International Journal of Recent Technology and Engineering (IJRTE)
ISSN: 2277-3878 (Online), Volume-8, Issue-1, May 2019

Error Detection and Correction using Hamming Code

A. Viswesh Iyer, Shantanu Pandey, Aniket Nikhil Chafekar

Abstract: ***Hamming code is used for single bit error detection and correction. Unlike ARQ protocols and simple parity code which can detect only odd number of bits in error, hamming codes can correct the data without retransmission of the whole data which saves a lot of time and reduces the probability of getting an error again. To do so, it uses 3 redundancy bits for a 4-bit data, 4 redundancy bits for 7-bit data and so on. These redundancy bits are placed at bit positions 1, 2, 4, 8… and so on with the original data. On the receiving end, if any error is detected, it is corrected by negating the bit and the redundancy bits are removed from the data. Hamming codes are considered as perfect codes for single bit errors and is generally used for 7-bit data transmission. In the proposed simulation done using Proteus software and Atmel studio, redundancy bits will be added to a 4-bit data to detect and correct single bit error. In the transmitter side, the encoded data that is the original 4-bit data along with redundancy bits are displayed on LCD screen. When the data is received by the receiver, error is detected and corrected by using the redundancy bits. The bit on which error has occurred along with the received data is displayed on another LCD display connected to the receiver. The corrected data is shown by making the LEDs glow.***

Index Term: Hamming Code, Error Detection, Error Correction, Redundancy Bits.

I. INTRODUCTION

Error coding is a method of detecting and correcting these errors to ensure that information is transferred intact from its source to its destination. Error coding is used for fault tolerant computing in magnetic and optical data storage media, computer memory satellite and deep space communications, network communications, cellular telephone networks, and almost any other form of digital data communication. Error coding incorporates formulas to encode data bits at the source into longer bit words for transmission. The "codeword" can then be decoded at the destination to retrieve the information. The extra bits in the code word provide redundancy that, according to the coding scheme used, will allow the destination to use the decoding process to determine if the communication medium introduced errors and in some cases correct them so that the data need not be retransmitted.

In older days, parity bit was used to detect an error in transmission. It is a single bit added to original data to make the number of 1's in the data even or odd. The occurrence of bits whose value is 1 is counted. For even parity, if the count is odd, then the parity bit is set and in case of odd parity, if the count is even, the parity bit is set. Error detection with parity bit gives correct result only when error occurs at odd no. of bits. Hence, if error occurs at even no. of bits, then no error will be detected. When an error is detected, the code cannot determine the exact location of error due to which whole data needs to be sent again by the transmitter from scratch. In a noisy environment the transmission can take a longer time.

Another error-control scheme, ARQ (Automatic Repeat Request) Protocol was introduced which uses acknowledgment signal and timeouts to achieve reliable transmissions. There are 3 types of ARQ protocols i.e. Stop-and-wait ARQ, Go-Back-N ARQ, and Selective Repeat ARQ. All these protocols determine which packet needs to be retransmitted by the transmitter. But, neither of these protocols gives the exact location of the error due to which whole frame needs to be transmitted again. In a faulty circuit or noisy environment, it may take a longer period of time to successfully transmit a frame of data since every time an error is detected; the data needs to be transmitted again.

Hamming code was first introduced by Richard Hamming code is implemented as set of error-correction codes that can be used to detect and correct the single bit error that can occur when a data is transmitted. Hamming code is capable of detecting errors even when 2 bits are corrupted but may not show the exact position. It uses redundancy bits which are transmitted along with the original data to detect and correct the error. Number of redundancy bits depends on the size of data to be transmitted. For e.g. a 4-bit data uses 3 redundancy bits, a 7-bit data uses 4 redundancy bits, and so on.

It is used for detecting and correcting a single bit error or can be used to detect 2-bit error. Suppose there are 'k' data bits, 'm' redundancy bits and n total number of bit, then

n=2m - 1 and

k=2m - m - 1

Substituting different values of m, we get

3 redundancy bits for 4 bit data

4 redundancy bits for 11 bit data

5 redundancy bits for 26 bit data

6 redundancy bits for 57 bit data

We have proposed an algorithm of hamming code for 4-bit data with 3 redundancy bits. The idea is implemented using Atmel Studio and Proteus software.

Revised Manuscript Received on 30 May 2019.

***** Correspondence Author

A. Viswesh Iyer*, Student, Department of EEE, SRM Institute of Science and Technology, Chennai, India.

Shantanu Pandey, Student, Department of EEE, SRM Institute of Science and Technology, Chennai, India.

Aniket Nikhil Chafekar, Student, Department of EEE, SRM Institute of Science and Technology, Chennai, India.

Atmel Studio is used to compile the code to obtain a hex file of the hamming code. Proteus software is used to create schematics and to design automation. The hex file generated by Atmel Studio for both transmitter and receiver is uploaded to both the microcontrollers. A random 4-bit data along with 3 redundancy bits (calculated by the microcontroller) is transmitted. The encoded data is displayed on an LCD. The receiver receives the data and determines the location of error (if any) and displays corrupted bit, received data on another LCD and corrected data on LEDs.

II. COMPONENTS REQUIREMENT

There are several components used in the simulation which are listed below:

- Atmega16 microcontroller
- LCD
- LED
- Switches
- NOT gate

1. *ATmega16*

Atmega16 is an 8-bit microcontroller of AVR family with low power consumption. It is based on RISC (Reduced Instruction Set Computing) architecture with 131 powerful instructions. Atmega16 can work on a maximum frequency of 16MHz. It is a 40 pin microcontroller having 32 I/O lines which are divided into four 8-bit ports designated as PORTA, PORTB, PORTC and PORTD.

2. *LCD*

Two 16*2 LCDs are used for simulation. A 16x2 LCD means it can display 16 characters per line and there are 2 such lines. In this LCD each character is displayed in 5x7 pixel matrix. This LCD has two registers, namely, Command and Data. The command register stores the command instructions given to the LCD. The data register stores the data to be displayed on the LCD. The data is the ASCII value of the character to be displayed on the LCD.

3. *LED*

A light-emitting diode (LED) is a semiconductor light source that emits light when current flows through it. There are two terminals i.e. anode and cathode. Anode is grounded and cathode is given logic 1 to turn it on.

4. *Switch*

Two types of switches i.e. SPST (Single Pole Single Throw) and SPDT (Single Pole Double Throw) are used in the simulation. SPST switch is used to send a new data and SPDT is used to send an error.

5. *NOT gate*

A NOT gate gives a negated output for a single bit input. This is used in simulation to instigate an error in the transmission.

III. WORKING

For the simulation of hamming code, Atmel Studio and Proteus software is used. Atmel Studio is used to compile the code to obtain a hex file of the hamming code. Proteus software is used to create schematics and to design automation. Atmega16 is an AVR microcontroller, a compact integrated circuit designed to govern specific operation. In the simulation shown, two microcontrollers are used one of which acts as transmitter and the other acts as receiver. On the transmitter side an LCD is connected to PORT A of the transmitter to display the encoded data. A switch is connected to 8th bit of PORT B of the transmitter to send another 4-bit data. The 4-bit data, along with 3 redundancy bits are transferred in parallel to the receiver from PORT B. In real time transmission, errors in communication are caused due to noise or electrical distortion, circuit failures, echo, attenuation, harmonic distortion, etc. But, these errors are hard to implement in the simulation. Hence, NOT gate along with SPDT (Single Pole Double Throw) switch is connected across the data line connecting the transmitter and receiver. The user can select when to give error. On the receiving end, the receiver receives the 4-bit data along with the 3 redundancy bits on PORT D. Three variables are declared which are set or reset based on the redundancy bits. The values of these variables determine the position at which the error has occurred. An LCD is connected to PORT A of the receiver which displays the position of error, if any, and the received data. Numbers of LEDs are connected to PORT B of the receiver to show the data after correction.

The proposed algorithm for transmitter is given below in the form of flowchart as:

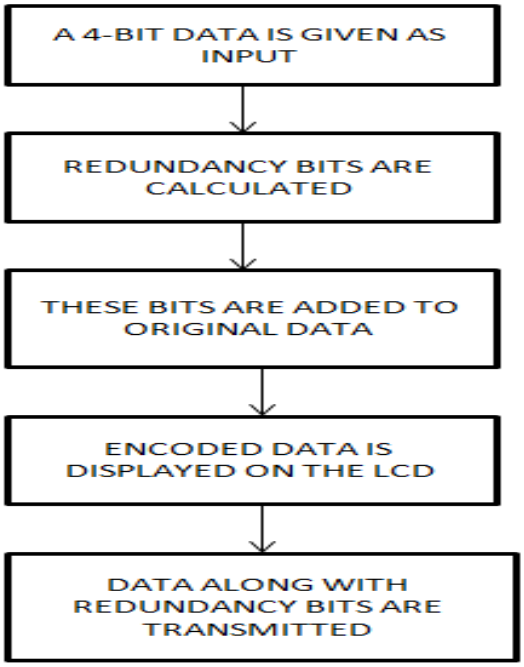

Fig. 1: Proposed Flow Chart for Transmitter

The proposed algorithm for Receiver is given below in the form of flowchart as:

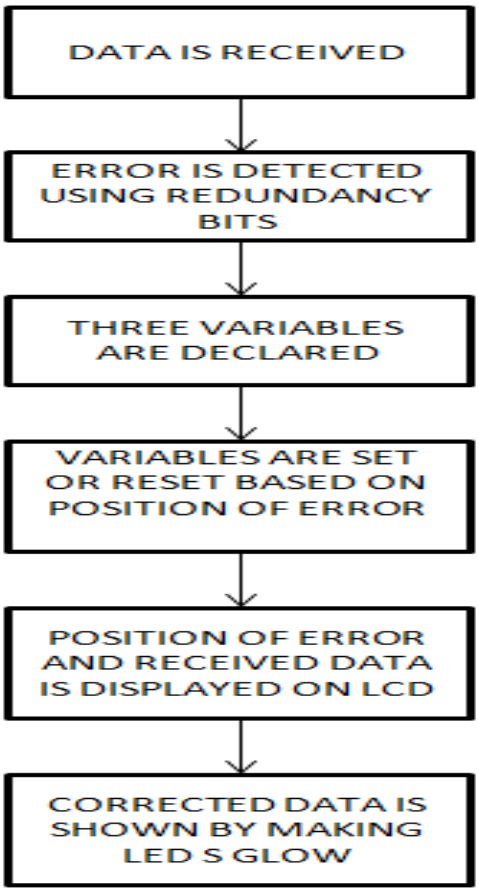

Fig. 2: Proposed Flow Chart for Receiver

IV. RESULT

Using Proteus software and Atmel Studio, following results were obtained after the simulation.

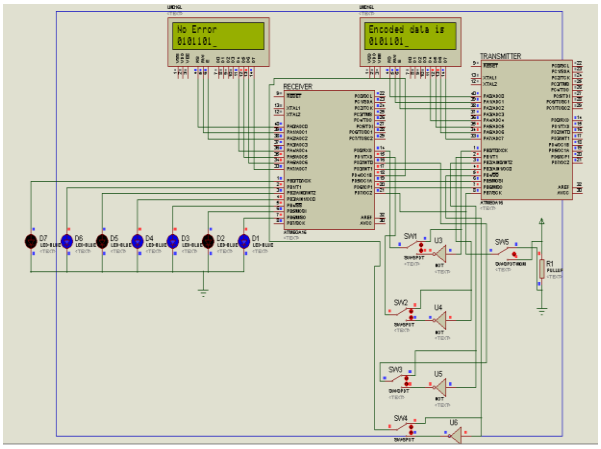

Fig. 3: All switched connected to data lines

When all the switches i.e. SW1, SW2, SW3 and SW4 are open no error was given as a result of which a message showing “No Error” and the received data is displayed on the LCD. Also the LEDs glow in similar pattern to that of input.

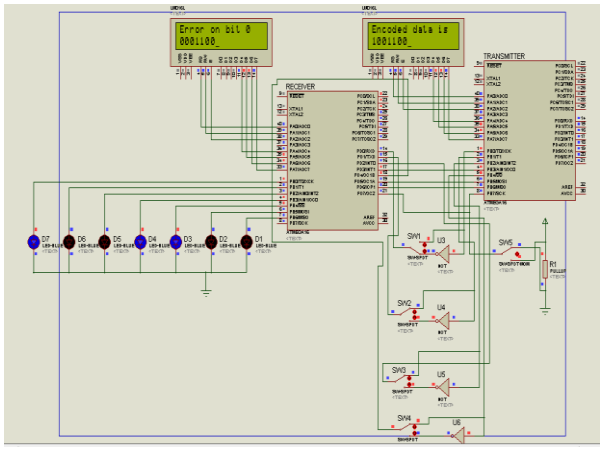

Fig. 4: SW1 connected to NOT gate

When SW1 is connected to the NOT gate, an error is instigated on 0th bit of the data. The microcontroller detects the error position and displays it on the LCD along with the received data. The corrected data is shown by making LEDs glow in similar pattern.

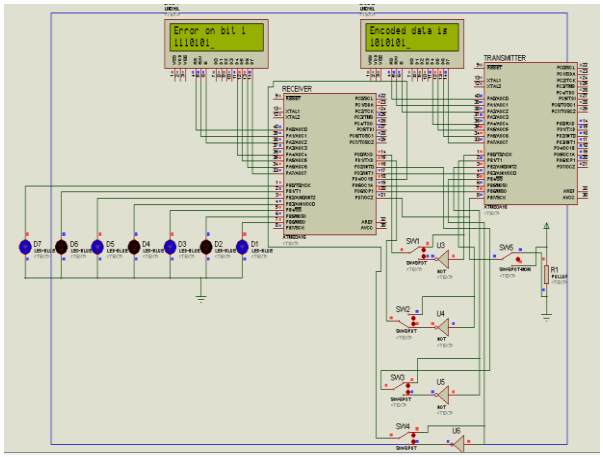

Fig. 5: SW2 connected to NOT gate

When SW2 is connected to the NOT gate, an error is instigated on 1st bit of the data. The microcontroller detects the error position and displays it on the LCD along with the received data. The corrected data is shown by making LEDs glow in similar pattern.

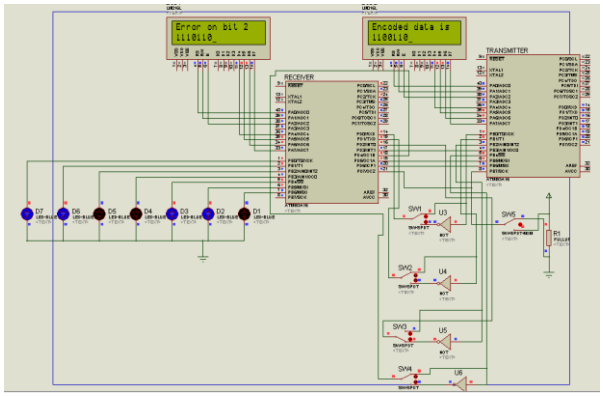

Fig. 6: SW3 connected to NOT gate

When SW3 is connected to the NOT gate, an error is instigated on 2nd bit of the data. The microcontroller detects the error position and displays it on the LCD along with the received data. The corrected data is shown by making LEDs glow in similar pattern.

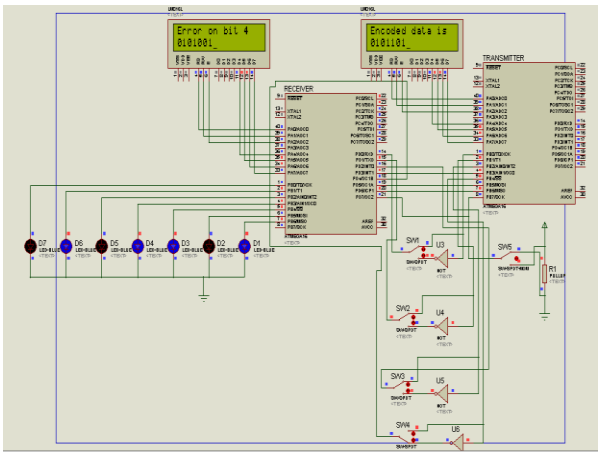

Fig. 7: SW4 connected to NOT gate

When SW4 is connected to the NOT gate, an error is instigated on 3rd bit of the data. The microcontroller detects the error position and displays it on the LCD along with the received data. The corrected data is shown by making LEDs glow in similar pattern.

The switch SW5 is a momentary switch which is used to send a new data. In each case mentioned above, SW5 was pressed every time to send new data.

Table i
Results obtained for different data

4-BIT DATA	ENCODED DATA	SWITCHES				REMARKS
		SW1	SW2	SW3	SW4	
0101	0101101	0	0	0	0	NO ERROR
1001	1001100	1	0	0	0	ERROR ON BIT 0
1011	1010101	0	1	0	0	ERROR ON BIT 1
1101	1100110	0	0	1	0	ERROR ON BIT 2
0101	0101101	0	0	0	1	ERROR ON BIT 4

V.CONCLUSION

Hamming codes corrected the data without retransmission of the whole data which saved a lot of time and reduced the probability of getting an error again.At the receiving side, if an error was detected, it was corrected by negating that bit and the redundancy bits were removed from the data. Therefore, using hamming codes for error detection and correction is considered as an optimum technique for single bit errors and is generally used for 7-bit data transmission.

In the proposed simulation, redundancy bits were added to a 4-bit data to detect and correct single bit error. At the transmitter side, the encoded data that is the original 4-bit data along with its redundancy bits got displayed on LCD screen. When that data was received by the receiver, error was detected and corrected. The bit on which error has occurred along with the received data was displayed on another LCD display connected to the receiver. The corrected data is shown by making the LEDs glow.

At the receiving side, the receiver received the 4-bit data along with its 3 redundancy bits. Three variables are declared which are set or reset based on the redundancy bits. The values of these variables determined the position at which the error has occurred. Numbers of LEDs are connected to another port of the microcontroller to show the data after correction.

REFERENCES

1. Anil Kumar Singh, “Error detection and correction by hamming code”, 2016 International Conference on Global Trends in Signal Processing, Information Computing and Communication.
2. Shivani Tambatkar, Siddharth Narayana Menon ,V. Sudarshan, M. Vinodhini, N. S. Murty “Error detection and correction in semiconductor memories using 3-d parity check code with hamming code”, 2017 International Conference on Communication and Signal Processing .
3. John Metzger, “Correction of Two (or Often More) Vector Symbol Errors with the Outer Structure of a Hamming Single Error Correcting Code”.
4. Manish Gupta; JaskaranS. Bhullar ; Bharat Naresh Bansal, “Undetected error probability of hamming code for any number of symbols”.
5. Minghui Yang; Jin Li ; Keqin Feng ; Dongdai Lin, “Generalized Hamming Weights of Irreducible Cyclic Codes”.