Penerapan Aljabar Boolean dalam Koreksi Kesalahan pada Sistem Komunikasi Digital dengan Penggabungan Kode BCH dan Kode Reed-Solomon

Zahira Dina Amalia - 13522085[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13522085@std.stei.itb.ac.id

Meskipun kita menikmati kemudahan pengiriman informasi digital saat ini, di baliknya terdapat teknologi yang memungkinkan proses tersebut. Algoritma digunakan untuk enkripsi dan dekripsi setiap informasi yang dikirim, memastikan pemulihan data meskipun mengalami kerusakan selama pengiriman. Salah satu algoritma yang memainkan peran krusial dalam mendeteksi dan memperbaiki kesalahan satu bit pada informasi adalah Kode Hamming. Melalui penerapan konsep Aljabar Boolean, proses ini dapat diotomasi menggunakan ekspresi Boolean dan rangkaian logika. Namun, dalam konteks yang lebih luas, penerapan Aljabar Boolean melibatkan penggabungan kode BCH dan kode Reed-Solomon untuk meningkatkan efisiensi dan keandalan koreksi kesalahan pada Sistem Komunikasi Digital.

***Kata kunci*—Aljabar Boolean, Kode BCH, Kode Reed-Solomon, Koreksi Sistem Komunikasi Digital**

I. Pendahuluan

Kesalahan komunikasi dapat terjadi saat pengiriman atau penyimpanan data, seperti pembalikan bit (1 menjadi 0 atau sebaliknya). Kesalahan bisa terjadi karena kesalahan bit (rusaknya satu atau beberapa bit) dan/atau kesalahan paket (rusaknya seluruh paket data). Kesalahan bit biasanya disebabkan oleh noise atau interferensi, sedangkan kesalahan paket dapat terjadi akibat degradasi atau interferensi sinyal yang ekstrim. Kode koreksi kesalahan digunakan untuk mendeteksi dan mengidentifikasi kesalahan, juga memastikan keandalan pengiriman data. Penerapan kode koreksi kesalahan menjadi penting untuk menjaga integritas data.

Teknik kontrol kesalahan pada komunikasi biasa disebut dengan *Forward Error Correction* (FEC) yang menambahkan redudansi (kode koreksi kesalahan) ke data sehingga penerima dapat mendeteksi dan mengoreksi kesalahan tanpa perlu meminta transmisi ulang. Terdapat beberapa macam teknik FEC yang umum digunakan di antaranya kode Turbo, kode LDPC (*Low-Density Parity-Check*), kode Konvolusi, Kode Hamming, kode Reed-Solomon, dan kode BCH (Bose-Chaidhuri-Hocquenghem). Teknik-teknik FEC tersebut menerapkan konsep-konsep Aljabar Boolean dalam implementasinya.

Berdasarkan penjelasan di atas, pertanyaan penelitian yang diajukan adalah bagaimana penerapan konsep Aljabar Boolean dapat digunakan dalam proses Koreksi Kesalahan pada Sistem Komunikasi Digital. Untuk menjawab pertanyaan ini, penulis melakukan eksplorasi terhadap dua topik utama, yaitu matematika diskrit dengan fokus pada Aljabar Boolean, serta topik terkait komunikasi digital. Dalam penelitian ini, penulis berusaha menjawab pertanyaan tersebut melalui studi literatur dan eksplorasi terhadap sistem komunikasi digital dengan mengintegrasikan dua jenis kode koreksi kesalahan, yaitu kode BCH dan kode Reed-Solomon. Penekanan dilakukan pada bagaimana konsep-konsep Aljabar Boolean dapat diterapkan untuk meningkatkan efisiensi dan kehandalan proses koreksi kesalahan dalam konteks komunikasi digital.

II. Landasan Teori

A. Aljabar Boolean

Aljabar Boolean, ditemukan oleh George Boole pada tahun 1854, merupakan suatu konsep matematika yang fokus pada manipulasi logika biner dan operasi variabel biner. George Boole mengamati kemiripan sifat antara himpunan dan logika proposisi, menciptakan aturan-aturan dasar logika yang membentuk struktur matematika yang dikenal sebagai aljabar Boolean. Konsep ini, dengan representasi nilai true dan false sebagai 0 dan 1, memiliki aplikasi penting dalam perancangan rangkaian pensaklaran, rangkaian digital, dan integrasi sirkuit komputer (IC), menyediakan dasar formal untuk perkembangan teknologi elektronik modern. Boole, melalui karyanya yang terkenal, "The Laws of Thought," memaparkan aturan-aturan dasar logika yang membentuk dasar struktur matematika yang dikenal sebagai aljabar Boolean.

1) Aksioma, Operasi Dasar, dan Hukum-Hukum pada Aljabar Boolean

   Aljabar Boolean, dinotasikan sebagai tupel,
   <B, +, ., ', 0, 1>
   Elemen 0 dan 1 merepresentasikan nilai yang berbeda. Untuk setiap a, b, dan c dalam B, aljabar Boolean mematuhi aksioma-aksioma berikut:

1. Identitas
   i. a + 0 = a
   ii. a · 1 = a
2. Komutatif
   i. a + b = b + a
   ii. a · b = b · a
3. Distributif
   i. a · (b + c) = (a · b) + (a · c)
   ii. a + (b · c) = (a + b) · (a + c)
4. Komplemen
   Untuk setiap a ∈ B terdapat elemen unik a‘∈ B sehingga
   i. a + a’ = 1
   ii. a · a’ = 0

Berikut merupakan operasi-operasi dasar pada aljabar Boolean yakni,

i. 1 + 1 = 1
ii. 1 + 0 = 1
iii. 0 + 0 = 0
iv. 1 · 1 = 1
v. 1 · 0 = 0
vi. 0 · 0 = 0

Hukum-Hukum yang terdapat pada aljabar Boolean, ialah,

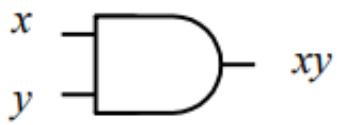
1. Hukum Identitas
   i. a + 0 = a
   ii. a · 1 = a
2. Hukum Indempoten
   i. a + a = a
   ii. a · a = a
3. Hukum Komplemen
   i. a + a’ = 1
   ii. aa’ = 0
4. Hukum Dominansi
   i. a · 0 = 0
   ii. a + 1 = 1
5. Hukum Involusi
   i. (a’)’ = a
6. Hukum Penyerapan
   i. a + ab = a
   ii. a (a+b) = a
7. Hukum Komutatif
   i. a + b = b + a
   ii. a · b = b · a
8. Hukum Asosiatif
   i. a + (b + c) = (a + b) + c
   ii. a · (b · c) = (a · b) · c
9. Hukum Distributif
   i. a · (b + c) = a · b + a · c
   ii. a + (b · c) = (a + b) · (a + c)
10. Hukum De Morgan
    i. (a + b)’ = a’b’
    ii. (ab)’ = a’ + b’
11. Hukum 0/1
    i. 0’ = 1
    ii. 1’ = 0

2) Rangkaian Logika

Pada aljabar Boolean terdapat fungsi boolean yang mengambil satu atau lebih variabel biner (yang bernilai 0 atau 1). Menggunakan aksioma, operas-operasi dasar dan hukum-hukum pada aljabar Boolean yang telah dijabarkan sebelumnya, fungsi Boolean dapat direpresentasika dalam bentuk rangkaian logika. Pada rangkaian logika terdapat tiga gerbang dasar, yaitu gerbang AND, gerbang OR, dan gerbang NOT. Dari tiga gerbang dasar ini terdapat gerbang logika turunan, yakni gerbang NAND, gerbang NOR, gerbang XOR, dan gerbang XNOR. Berikut ilustrasi dan penjelasan ringkas mengenai gerbang-gerbang logika tersebut.

a. Gerbang AND

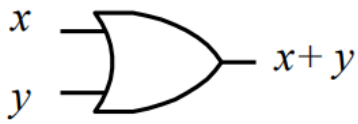
Gerbang AND menghasilkan keluaran 1 hanya jika kedua inputnya adalah 1. Representasi matematisnya adalah x · y, di mana x dan y adalah masukan dan simbol ” · “ menunjukkan operasi perkalian.



Gambar 2.1 Gerbang AND
(Sumber:https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/11-Aljabar-Boolean-(2023)bagian1.pdf)

b. Gerbang OR

Gerbang OR menghasilkan keluaran 1 jika salah satu atau kedua inputnya adalah 1. Representasi matematisnya adalah x + y, di mana x dan y adalah masukan, dan simbol + menunjukkan operasi penjumlahan.



Gambar 2.2 Gerbang OR
(Sumber:https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/11-Aljabar-Boolean-(2023)bagian1.pdf)

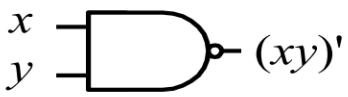
c. Gerbang NOT

Gerbang NOT menghasilkan keluaran yang merupakan kebalikan dari inputnya. Representasi matematisnya adalah x atau x′, di mana x adalah masukan.

Gambar 2.3 Gerbang NOT
(Sumber:https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/11-Aljabar-Boolean-(2023)bagian1.pdf)

d. Gerbang NAND

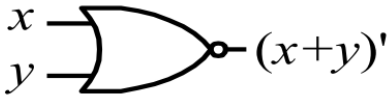
Gerbang NAND menghasilkan keluaran 0 hanya jika kedua inputnya adalah 1. Representasi matematisnya adalah (x·y)', di mana x dan y adalah masukan.



Gambar 2.4 Gerbang NAND
(Sumber:https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/11-Aljabar-Boolean-(2023)bagian1.pdf)

e. Gerbang NOR

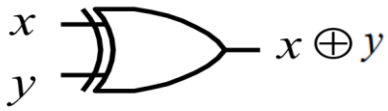
Gerbang NOR menghasilkan keluaran 1 jika kedua inputnya adalah 0. Representasi matematisnya adalah (x + y)', di mana x dan y adalah masukan.



Gambar 2.5 Gerbang NOR
(Sumber:https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/11-Aljabar-Boolean-(2023)bagian1.pdf)

f. Gerbang XOR

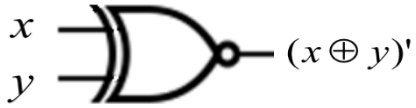
Gerbang XOR menghasilkan keluaran 1 jika masukannya berbeda (satu 0 dan satu 1), dan keluaran 0 jika inputnya sama. Representasi matematisnya adalah x ⊕ y di mana x dan y adalah masukan.



Gambar 2.6 Gerbang XOR
(Sumber:https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/11-Aljabar-Boolean-(2023)bagian1.pdf)

g. Gerbang XNOR

Gerbang XNOR menghasilkan keluaran 1 jika kedua masukannya sama (dua 0 atau dua 1), dan keluaran 0 jika inputnya berbeda. Representasi matematisnya adalah (x ⊕ y)' di mana x dan y adalah masukan.



Gambar 2.7 Gerbang XNOR
(Sumber:https://informatika.stei.itb.ac.id/~rinaldi.munir/Matdis/2023-2024/11-Aljabar-Boolean-(2023)bagian1.pdf)

Dapat disimpulkan dari penjelasan tiap gerbang tersebut untuk masukan x dan y sebagai berikut akan menghasilkan hasil sebagaimana pada kedua tabel berikut.

x	y	x'	y'	x·y	x+y	(x·y)'	(x+y)'	x⊕y	(x⊕y)'
0	0	1	1	0	0	1	1	0	1
0	1	1	0	0	1	1	0	1	0
1	0	0	1	0	1	1	0	1	0
1	1	0	0	1	1	0	0	0	1

Tabel 2.1 Hasil dar Setiap Gerbang Logika

B. Koreksi Kesalahan pada Sistem Komunikasi Digital

Koreksi kesalahan pada sistem komunikasi digital menjadi aspek kritis untuk memastikan integritas dan akurasi informasi yang dikirim melalui saluran komunikasi yang rentan terhadap gangguan atau kesalahan. Berbagai landasan teori umum mendukung upaya koreksi kesalahan ini, termasuk konsep-konsep seperti kode kesalahan. Terdapat dua macam kode yang biasa digunakan pada zaman ini yaitu kode Blok dan kode Konvolusional.

Kode Blok adalah jenis kode dalam teori kesalahan dan komunikasi yang membagi urutan informasi menjadi blok pesan dengan panjang tertentu. Setiap blok pesan diubah menjadi kata kode yang sesuai, yang dapat digunakan untuk deteksi dan koreksi kesalahan. Kode blok memiliki sifat tanpa memori, di mana setiap blok pesan dienkripsi secara independen. Salah satu contoh kode blok adalah Reed-Solomon codes, yang sering digunakan dalam penyimpanan data dan transmisi data digital. Contoh kode blok adalah Reed-Solomon codes, yang digunakan pada CD, DVD, dan QR Code. Dalam Reed-Solomon codes, setiap blok pesan diubah menjadi kata kode dengan bit redundan untuk deteksi dan koreksi kesalahan.

Kode Konvolusional adalah jenis kode kesalahan yang melibatkan penggunaan filter konvolusi untuk menghasilkan urutan keluaran berdasarkan urutan masukan. Dalam konteks komunikasi digital, encoder kode konvolusional menerapkan operasi konvolusi pada bit-bit masukan untuk menghasilkan bit keluaran. Kode konvolusional umumnya bersifat memiliki memori, yang berarti keluaran pada suatu waktu tidak hanya bergantung pada bit masukan saat ini tetapi juga pada bit-bit sebelumnya. Kode konvolusional sering digunakan dalam komunikasi nirkabel dan sistem komunikasi yang rentan terhadap kesalahan. Contoh kode konvolusional adalah Kode Konvolusional 1/2, di mana setiap bit keluaran bergantung pada dua bit masukan sebelumnya melalui operasi konvolusi. Kode konvolusional sering digunakan dalam komunikasi nirkabel seperti GSM untuk meningkatkan kehandalan transmisi data.

Makalah ini berfokus pada penerapan dua kode yang merupakan jenis kode blok dalam koreksi kesalahan pada sistem komunikasi digital, yakni kode BCH (Bose-Chaudhuri-Hocquenghem) dan kode Reed-Solomon.

C. Kode BCH

Pada tahun 1959, Hocquenghem mengusulkan metode penyandian siklis dengan tingkat perbaikan kesalahan tinggi. Pada tahun 1960, Bose dan Chaudhury memperbaiki metode ini menjadi Bose Chaudhury Hocquenghem atau BCH, sebuah kelompok kode siklis efektif untuk koreksi kesalahan acak, yang menggeneralisasi fungsi Hamming codes untuk koreksi kesalahan ganda. Metode pengkodean BCH, dikembangkan oleh Bose dan Ray-Chaudhuri pada tahun 1960, secara terpisah mengalami invensi yang mendalam oleh Hocquenghem pada 1959. Metode ini awalnya diterapkan untuk beberapa nilai m pada kode biner dengan panjang 2^m - 1, dan kemudian dikembangkan lebih modern oleh Gorenstein dan Zierler pada tahun 1961 dengan menggunakan simbol dari *Galois Field* (GF). Dalam pembangunannya, operasi logika biner, seperti AND, OR, dan NOT, digunakan untuk membentuk polinomial pembangkit karakteristik. Polinomial ini, yang mencerminkan struktur siklis kode, dibangun dengan melibatkan operasi logika biner pada lapangan Galois.

1) Proses Enkode Kode BCH

Prosesnya dimulai dengan membentuk *Galois Field* (GF). Lapangan Galois atau yang biasa disebut dengan *Galois Field* (GF) adalah suatu area dari dua elemen biner yang operasi penjumlahan dan perkaliannya didefinisikan pada modulo-2 dan dinyatakan sebagai GF (2^m). Panjang kata sandi (n) pada BCH dihitung dengan,

$$n = 2^m - 1$$

Lalu, kemampuan perbaikan kesalahan (t) dihitung dari panjang kata sandi (n), bit informasi (k), dan m pada GF (2^m) dengan rumus,

$$t \geq \frac{n-k}{m}$$

Tahap berikutnya adalah membentuk polinomial generator. Sebelum itu, perlu diketahui terdapat polinomial primitif yakni p(x) untuk t pada BCH. Polinomial ini diberntuk dari polinomial minimal untuk GF (2^m) yang tidak dapat difaktorkan lagi. Berikut polinomial primitif untuk GF (2^m) dari m = 2 sampai m = 21.

m	p(x)	m	p(x)
2	$x^2 + x + 1$	12	$x^{12} + x^6 + x^4 + x + 1$
3	$x^3 + x + 1$	13	$x^{13} + x^4 + x^3 + x + 1$
4	$x^4 + x + 1$	14	$x^{14} + x^{10} + x^6 + x + 1$
5	$x^5 + x + 1$	15	$x^{15} + x + 1$
6	$x^6 + x + 1$	16	$x^{16} + x^{12} + x^3 + x + 1$
7	$x^7 + x + 1$	17	$x^{17} + x^3 + 1$
8	$x^8 + x^4 + x^3 + x^2 + 1$	18	$x^{18} + x^7 + 1$
9	$x^9 + x^4 + 1$	19	$x^{19} + x^5 + x^2 + x + 1$
10	$x^{10} + x^3 + 1$	20	$x^{20} + x^3 + 1$
11	$x^{11} + x^2 + 1$	21	$x^{21} + x^2 + 1$

Tabel 2.2 Polinomial Primitif GF (2^m)

Polinomial generator yakni g(x) dibentuk berdasarkan polinomial minimal m(x). Polinomial minimal m(x) ditentukan untuk setiap g(x) yang memiliki koefisien dan akar-akarnya saling berkonjugasi. Maka polinomial generator dengan tingkat kemampuan perbaikan t pada BCH dibentuk dari gabungan perkalian terkecil LCM (*Least Common* Multiple) dari $\phi_1(x)$, $\phi_2(x), \ldots, \phi_{2t}(x)$. Dinotasikan sebagai berikut,

$$g(x) = LCM\{\phi_1(x), \phi_2(x), \ldots, \phi_{2t}(x)\}$$

Konjugat	Polinomial Minimal
0	x
1	$1 + x$
$\alpha, \alpha^2, \alpha^4, \alpha^8$	$1 + x + x^4$
$\alpha^3, \alpha^6, \alpha^9, \alpha^{12}$	$1 + x + x^2 + x^3 + x^4$
α^5, α^{10}	$1 + x + x^2$
$\alpha^7, \alpha^{11}, \alpha^{13}, \alpha^{14}$	$1 + x^3 + x^4$

Tabel 2.3 Polinomial Minimal dari Elemen pada GF (2^4) dibangkitkan oleh p(x) = 1 + x + x^4

Derajat polinomial tertinggi pada g(x) akan sesuai dengan teori $\left|\deg(g(x))\right| = n - k$. Untuk melakukan penyandian dibentuk suatu konstruksi dari perhitungan aritmatika dengan suatu nilai α yang mewakili posisi biner dari kombilasi biner 2^3.

Representasi Eksponen	Representasi Polinomial	Representasi Vektor		
		b0	b1	b2
0	0	0	0	0
1	1	1	0	0
α	α	0	1	0
α^2	α^2	0	0	1
α^3	α^3	1	1	0
α^4	$1 + \alpha$	0	1	1
α^5	$\alpha + \alpha^2$	1	1	1
α^6	$\alpha^2 + \alpha^3$	1	0	1

Tabel 2.4 Contoh representasi α dari elemen GF (2^3) dibangkitkan oleh p(x) = 1 + x + x^3

Tahap berikutnya adalah penyandian kode BCH yang mengikuti prinsip dasar penyandian siklis. Misalkan dalam BCH (15,5) dengan panjang kata sandi n = 15 dan panjang informasi k = 5, dan $g(x) = 1 + x + x^2 + x^4 + x^5 + x^8 + x^{10}$, dapat dibentuk kata sandi c(x) dengan memanfaatkan beberapa polinomial m(x) yaitu polinomial informasi, g(x) yaitu polinomial generator, v(x) yaitu polinomial hasil pembagian $x^{n-k}d(x)$ terhadap g(x), dan h(x) yaitu polinomial sisa pembagian alias,

$$h(x) + x^{n-k}m(x) = v(x)g(x)$$

Kata sandi c(x) sebafai informasi yang akan dikirim dirumuskan sebagai berikut,

$$c(x) = h(x) + x^{n-k}m(x)$$

2) Proses Dekode Kode BCH

Pada tahap penguraian sandi dilibatkan perhitungan sindrom dari kata sandi yang diterima. Sindrom, disimbolkan sebagai Si, dihitung menggunakan rumus umum:

$$S_i = r(a^i) = b_i(a^i)$$

dengan r(x) adalah polinomial kata sandi yang diterima, dan $b_i(x)$ adalah sisa dari pembagian r(x) dengan polinomial minimal $\phi_i(x)$ yang berkaitan dengan akar a^i. Selanjutnya, sindrom S_i digunakan untuk menentukan polinomial pola kesalahan e(x) dengan hubungan:

$$S_i = e(a^i)$$

Pola kesalahan e(x) memiliki w kesalahan pada lokasi $x^{j1}, x^{j2}, \ldots, x^{jw}$. Kemudian sindrom akan digunakan untuk mengidentifikasi lokasi dan jumlah kesalahan dalam kata sandi.

D. Kode Reed-Solomon

Pada bulan Desember 1958, Irving S. Reed dan Gustavo Solomon menyelesaikan laporan berjudul "Polynomial Codes Over Certain Finite Fields" di M.I.T. Lincoln Laboratory. Pada tahun 1960, laporan tersebut mengalami sedikit modifikasi dan diumumkan dalam jurnal *Society for Industrial and Applied Mathematics* (SIAM) sebagai Reed-Solomon Codes, yang menjadi inovasi terkini dalam perbaikan kesalahan. Reed-Solomon (RS) digunakan luas sebagai sistem koreksi kesalahan dalam berbagai aplikasi, seperti media penyimpanan data (hard disk, CD, DVD, dan barcode), komunikasi nirkabel (telepon bergerak dan microwave links), televisi digital, komunikasi satelit, dan modem broadband (ADSL, xDSL).

1) Proses Enkode Kode Reed-Solomon

Penyandian RS melibatkan penambahan sandi tambahan ke dalam data asli, memungkinkan deteksi dan perbaikan kesalahan saat data dibaca kembali atau diterima. Proses ini menggunakan polinomial generator yang dihasilkan dari perhitungan aritmatika finite field pada elemen GF(q), di mana q adalah jumlah elemen dalam finite field yang dibentuk dari q = 2^m. Kode Reed-Solomon dengan simbol GF(q) memiliki beberapa parameter yaitu,

Panjang sandi blok : n = q - 1 = 2^m - 1
Jumlah digit *parity check* : n – k = 2t
Jarak minimum : d_{min} = 2t + 1

dengan ini, generator polinomial dari sebuah polinomial primitif dengan panjang n adalah:

$$g(x) = (x + \alpha)(x + \alpha^2) \ldots (x + \alpha^{2t})$$
$$= g_0 + g_1 x + \cdots + g_{2t} x^{2t-1} + x^{2t}$$

dengan keterangan g(x) memiliki $\alpha, \alpha^2, \ldots, \alpha^{2t}$ sebagai akar-akar koefisien dari GF(2^m).

Dari segi proses pembentukan kata sandi pada Reed-Solomon sama dengan BCH. Misalkan,

$$a(x) = a + a_1 x + a_2 x^2 + \cdots + a_{k-1} x^{k-1}$$

adalah informasi yang hendak dijadikan kata sandi dengan k = n – 2t. Digit dari *parity check* (2t) adalah koefisien sisa pembagian

$$h(x) = h_0 + h_1 x + h_2 x^2 + \cdots + h_{2t-1} x^{2t-1}$$

yang dihasilkan dari pembagian polinomial informasi $x^{2t} a(x)$ terhadap polinomial g(x).

2) Proses Dekode Kode Reed-Solomon

Proses penguraian kode Reed-Solomon melibatkan tiga tahapan, termasuk perhitungan nilai kesalahan. Komponen sindrom 2t diperoleh dengan menggantikan nilai a^i ke dalam polinomial kata sandi yang diterima r(x) untuk i = 1, 2, ..., 2t sehingga

$S_1 = r(a)$
$S_2 = r(a^2)$
$S_{2t} = r(a^{2t})$

Komponen sindrom S_i juga dapat dihitung dengan melakukan pemabagian r(x) terhadap $x + a^i$.

$$r(x) = c_i(x)(x + a^i) + b_i$$

Dengan b_i sebagai sisa dari pembagian adalah sebuah konstanta dalam GF(2^m). Ketika a^i disubstitusikan ke dalam persamaan di atas maka akan didapatkan $S_i = b_i$.

Pencarian olinomial dari lokasi kesalahan dilakukan dengan perumusan,

$$\tau(x) = (1 + \beta_1 x)(1 + \beta_2 x) \ldots (1 + \beta_p x)$$
$$= 1 + \tau_1 x + \cdots + \tau_p x$$

Dengan $\beta_l = \alpha^{jl}$ untuk l = 1, 2, ..., p sebagai jumlah lokasi kesalahan.

Selanjutnya, polinomial lokasi kesalahan digunakan untuk menemukan dan memperbaiki kesalahan dalam kata sandi yang diterima, memastikan akurasi dan integritas data yang dikirim atau disimpan.

III. PEMBAHASAN

A. Penggabungan Kode BCH dan Kode Reed-Solomon pada Koreksi Kesalahan dalam Sistem Komunikasi Digital

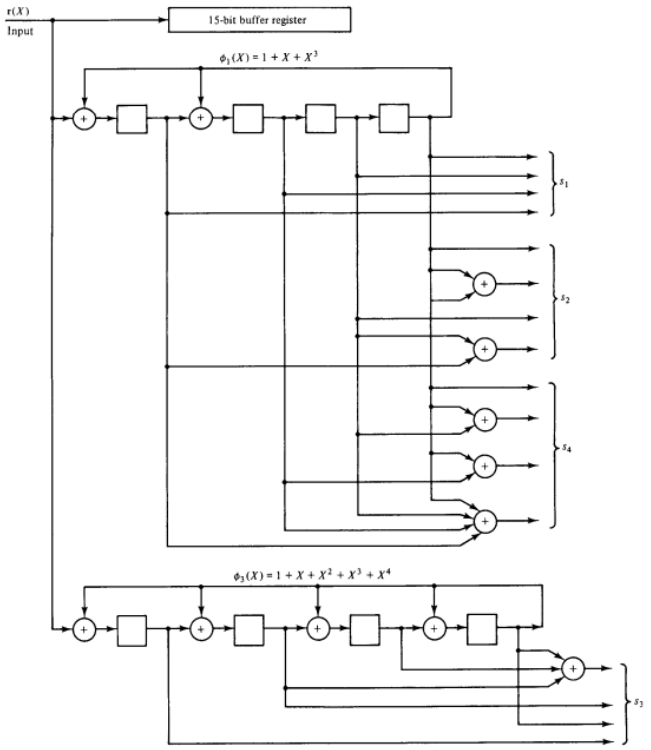
Penggabungan Reed-Solomon (15,5) dan BCH (15,5) di perangkat keras seringkali kompleks karena perbedaan karakteristik keduanya. Namun, solusi muncul dengan mengimplementasikannya pada prosesor DSP menggunakan program simulink. Pendekatan serial dipilih untuk menyederhanakan proses penyandian dan penguraian, memanfaatkan keluaran RS (15,5) sebagai masukan langsung untuk penyandi BCH (15,5), mengurangi kompleksitas perhitungan.

Proses serial melibatkan dua tahap terkait: penyandian RS (15,5) diikuti oleh penyandian ulang menggunakan BCH (15,5). Penguraian dilakukan dalam dua tahap terpisah, pertama dengan teknik BCH (15,5), dan kedua dengan teknik RS (15,5). Pendekatan ini, dengan penyandi RS sebagai "luar" dan penyandi BCH sebagai "dalam," terbukti efektif mengatasi kesalahan baik secara acak maupun berurutan, memungkinkan koreksi kesalahan yang mungkin tidak dapat diatasi oleh satu teknik saja.

B. Ilustrasi Penyandian Kode BCH dan Kode Reed-Solomon

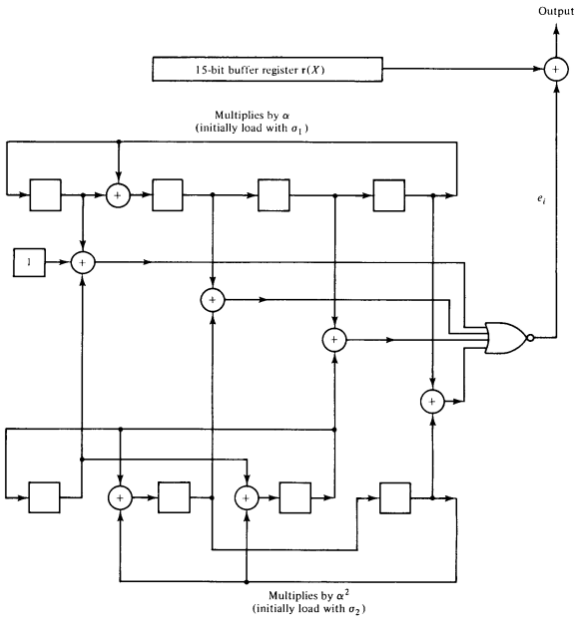
Pelaksanaan koreksi kesalahan pada decoding kode BCH dapat diwujudkan baik melalui perangkat keras digital maupun perangkat lunak yang diprogram pada komputer tujuan umum. Setiap bentuk implementasi memiliki kelebihan tersendiri, dan kami akan mempertimbangkan keduanya. Langkah awal dalam mendekode kode BCH untuk koreksi kesalahan-t adalah menghitung komponen sindrom 2t, seperti S1, S2, hingga S2t. Proses perhitungan ini melibatkan substitusi elemen bidang $\alpha, \alpha^2, \ldots, \alpha^{2t}$ ke dalam polinomial penerimaan r(x). Pada implementasi perangkat lunak, substitusi ini dilakukan dengan menggantikan nilai α^i ke r(x). Perhitungan ini melibatkan n - 1 penjumlahan dan n - 1 perkalian. Untuk kode biner BCH, dapat ditunjukkan bahwa S2t = Si^2. Dengan persamaan ini, komponen sindrom 2t dapat dihitung dengan melakukan (n-1)t penjumlahan dan nt perkalian.

Pada implementasi perangkat keras, komponen sindrom dapat dihitung menggunakan register pergeseran umpan balik. Dalam hal ini, kita dapat menggunakan salah satu jenis rangkaian yang ditunjukkan. Dari persamaan tersebut, kita dapat menyimpulkan bahwa polinomial generator adalah hasil kali dari paling banyak t polinomial minimal. Oleh karena itu, diperlukan paling banyak t register pergeseran umpan balik, masing-masing terdiri dari paling banyak m tahapan, untuk membentuk komponen sindrom 2t. Proses perhitungan ini dilakukan saat polinomial r(x) yang diterima memasuki decoder. Begitu seluruh r(x) memasuki decoder, komponen sindrom 2t dapat terbentuk. Seluruh komputasi memerlukan n siklus jam untuk diselesaikan. Sirkuit komputasi sindrom untuk kode BCH koreksi kesalahan ganda (15, 7) diperlihatkan pada Gambar 3.6, di mana dua register pergeseran umpan balik, masing-masing terdiri dari empat tahapan, digunakan. Pada rangkaian tersebut digunakan konsep dasar dari rangkaian logika yang merupakan salah satu ekspresi dari fungsi Boolean.



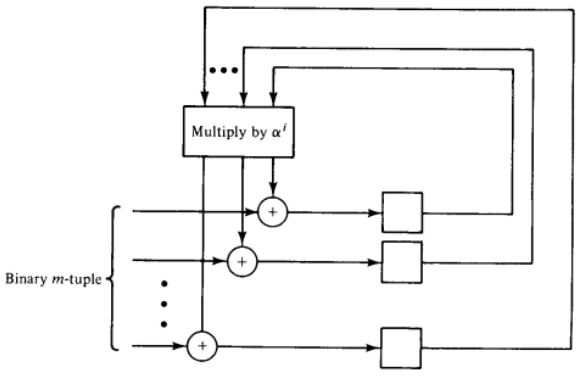
Gambar 3.6 Rangkaian Komputasi Sindrom untuk Koreksi Kesalahan Ganda Kode BCH (15,7)
(Sumber: https://pg024ec.files.wordpress.com/2013/09/error-control-coding-by-shu-lin.pdf)

Langkah terburuk dalam perhitungan lokasi kesalahan dan koreksi melibatkan substitusi n elemen dalam bidang ke dalam polinomial σ(X) berderajat 1 untuk menemukan akar-akarnya. Implementasi perangkat lunak memerlukan banyak operasi dan tambahan nt, tetapi dapat dilakukan lebih efisien dengan perangkat keras menggunakan rangkaian pencarian Chien. Rangkaian ini memerlukan pengganda yang mengalikan dengan $\alpha, \alpha^2, \ldots, \alpha^t$ secara berturut-turut. Hasil dari langkah 2 dimuat ke dalam register pengganda dan digeser n kali. Jika jumlahnya nol, α^{n-1} diidentifikasi sebagai nomor lokasi kesalahan. Jika tidak, α^{n-1} bukan nomor lokasi kesalahan. Langkah ini membutuhkan n siklus jam atau hanya k siklus jam untuk mengoreksi digit pesan. Rangkaian pencarian Chien untuk kode BCH koreksi kesalahan ganda (15, 7) ditunjukkan dalam Gambar 3.7



Gambar 3.7 Rangkaian Pencarian Chien untuk Kode BCH Koreksi Kesalahan Ganda (15,7)
(Sumber: https://pg024ec.files.wordpress.com/2013/09/error-control-coding-by-shu-lin.pdf)

Untuk nilai t dan m, biaya pembangunan pengali $\alpha, \alpha^2, \ldots, \alpha^t$ menjadi tinggi, alias t buah kabel. Inisialisasinya, σ_i akan berada pada register B dan α^i di register C. Setelah m siklus jam, produk $\sigma_i\alpha^i$ akan berada di register A. Untuk membentuk $\sigma_i\alpha^{2i}$ maka $\sigma_i\alpha^i$ dimasukkan ke register B. Setelah m siklus jam berikutnya, $\sigma_i\alpha^{2i}$ akan berada di register A. Dengan pengali ini, diperlukan siklus nm jam untuk menyelesaikan langkah ketiga dalam mendekode kode biner BCH. Langkah 1 dan 3 melibatkan jumlah perhitungan yang sekitar sama. Dengan implementasi perangkat keras, langkah 1 dapat dilakukan saat membaca polinomial yang diterima r(x), dan langkah 3 dapat diselesaikan saat membaca r(x). Waktu komputasi pada langkah 1 dan 3 pada dasarnya dapat diabaikan. Hal yang sama juga dilakukan pada kode Reed-Solomon yang disesuaikan dengan perhitungan pada enkode dan dekode dari Reed-Solomon.



Gambar 3.8 Rangkaian Komputasi Sindrom untuk Kode Reed-Solomon dalam Bentuk Biner
(Sumber: https://pg024ec.files.wordpress.com/2013/09/error-control-coding-by-shu-lin.pdf)

Seperti yang dapat dilihat pada gambar 3.6, gambar 3.7, dan gambar 3.8, penerapan aljabar boolean pada enkode dan dekode kode Reed-Solomon serta kode BCH memiliki peran sentral dalam memastikan kehandalan komunikasi digital. Pada proses penyandian (enkode) kedua kode tersebut, operasi XOR (OR eksklusif) menggunakan prinsip aljabar boolean untuk menggabungkan bit-bit data dengan tepat. Selain itu, perhitungan polinomial pada bidang Galois, yang melibatkan operasi aljabar boolean, turut mendukung proses penyandian pada kedua jenis kode. Dalam proses dekode, khususnya pada kode Reed-Solomon, operasi XOR dan penyusunan ulang bit dengan prinsip aljabar boolean digunakan untuk mendeteksi dan memperbaiki kesalahan. Pada kode BCH, pencarian lokasi kesalahan dengan metode Chien Search memanfaatkan operasi XOR dan logika boolean pada bidang Galois. Keseluruhan, pemahaman yang mendalam tentang aljabar boolean menjadi kunci dalam mengimplementasikan operasi-operasi bit yang esensial untuk keandalan komunikasi digital pada kedua jenis kode tersebut.

C. Percobaan Perhtungan Manual dalam Penyandian Kode Reed-Solomon dan BCH

Berikut akan dilakukan percobaan perhitungan secara manual dengan mengikuti persamaan-persamaan yang telah dijelaskan sebelumnya pada landasan teori untuk kode BCH dan kode Reed-Solomon dengan operasi aritmatik terhadap koefisien mengacu pada nilai α tabel 2.4. Pada pengujian akan diuji teori penyandian BCH (7,4) dan teori penyandian Reed-Solomon (7,5).

1) Penyandian BCH

step	Perhitungan
1	Pesan m = [1 1 0 1]
2	$m(x)$? $m(x) = 1 + x + x^3$
3	$g(x)$? $g(x) = p(x) = 1 + x + x^3$
4	$x^{n-k}m(x) = x^3 + x^4 + x^6$
5	$\frac{x^{n-k}m(x)}{g(x)} = v(x) + \frac{h(x)}{g(x)}$ $v(x) = x^3$ $h(x) = 0$
6	$c(x) = h(x) + x^{n-k}m(x)$ $c(x) = x^3 + x^4 + x^6$ c = [0 0 0 1 1 0 1]
7	$r(x) = c(x) + e(x)$ r = [0 0 0 1 0 1] $r(x) = x^4 + x^6$
8	$\frac{r(x)}{g(x)} = v(x) + \frac{s(x)}{g(x)}$ $s(x) = x^3$
9	Kata sandi hasil dekode $c'(x) = s(x) + r(x)$ $c'(x) = x^3 + x^4 + x^6$ c' = [0 0 0 1 1 0 1]
12	Pesan hasil penyandian m' = [1 1 0 1]

Tabel 3.1 Perhitungan Manual Penyandian Kode BCH (7,4)

2) Penyandian Reed-Solomon

step	Perhitungan
1	Pesan m = [2 6 1 0 3] = [010 110 001 000 011]
2	$m(x)$? $m(x) = \alpha + \alpha^3 x + \alpha^2 x^2 + \alpha^4 x^4$
3	$g(x)$? $g(x) = (x + \alpha)(x + \alpha^2)$
4	$x^{n-k}m(x)$ $= \alpha x^2 + \alpha^3 x^3 + \alpha^2 x^4 + \alpha^4 x^6$
5	$\frac{x^{n-k}m(x)}{g(x)} = v(x) + \frac{h(x)}{g(x)}$ $v(x) = \alpha x + \alpha^3 + \alpha x^2 + \alpha x^3 + \alpha^4 x^4$ $h(x) = \alpha^6 + \alpha^5 x$
6	$c(x) = h(x) + x^{n-k}m(x)$ $c(x) = \alpha^6 + \alpha^5 x + \alpha x^2 + \alpha^3 x^3 + \alpha^2 x^4 + \alpha^4 x^6$ c = [101 111 010 110 001 000 011] = [5 7 2 6 1 0 3]
7	$r(x) = c(x) + e(x)$ r = [5 7 2 0 1 0 3] = [101 111 010 000 001 000 011] $r(x) = \alpha^6 + \alpha^5 x + \alpha x^2 + \alpha^2 x^4 + \alpha^4 x^6$
8	$S_i = r(a^i)$ $S_1 = \alpha^6 + \alpha^6 + \alpha^3 + \alpha^6 + \alpha^{10}$ $= \alpha^6$ $S_2 = \alpha^6 + \alpha^7 + \alpha^5 + \alpha^{10} + \alpha^9$ $= \alpha^2$
9	Letak kesalahan σ $[S_1][\sigma] = [S_2] \rightarrow [\sigma] = [S_1]^{-1}[S_2]$ $[\sigma] = [\alpha^6]^{-1}[\alpha^2] = \alpha^3$
10	Letak kesalahan e $[e][\sigma] = [S_1] \rightarrow [e] = [\sigma]^{-1}[S_1]$ $[e] = [\alpha^3]^{-1}[\alpha^6] = \alpha^3$ $e(x) = \alpha^3 x^3$
11	Kata sandi hasil dekode $c'(x) = r(x) + e(x)$ $c'^{(x)} = \alpha^6 + \alpha^5 x + \alpha x^2 + \alpha^2 x^4 + \alpha^4 x^6 + \alpha^3 x^3$ c' = [101 111 010 110 001 000 011] = [5 7 2 6 1 0 3]
12	Pesan hasil penyandian m' = [2 6 1 0 3]

Tabel 3.2 Perhitungan Manual Penyandian Kode Reed-Solomon (7,5)

IV. KESIMPULAN

Kesimpulan dari teks di atas menyoroti pentingnya penerapan konsep Aljabar Boolean dalam koreksi kesalahan pada Sistem Komunikasi Digital dengan menggunakan penggabungan Kode BCH dan Kode Reed-Solomon. Aljabar Boolean, yang merupakan dasar matematis untuk manipulasi logika biner, memberikan landasan teoretis bagi pengembangan teknik koreksi kesalahan seperti Kode BCH dan Kode Reed-Solomon. Dalam konteks ini, Kode Hamming, yang memanfaatkan konsep Aljabar Boolean, menjadi krusial dalam mendeteksi dan memperbaiki kesalahan satu bit pada informasi. Lebih lanjut, penggabungan Kode BCH dan Kode Reed-Solomon, dengan menerapkan konsep Aljabar Boolean, meningkatkan efisiensi dan keandalan koreksi kesalahan pada Sistem Komunikasi Digital. Dengan memanfaatkan operasi dasar dan hukum-hukum Aljabar Boolean, proses enkode dan dekode pada Kode BCH dan Kode Reed-Solomon dapat diimplementasikan secara efektif, menjadikan koreksi kesalahan semakin tangguh dalam menghadapi tantangan kesalahan bit dan paket pada saluran komunikasi digital.

V. UCAPAN TERIMA KASIH

Dengan penuh rasa syukur kepada Tuhan Yang Maha Esa atas rahmat dan kasih karunia-Nya, penulis berhasil menyelesaikan makalah ini tanpa kendala yang berat. Penulis ingin mengungkapkan terima kasih kepada keluarga yang memberikan dukungan, serta kepada dosen-dosen pengampuh mata kuliah IF2120 Matematika Diskrit – Sem. I Tahun 2023/2024, khususnya kepada Ibu Fariska Zakhralativa Ruskanda, S.T.,M.T. yang telah memberikan pengetahuan yang sangat berharga. Semoga makalah ini dapat bermanfaat sebagai referensi bagi para pelajar dan pembaca yang ingin mendalami ilmu ini atau bahkan menjadi sumber inspirasi bagi penulis di masa depan.

REFERENCES

[1] Lin, Shu, *Error Control Coding: Fundamentals and Applications*, New Jersey, CA: Prentice-Hall, 1983

[2] Fajrian. Khotman Hilmy, *Simulasi Error Correction dengan Penggabungan Teknik Reed-Solomon Code (15,5) dan BCH code (15,5) menggunakan dsk TMs320c6713*, Depok: Universitas Indonesia, 2010.

[3] https://herdaradio.com/id/blog/radioknowledge/forward-error-correction/ (diakses, Jumat, 8-12-2023)

[4] https://repository.unair.ac.id/58144/1/kk%20mpm%20246.91%20Suk%20p.pdf (diakses, Jumat, 8-12-2023)

[5] Munir, Rinaldi. 2023. Aljabar Boolean (Bag. 1) Bahan Kuliah IF 2120 Matematika Diskrit. Bandung. Prodi Informatika, Sekolah Teknik Elektro dan Informatika, (diakses pada 7 Desember 2023).

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 8 Desember 2023

Zahira Dina Amalia 13522085