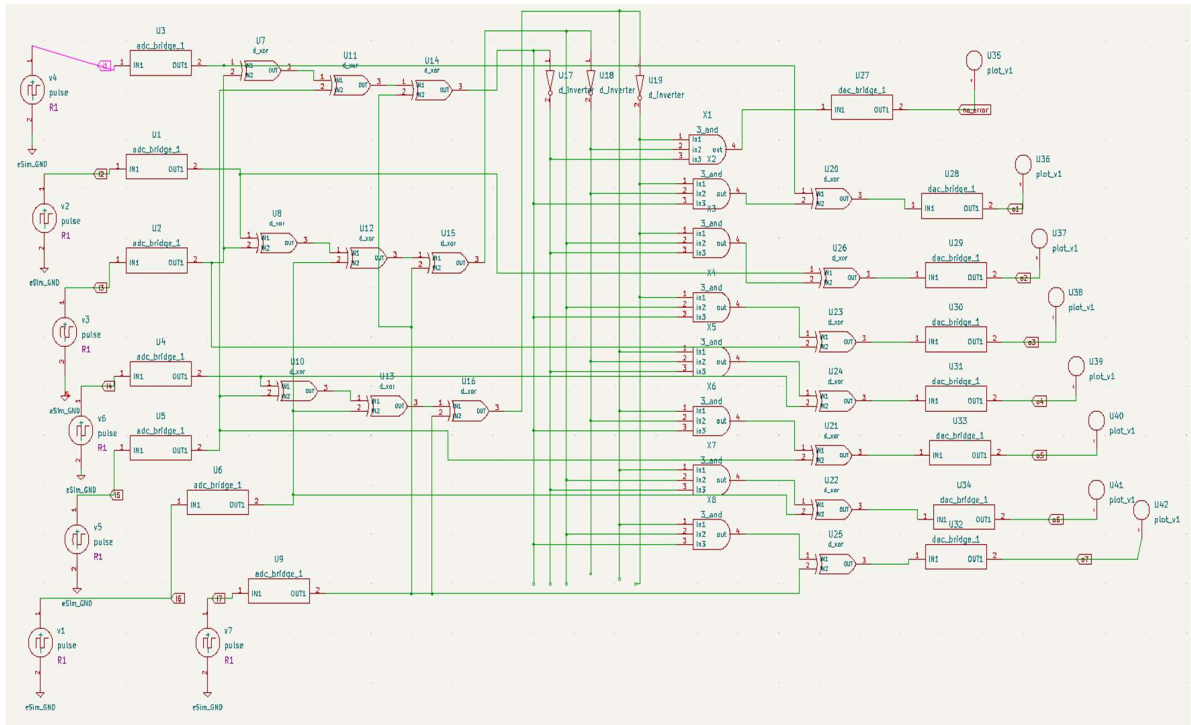


Figure 1: eSim schematic

### Results (Input, Output waveforms and/or Multimeter readings):

Suppose we have 4-bit data- D1 D2 D3 D4 = 1 0 0 1

Hamming (7,4) bit positions:

Position	1	2	3	4	5	6	7
Type	P1	P2	D1	P3	D2	D3	D4

Parity bits are calculated as XOR of specific positions:

- P1 = XOR of bits 3, 5, 7 = 0
- P2 = XOR of bits 3, 6, 7 = 0
- P3 = XOR of bits 5, 6, 7 = 1

Thus, the code sent would be

Position	1	2	3	4	5	6	7
Bit	0	0	1	1	0	0	1

### Case 1: Single-Bit Error

If we introduce an error at 2<sup>nd</sup> bit-

Sending data with error- 0 1 1 1 0 0 1 (2<sup>nd</sup> bit flipped)

Syndrome Calculation-

- S1 = XOR of bits 1,3,5,7 = 0
- S2 = XOR of bits 2,3,6,7 = 1
- S3 = XOR of bits 4,5,6,7 = 0

Syndrome =  $S_3 S_2 S_1 = 0\ 1\ 0 = 2^{\text{nd}}$  bit. The syndrome correctly indicates that the error is at bit 2.

Error Correction: Flip the 2nd bit back:

$0\ 1\ 1\ 1\ 0\ 0\ 1 \Rightarrow 0\ 0\ 1\ 1\ 0\ 0\ 1$

- Output is corrected
- no\_error signal = LOW

INPUT:

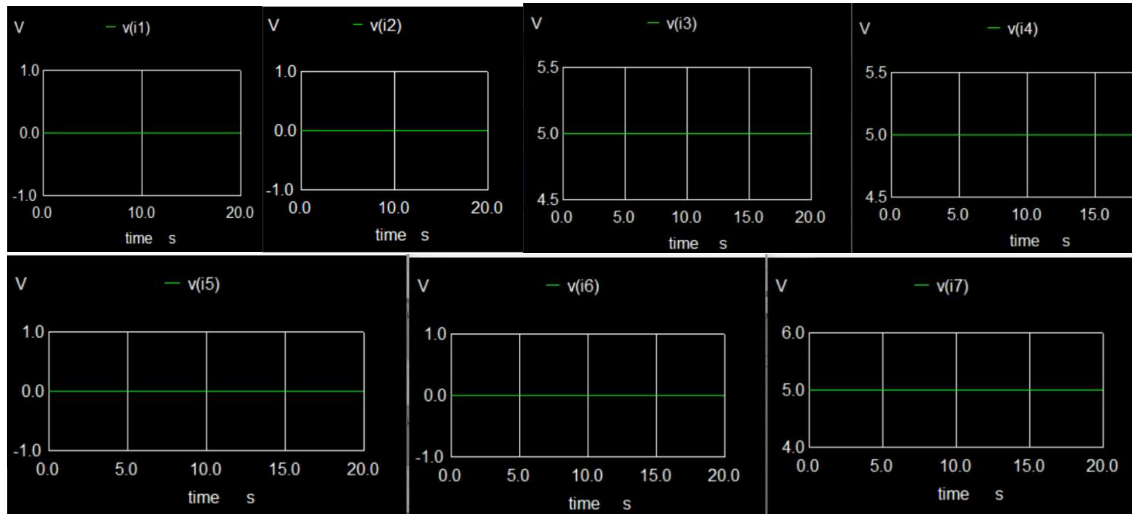


Figure 2: ngspice input waveforms

OUTPUT:

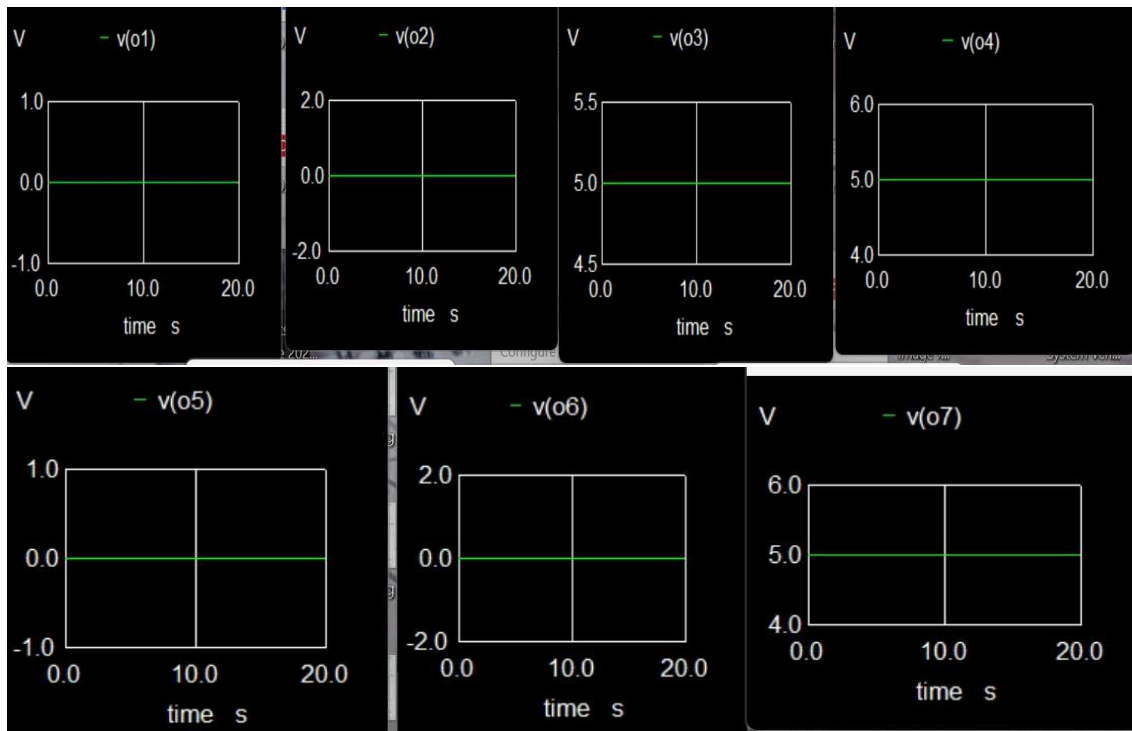


Figure 3: ngspice output waveforms

no\_error=0 (active-low to detect error):

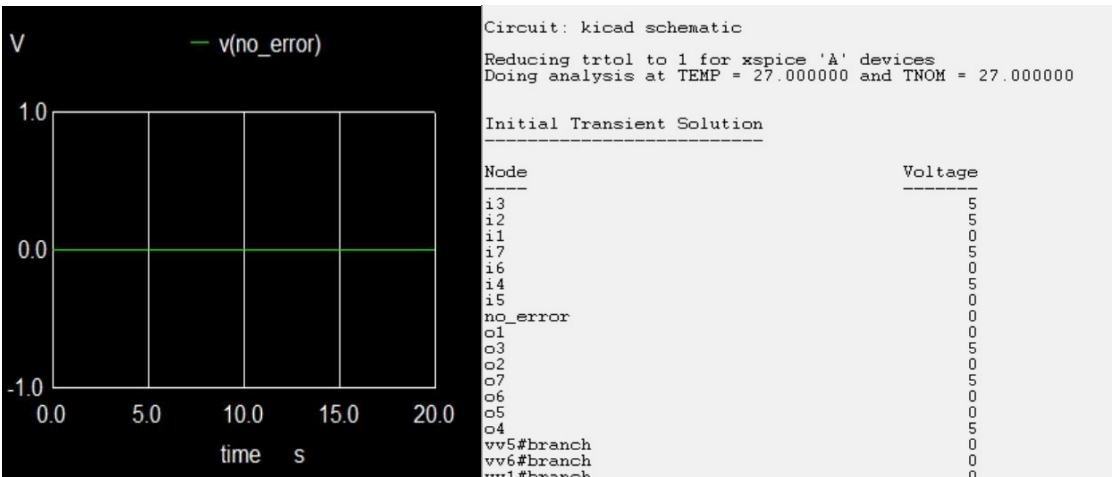


Figure 4&5: ngspice no\_error waveform and ngspice output window

Case 2: No Error

Input data **0011001** was applied without any error.

The output remained **0011001**, and the **no\_error** signal stayed **HIGH**, indicating error-free transmission.

INPUT:

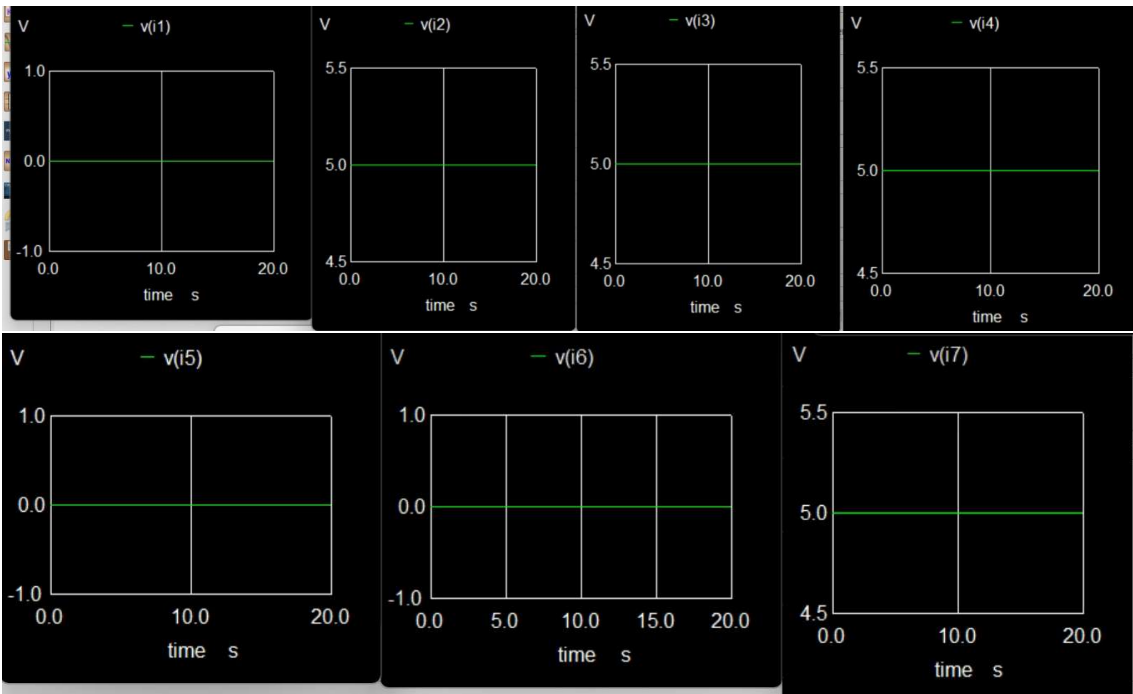


Figure 6: ngspice input waveforms

OUTPUT:

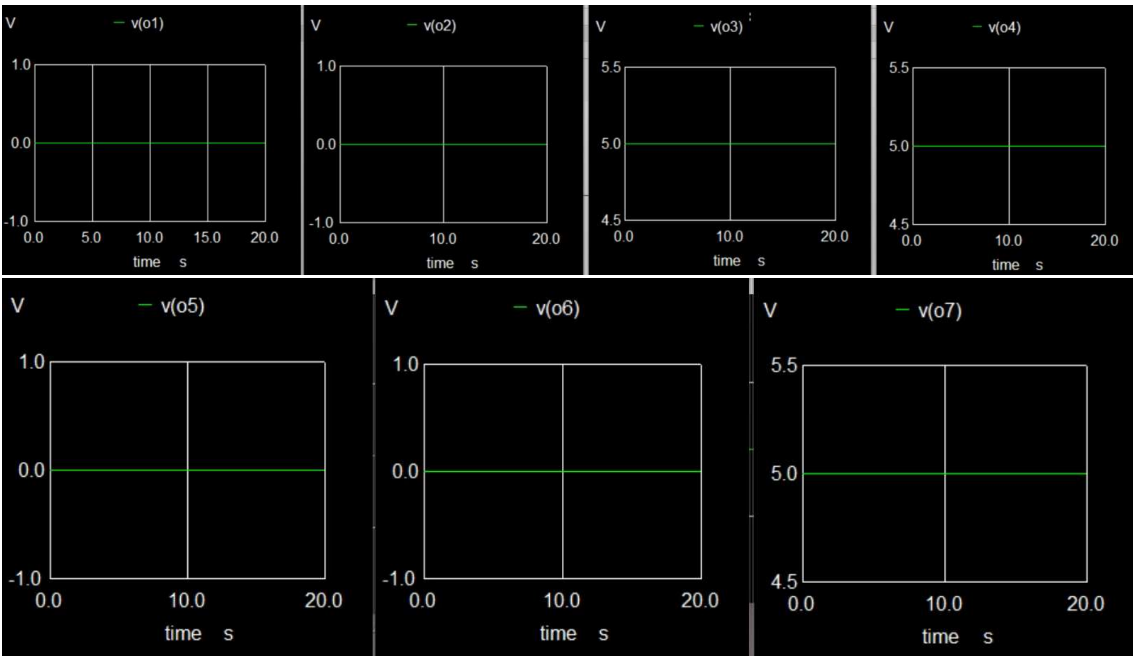


Figure 7: ngspice output waveforms

no\_error=1(active low to indicate error):

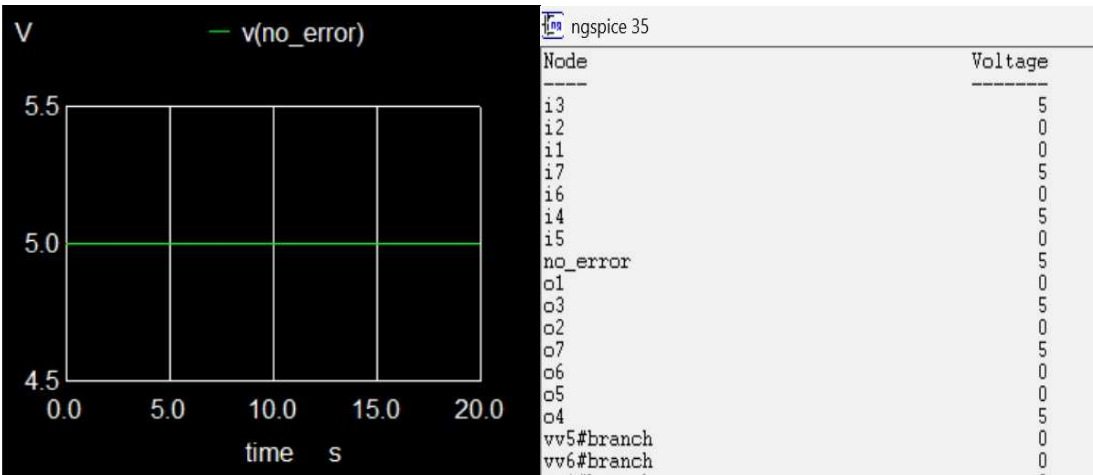


Figure 8&9: ngspice no\_error waveform and ngspice output window

The Hamming (7,4) circuit successfully detects and corrects single-bit errors in transmitted data. Simulation results show that the erroneous bit is automatically corrected, and the no\_error signal accurately indicates error-free or error-present conditions. The circuit works reliably for both error-free and erroneous input data, confirming correct implementation of Hamming code logic.

**Source/Reference(s):**

- M. Morris Mano, "Digital Design," 6th Edition, Pearson, 2017.
- R.W. Hamming, "Error Detecting and Error Correcting Codes," Bell System Technical Journal, vol. 29, pp. 147–160, 1950.