BIMASATI (Bulletin of Multi-Disciplinary Science and Applied Technology)
ISSN : 2809-6096 (Media Online)
Vol 1, No 5, Agustus 2022
Hal : 159-163

Pengecekan Bit *Error* Pada Media Transmisi Pengiriman Gambar Menggunakan Metode *Hamming Code*

Deli Alvinda[*], Achmad Fauzi, Husnul Khair

Program Studi Teknik Informatika, STMIK Kaputama, Binjai, Indonesia
Email: [1,*]alvindadeli@gmail.com, [2]fauzyrivai88@gmail.com, [3]khairhusnul@yahoo.co.id
Email Penulis Korespodensi: alvindadeli@gmail.com

Abstrak-Keberhasilan penyampaian informasi dari pengirim (transmitter) ke penerima (receiver) merupakan salah satu hal yang sangat penting dalam menentukan keandalan sebuah sistem komunikasi. Dalam dunia komunikasi baik komunikasi yang menggunakan kabel atau yang menggunakan udara sebagai media transmisi pasti akan mengalami gangguan-gangguan dalam proses komunikasi yang disebabkan oleh gangguan bernama noise. Noise merupakan sinyal listrik yang tidak diinginkan. Tambahan sinyal yang tidak diinginkan ini dalam suatu proses komunikasi ini merupakan faktor pembatas utama dalam sistem komunikasi data. Dalam proses komunikasi data, kemungkinan kesalahan data yang diterima dapat terjadi karena data menjadi mengalami error dan harus dikirim ulang. Hamming code merupakan suatu metode pendeteksi error yang mampu mendeteksi beberapa error, namun hanya mampu mengoreksi satu error (single error correction). Metode pendeteksi error ini sangat cocok digunakan pada situasi dimana terdapat beberapa error yang teracak (randomly occuring errors).

Kata Kunci: Error; Hamming Code; Noise; Pengirim; Penerima.

Abstract-The successful delivery of information from the sender (transmitter) to the receiver (receiver) is one thing that is very important in determining the reliability of a communication system. In the world of communication, both communication using cables or using air as a transmission medium will inevitably experience disturbances in the communication process caused by a disturbance called noise. Noise is an unwanted electrical signal. The addition of this unwanted signal in a communication process is a major limiting factor in data communication systems. In the process of data communication, the possibility of data errors received can occur because the data becomes an error and must be re-sent. Hamming code is an error detection method that is able to detect several errors, but is only able to correct one error (single error correction). This error detection method is very suitable for use in situations where there are several random errors.

Keywords: Error; Hamming Code; Noise; Receiver; Transmitter.

1. PENDAHULUAN

Keberhasilan penyampaian informasi dari pengirim (*transmitter*) ke penerima (*receiver*) merupakan salah satu hal yang sangat penting dalam menentukan keandalan sebuah sistem komunikasi. Keandalan sebuah sistem komunikasi data bukan hanya diukur dari kecepatan transfer data atau yang disebut dengan bit *rate* dalam satuan bps (*bit per second*) tetapi juga keberhasilan sampainya data yang dikirim oleh pengirim pada penerima dengan jelas dan benar.

Dalam dunia komunikasi baik komunikasi yang menggunakan kabel atau yang menggunakan udara sebagai media transmisi pasti akan mengalami gangguan-gangguan dalam proses komunikasi. Gangguan ini biasanya disebut dengan noise atau derau. *Noise* merupakan sinyal listrik yang tidak diinginkan. Tambahan sinyal yang tidak diinginkan ini dalam suatu proses komunikasi ini merupakan faktor pembatas utama dalam sistem komunikasi data. Bila *noise* terjadi dalam suatu sistem komunikasi maka sistem komunikasi akan mengalami gangguan. Gangguan yang terjadi dapat menyebabkan proses komunikasi terganggu atau bahkan dapat memutuskan proses komunikasi. Dalam proses komunikasi data, kemungkinan kesalahan data yang diterima oleh penerima (*receiver*) dapat terjadi, sehingga sering sekali data tersebut harus dikirim ulang oleh pengirim (*transmitter*) kepenerima sampai data tersebut diterima dengan benar sesuai dengan data yang dikirim oleh pengirim, sehingga waktu penyampaian data secara keseluruhan mengalami keterlambatan.

Dalam ilmu komputer, terdapat bermacam–macam logika untuk mendeteksi dan mengoreksi error tersebut. Salah satu cara untuk mendeteksi *error* yang sederhana adalah dengan menggunakan *Hamming Code*. *Hamming Code* adalah suatu metode pendeteksi *error* yang mampu mendeteksi beberapa error, namun hanya mampu mengoreksi satu error (*single error correction*). Metode pendeteksi error ini sangat cocok digunakan pada situasi dimana terdapat beberapa error yang teracak (*randomly occuring errors*). Agar proses pengiriman data berlangsung dengan cepat, maka pada penerima harus dapat mendeteksi dan mengoreksi data yang salah tersebut sehingga tidak dibutuhkan transfer ulang oleh pengirim terhadap data yang salah diterima oleh penerima. Keuntungan yang didapatkan untuk mendeteksi *error* dengan metode *Hamming Code* adalah cara kerjanya yang cukup sederhana dan tidak membutuhkan alokasi memori yang banyak.

Kode *Hamming* adalah metode deteksi kesalahan yang dapat mendeteksi beberapa kesalahan, tetapi hanya mampu koreksi kesalahan tunggal. Metode deteksi kesalahan ini cocok untuk digunakan dalam situasi di mana ada beberapa kesalahan acak. Kesalahan mengakibatkan perubahan pada konten data yang ditransfer. Ada berbagai logika untuk mendeteksi dan memperbaiki kesalahan. Salah satu cara untuk mendeteksi kesalahan yang hanya dengan menggunakan *Hamming Code*, teknik ini adalah yang paling nyaman untuk menemukan kesalahan dalam transmisi data bit [1].

Hamming code merupakan salah satu contoh teknik *error control coding* yang dapat mendeteksi dan mengkoreksi *error*. Pada penelitian ini dianalisa metode untuk mendeteksi dan mengkoreksi multi bit *error* pada pesan yang dikirimkan menggunakan partition *hamming code.* Sebelum data dikirim ke penerima, pengirim membuat pola partition bit pesan yang lebih kecil dari *hamming code* (7,4) dan menambahkan bit parity dalam semua blok pesan yang dipecah sehingga

menjadi sebuah codeword baru. *Partition* digunakan agar penerima mudah dalam mendeteksi dan mengkoreksi multi bit *error* [2].

Metode *Hamming Code* yang diterapkan pada Arduino dan komunikasi UART. Metode *Hamming Code* adalah metode yang menambahan beberapa *parity* bit tambahan pada bit data dengan menggunakan logika XOR. *Parity* bit tambahan yang dihasilkan dari proses logika XOR didapatkan dari jumlah data yang dimasukkan dalam prosesnya. Berdasarkan hasil pengujian, metode *Hamming Code* dapat melakukan proses encode dan decode data, serta dapat melakukan deteksi dan koreksi error pada data yang mengalami *error* dalam proses pengujian. Rata-rata *delay* yang didapatkan berjumlah 102,7ms untuk data 5 bit dan 109,5ms untuk data 4 bit pada proses *encode*. Serta 17,5 ms untuk data 10 bit dan 100,1ms untuk data 11 bit pada prose *decode* [3].

Hamming code merupakan metode yang dapat mendeteksi sekaligus melakukan perbaikan kesalahan pada *parity* bit data dengan cara melakukan pengecekan terhadap struktur bit data yang bersangkutan. *Metode hamming code* ini diimplementasi ke dalam sebuah bentuk simulasi, sehingga menghasilkan simulasi cara kerja *hamming code* yang menunjukkan bagaimana metode ini melakukan bit *error correction* atau perbaikan kesalahan pada bit data digital. Simulasi ini dirancang menggunakan Bahasa pemrograman Microsoft Visual Studio 2010, dengan bentuk tampilan langkah demi langkah proses pengecekan kesalahan dan perbaikan kesalahan berdasarkan cara kerja metode *hamming code* yang digunakan [2].

2. METODOLOGI PENELITIAN

2.1 *Hamming Code*

Kode *Hamming* digunakan untuk mendeteksi *error* dan perbaikan kode pesan terkirim, kode koreksi *error* adalah sebuah algoritma untuk mendeteksi adanya kesalahan dalam pesan yang dikirimkan sekaligus memperbaiki pesan tersebut sehingga pesan dapat tersampaikan dengan benar melalui sistem transmisi data melalui sistem jaringann berbasis pada isi pesan itu sendiri. Sedang *Error* dapat terjadi yang disebabkan oleh berbagai sebab, sebuah bit dalam pesan mungkin ditambah, terhapus atau berubah. Kode koreksi *Error* banyak diaplikasikan pada CD *player*, *high speed modem*, dan *cellular phone*. Deteksi *error* lebih sederhana dibanding perbaikan sebuah *error*. Sebagai contoh pengujian digit sering kali dijumpai secara *embedded* pada sejumlah *credit card* dengan tujuan mendeteksi kesalahan. Berikut merupakan sebuah contoh bagaimana mendeteksi dan memperbaiki kesalahan pada pesan yang dikirimkan:

Aturan main :

1. Data asli yang akan dikirimkan dinyatakan dalam variabel Di dan check bit dengan (Ci).
2. Posisi biner diawali dari bit 1, posisi check bit Ci pada 2^n,yaitu 1, 2, 4, 8 dan 16.
3. Penentuan check bit dilakukan melalui EXOR untuk semua bit data.

Untuk penentuan kode *hamming* dari 4 bit data, maka terdapat D1, D2, D3 dan D4 dan untuk *check* bit 2^n didapat C0, C1 dan C2. Penentuan posisi bit dapat dilihat pada tabel berikut:

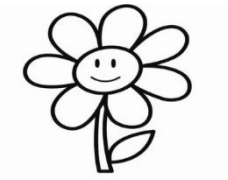
Tabel 1. Penentuan Posisi Bit

Ci = (2^n)	**2^6**	**2^5**	**2^4**	**2^3**	**2^2**	**2^1**	**2^0**
Posisi bit	1	1	1	1	0	0	0
	1	1	0	0	1	1	0
	1	0	1	0	1	0	1
Kode	D4	D3	D2	C2	D1	C1	C0

C0 = D1 xor D2 xor D4
C1 = D1 xor D3 xor D4
C2 = D2 xor D3 xor D4

3. HASIL DAN PEMBAHASAN

Pada analisa metode ini agar dapat menghasilkan data yang akurat. Pengujian ini menggunakan sebuah gambar. Kemudian gambar tersebut di konversi menjadi bilangan biner yang akan digunakan pada proses.



Gambar 1. Sample Gambar

Hasil konversi gambar yaitu mendapatkan angka biner *input* 10111010. Panjang *input* dan *output* data = 8 *bit* = 2^3 sehingga jumlah *parity check bit* : 3 + 1 = 4 dan panjang data *input* dan *output* dari algoritma *Hamming Code* = 12 bit. Hasil perhitungan panjang bit pada program dibawah ini.

ISSN : 2809-6096 (Media Online)
Vol 1, No 5, Agustus 2022
Hal : 159-163

Tabel 2. Bit Input dan Output Pada Program

POSISI BIT	1	2	3	4	5	6	7	8	9	10	11	12

Menandai semua posisi *bit* yang merupakan posisi dari *parity check bit*. Posisi selain posisi *parity check bit* merupakan posisi dari *data bit*.

Tabel 3. Posisi *Parity Check Bit* Dan Data Bit

Bit *Position*	Member *Position*	*Parity Check* Bit	Data Bit
12	1100		D_8
11	1011		D_7
10	1010		D_6
9	1001		D_5
8	1000	P_4	
7	0111		D_4
6	0110		D_3
5	0101		D_2
4	0100	P_3	
3	0011		D_1
2	0010	P_2	
1	0001	P_1	

Apabila proses tersebut dimasukkan kedalam aplikasi, akan muncul posisi *check* bit seperti tabel dibawah ini.

Tabel 4. Posisi *Parity Check Bit* Dan Data Bit Pada Program

POSISI BIT	1	2	3	4	5	6	7	8	9	10	11	12
KATEGORI	P1	P2	D1	P3	D2	D3	D4	P4	D5	D6	D7	D8
BIT INFORMASI			1		0	1	1		1	0	1	0

Kemudian menghitung nilai dari *parity check bit* P_1, P_2, P_3 dan P_4. Hasilnya sebagai berikut:

$P_1 = D_1 \oplus D_2 \oplus D_4 \oplus D_5 \oplus D_7 = 1 \oplus 0 \oplus 1 \oplus 1 \oplus 1 = 0$

$P_2 = D_1 \oplus D_3 \oplus D_4 \oplus D_6 \oplus D_7 = 1 \oplus 1 \oplus 1 \oplus 0 \oplus 1 = 0$

$P_3 = D_2 \oplus D_3 \oplus D_4 \oplus D_8 = 0 \oplus 1 \oplus 1 \oplus 0 = 0$

$P_4 = D_5 \oplus D_6 \oplus D_7 \oplus D_8 = 1 \oplus 0 \oplus 1 \oplus 0 = 0$

Nilai *parity check bit* untuk data *input*,

0 0 0 0

P4 P3 P2 P1

Apabila menghitung nilai dari *parity check bit* P_1, P_2, P_3 dan P_4 dimasukkan kedalam aplikasi maka tampilannya dapat dilihat pada tabel berikut:

Tabel 5. Hitung Nilai Dari *Parity Check Bit* Pada Program

POSISI BIT	1	2	3	4	5	6	7	8	9	10	11	12	XOR
KATEGORI	P1	P2	D1	P3	D2	D3	D4	P4	D5	D6	D7	D8	
BIT INFORMASI			1		0	1	1		1	0	1	0	
P1			✓		✓		✓		✓		✓		$1 \oplus 0 \oplus 1 \oplus 1 \oplus 1 = 0$
P2			✓			✓	✓			✓	✓		$1 \oplus 1 \oplus 1 \oplus 0 \oplus 1 = 0$
P3					✓	✓	✓					✓	$0 \oplus 1 \oplus 1 \oplus 0 = 0$
P4									✓	✓	✓	✓	$1 \oplus 0 \oplus 1 \oplus 0 = 0$

Setelah diproses maka data *input* menjadi:

0	1	0	1	**0**	1	1	0	**0**	1	**0**
	0									
D8	D7	D6	D5	**P4**	D4	D3	D2	**P3**	D1	**P2**
	P1									

Setelah diproses dalam sistem sebagai berikut:

Bit data yang akan dikirimkan adalah 010101100100

3.1 *Pengoreksi* Error

Data yang telah diproses yaitu 001001101010, akan tetapi *receiver* menerima data 011001111010. Kemudian dilakukan pengecekan *error* pada aplikasi dengan menginputkan data 011001111010.
Pengecekan *error* dilakukan operasi *XOR* terhadap *parity check bit input* (0000) dan *parity check bit output* (1010).

P4 P3 P2 P1	*Input*	0	0	0	0
	Output	1	0	1	0
		----------			⊕
		1	0	1	0

Kemudian mengkonversikan hasil operasi *XOR* ke dalam bentuk bilangan desimal,yakni sebagai berikut :
$(1010)_2 = 1 \times 2^3 + 0 \times 2^2 + 1 \times 2^1 + 0 \times 2^0 = 8 + 0 + 2 + 0 = 10$.

Pada proses manual, posisi bit *error* terdapat pada Bit *Position* 10 seperti tabel dibawah ini.

Tabel 6. Posisi *Error Bit*

Bit *Position*	Member *Position*	*Parity Check* Bit	Data Bit
12	1100		D_8
11	1011		D_7
10	**1010**		**D_6**
9	1001		D_5
8	1000	P_4	
7	0111		D_4
6	0110		D_3
5	0101		D_2
4	0100	P_3	
3	0011		D_1
2	0010	P_2	
1	0001	P_1	

Kemudian implementasi dari pengecekan *error* secara manual di dalam programnya seperti tabel dibawah ini.

Tabel 7. Hitung Nilai Dari *Parity Check Bit* Pada Program

POSISI BIT	12	11	10	9	8	7	6	5	4	3	2	1	XOR
KATEGORI	D8	D7	D6	D5	P8	D4	D3	D2	P4	D1	P2	P1	
BIT DATA	0	1	1	0	0	1	1	1	1	0	1	0	
P1		✓		✓		✓		✓		✓		✓	0 ⊕ 1 ⊕ 1 ⊕ 0 ⊕ 1 = 1
P2		✓	✓			✓	✓			✓	✓		0 ⊕ 1 ⊕ 1 ⊕ 1 ⊕ 1 = 0
P3	✓					✓	✓	✓	✓				1 ⊕ 1 ⊕ 1 ⊕ 0 = 1
P4	✓	✓	✓	✓									0 ⊕ 1 ⊕ 1 ⊕ 0 = 0

4. KESIMPULAN

Berdasarkan hasil penelitian yang telah dilakukan, maka kesimpulan yang dapat diambil yakni Sistem dapat membantu untuk mendeteksi error bit dalam proses transmisi data. Sistem dapat membantu untuk meminimalisir data yang error saat diterima oleh penerima data. Sistem dapat membantu untuk mengefisiensi waktu penerima karena data error yang diterima dapat dideteksi oleh sistem.

REFERENCES

[1] Achmad Fauzi, R. (2017). *Bit Error Detection and Correction With Hamming Code Algorithm*.
[2] Muhajir, F., Efendi, S., & Sutarman, &. (2016). DETEKSI DAN KOREKSI MULTI BIT ERROR DENGAN PARTITION HAMMING CODE. In *Jurnal Teknovasi* (Vol. 03, Issue 2).
[3] Fajar Andana, A., Rizqika Akbar, S., & Maulana, R. (2018). *Implementasi Deteksi Dan Koreksi Error Pada Komunikasi Serial Arduino Berbasis UART Dengan Metode Hamming Code* (Vol. 2, Issue 11).
[4] Firman, A., Wowor, H. F., Najoan, X., Teknik, J., Fakultas, E., & Unsrat, T. (2016). *Sistem Informasi Perpustakaan Online Berbasis Web*. *5*(2).
[5] Lavarino, D., & Yustanti, Wi. (2016). Rancang Bangun E - Voting Berbasis Website Di Universitas Negeri Surabaya. *Jurnal Manajemen Informatika*, *18*(2), 22280.
[6] Rasim, Setiawan, W., & Rahman, eka fitrajaya. (2008). Metodologi Pembelajaran Berbasis Komputer Dalam Upaya

Menciptakan Kultur Pembelajaran Berbasis Teknologi Informasi dan Komunikasi. *Pendidikan Teknologi Informasi Dan Komunikasi*, *1*, 1–17.

[7] Wiliani, N., & Syadid, Z. (2017). *Rancang Bangun Aplikasi Kasir Tiket Nonton Bola Bareng Pada X Kasir Di Suatu Lokasi X Dengan Visual Basic 2010 Dan MySql*. *6*(2), 77–83.

[8] Masri, M., Alam, H., Masri, M., Lubis, Z., & Izhni Pohan, M. (2019). SIMULASI BYTE ERROR CORRECTION DENGAN MENGGUNAKAN HAMMING CODE. In *Journal of Electrical Technology* (Vol. 4, Issue 3).

[9] Susilawati, H., & Sunardi, D. (2017). *MENGGUNAKAN METODE HAMMING CODE PADA PROSES PEGIRIMAN DATA BERBASIS ANDROID* (Vol. 8, Issue 1).

[10] Parinduri, I., & Hutagalung, S. N. (2018). *Seminar Nasional Royal (SENAR) 2018 ISSN 2622-9986 (cetak) STMIK Royal-AMIK Royal*, *hlm. 215-218 ISSN 2622-6510 (online) Kisaran*, *Asahan*.

[11] Fauzi, A., & Putri Rahayu, R. (2021). IMPLEMENTATION OF THE HAMMING CODE METHOD IN BIT DATA IMPROVEMENT TRANSMISSION PROCESS. *JIK)*, *5*(1).

[12] Sembiring, M., & Haris Simbolon, F. (2021). *Perancangan Perangkat Lunak Pembelajaran Algoritma Hamming Code dalam Mencari Bit Error pada Komunikasi Data*. *1*(1), 2798–9593.