HAMMING CODE

5301413057 - SUMARSONO

Dasar Teori

Kode ini dikenalkan oleh Richard Hamming (1950) sebagai kode tunggal pengoreksi kesalahan (*single error-correcting code*). Kode ini dan variansinya telah lama digunakan untuk kontrol kesalahan pada sistem komunikasi digital.[1]

Cara kerja dari hamming codse yaitu Bit paritas tambahan diberikan pada bit-bit informasi sebelum ditransmisikan, sedangkan pada sisi penerima dilakukan pengecekan dengan algoritma yang sama dengan peghitungan bit paritas[1].

Penambahan bit paritas diilustrasikan sebagai berikut:

11	10	9	8	7	6	5	4	3	2	1
D1	D2	D3	p	D4	D5	D6	p	D7	p	p

Kode Hamming biner dapat direpresentasikan dalam bentuk persamaan:

$$(n,k) = (2^r-1, 2^r-1-r)$$

r = jumlah paritas

k = jumlah data

n = jumlah bit informasi yang membentuk n sandi

Contoh:

jika r = 3 maka:

$n = 2^r - 1 = 7$

$k = 2^r - 1 - r = 4$

sehingga kode Hamming nya adalah C (7,4). [2]

Cara Menentukan Jumlah dan Posisi Bit Paritas

Jumlah paritas dapat ditentukan dengan persamaan:

$$2^r \geq k + r + 1$$

K = jumlah bit data

r = jumlah paritas

Contoh:

1. Jika diketahui data 1010, maka:
   k = 4
   r = yang mungkin adalah 3
2. Jika diketahui data 1001101, maka:

k = 7
r = yang mungkin adalah 4

Sedangkan lokasi peletakan bit paritas ditentukan dengan persamaan:

$$2^n, \text{ dengan } n \geq 0$$

Sehingga, lokasi yang mugkin adalah: 1, 2, 4, 8, 16, dst.

ENCODING HAMMING CODE

Misal diketahui data 1001, maka:

D1	D2	D3	D4
1	0	0	1

k = 4

r = 3

n = 7

sehingga

7	6	5	4	3	2	1
1	0	0	P4	1	P2	P1

Menghitung nilai P1:

P1 = VRC (Vertical redundancy check) dari bit 1, 3, 5, 6 ,7, 9, 11 (cek satu, lompati satu)

7	6	5	4	3	2	1
1	0	0	P4	1	P2	P1

P1 = 1 ⊕ 3 ⊕ 5 ⊕ 7

P1 = ? ⊕ 1 ⊕ 0 ⊕ 1

P1 = 0

INGAT:

X1	X2	X1 ⊕ X2
0	0	0
0	1	1
1	0	1
1	1	0

Sehingga menjadi:

7	6	5	4	3	2	1
1	0	0	P4	1	P2	0

Menghitung nilai P2:

P2 = VRC (Vertical redundancy check) dari bit 2, 3, 6 ,7, 10, 11 (Cek dua, lompati dua)

7	6	5	4	3	2	1
1	0	0	P4	1	P2	0

P2 = 2 ⊕ 3 ⊕ 6 ⊕ 7

P2 = ? ⊕ 1 ⊕ 0 ⊕ 1

P2 = 0

Sehingga menjadi:

7	6	5	4	3	2	1
1	0	0	P4	1	0	0

Menghitung nilai P4:

P4 = VRC (Vertical redundancy check) dari bit 4, 5, 6, 7 (Cek 4, lompati 4)

7	6	5	4	3	2	1
1	0	0	P4	1	P2	0

P4 = 4 ⊕ 5 ⊕ 6 ⊕ 7

P4 = ? ⊕ 0 ⊕ 0 ⊕ 1

P4 = 1

Sehingga menjadi:

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

Jadi hasil dari encodingnya adalah sebagai berikut:

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

Hasil encoding tersebut kemudian ditransmisikan untuk selanjutnya di decoding oleh penerima.

DECODING HAMMING CODE (DETEKSI ERROR)

Misal blok data hasil encoding adalah

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

Kemudian dimisalkan terjadi error pada bit ke 3 ketika proses transmisi sehingga data menjadi:

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

Data yang mengandung error ini kemudian di decode menggunakan metode hamming. Prosesnya adalah sebagai berikut:

1. Menentukan lokasi bit paritas, sama seperti proses encoding :

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

2. Menentukan nilai bit paritas

   P1 = 1 ⊕ 3 ⊕ 5 ⊕ 7
   P1 = 0 ⊕ 0 ⊕ 0 ⊕ 1 = 1

   P2 = 2 ⊕ 3 ⊕ 6 ⊕ 7
   P2 = 0 ⊕ 0 ⊕ 0 ⊕ 1 = 1

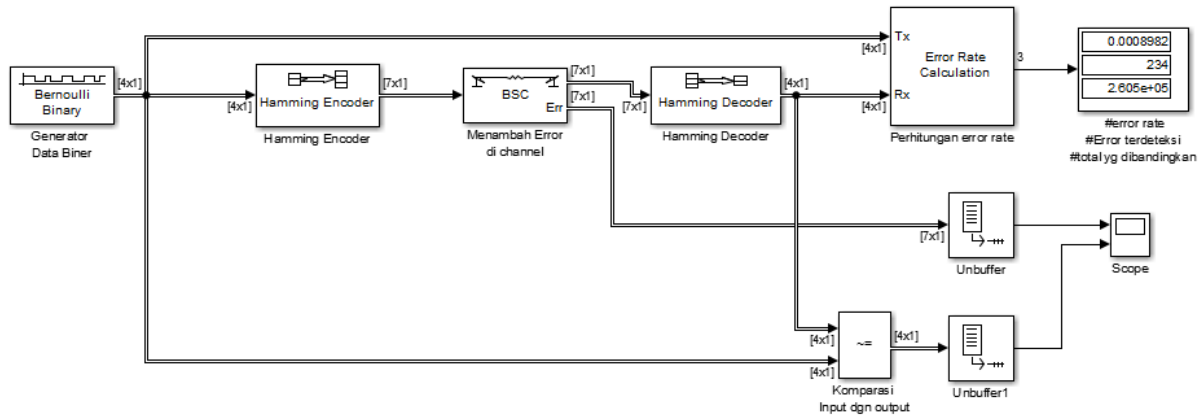
   P4 = 4 ⊕ 5 ⊕ 6 ⊕ 7
   P4 = 1 ⊕ 0 ⊕ 0 ⊕ 1 = 0

   0 1 1 = 3

   Jadi terdeteksi error pada bit ke 3 dari data yang diterima, selanjutnya error ini diperbaiki. Karena datanya binner maka jika nilai bit error adalah 0, diperbaiki dengan menggantinya menjadi 1. Begitu juga sebaliknya. Hamming code merupakan *single error-correcting code* maka hamming code hanya mampu mendeteksi satu error pada satu blok paket data.

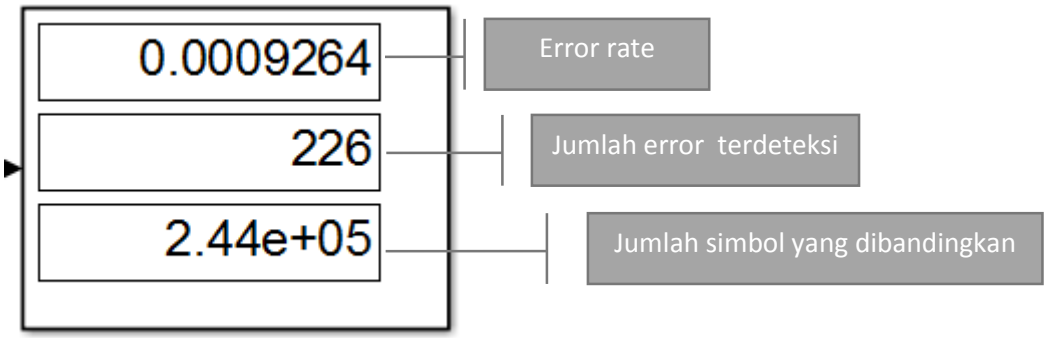
SIMULASI HAMMING CODE (7,4) MENGGUNAKAN SIMULINK MATLAB

Gambar rangkaian:



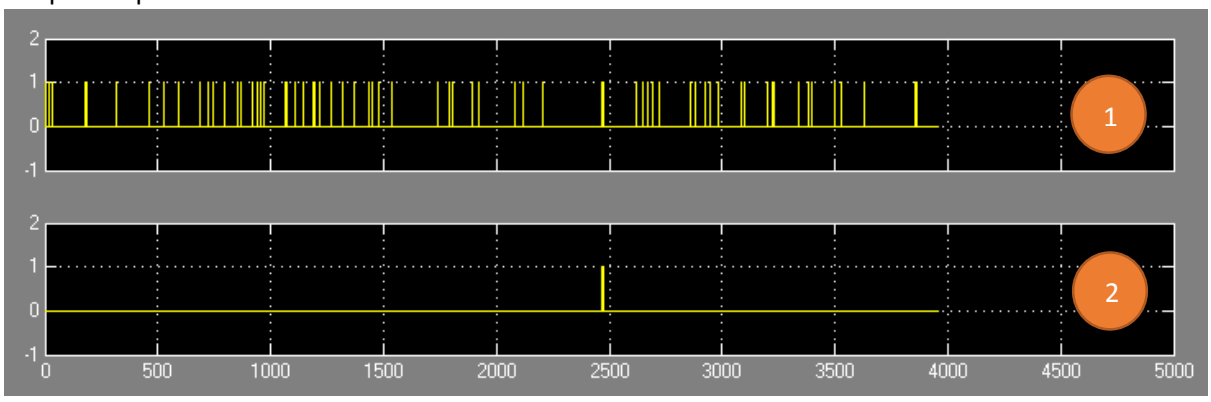
Cara kerja rangkaian:

Data biner frame-based 4 sample per frame secara acak dihasilkan oleh oleh bernoulli binary generator, data ini sebut saja X. Selanjutnya X masuk ke hamming encoder untuk di kodekan menggunakan metode hamming[7,4], hasil encode sebut saja Y. Selanjutnya Y ditransmisikan melalui kanal, dalam kanal data diberi error oleh binary symmetic channel, Y+error ini sebut saja Z. Kemudian Z diterima disisi pengguna, masuk ke hamming decoder untuk di pecahkan kodenya, dideteksi errornya dan diperbaiki guna memperoleh data asli yaitu X. Selanjutnya dilakukan perbandingan data asli dengan data hasil dekode, untuk melihat error rate dan jumlah error yang terdeteksi, dapat dilihat melalui display error rate.



Selain itu, dibandingkan juga jumlah error sebelum proses decode dan sesudah decode, dapat dilihat melalui scope

Output scope:

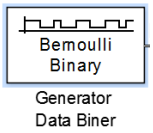


Grafik bagian atas (1), menunjukan jumlah error sebelum proses decode.

Grafik bagian bawah (2), menunjukan jumlah error sesudah proses decode

Penjelasan per blok[3]

Bernoulli Binary

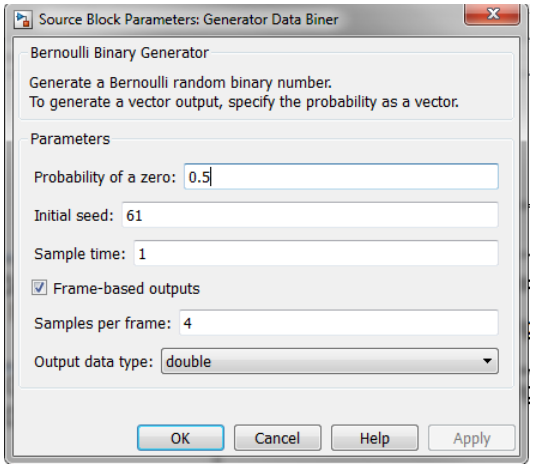


Generator
Data Biner

Merupakan blok yang menghasilkan data biner dengan prinsip kerja distribusi bernoulli.

$$f(k;p) = \begin{cases} p & \text{if } k = 1, \\ 1-p & \text{if } k = 0. \end{cases}$$

Blok ini memiliki parameter p, sehingga akan menghasilkan nol dengan probabilitas p dan menghasilkan satu dengan probabilitas 1-p, dengan 0 < p < 1.



Probability of a zero

Probabilitas munculnya keluaran nol. Isikan dengan nilai diantara 0 sampai 1. Nilai 0.5 artinya peluang munculnya 0 atau 1 sama besar.

Initial seed

Initial seed untuk generator bilangan acak.

Sample time

Merupakan jeda untuk setiap frame.

Frame-based output

Ceklist agar output dikonvert menjadi blok data sesuai frame size. Tujuannya agar sesuai dengan input encoder hamming.

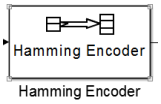
Samples per frame

Jumlah sample untuk setiap frame, isikan 4 menyesuaikan encoder hamming code.

Output data type

Tipe data output dari generator.

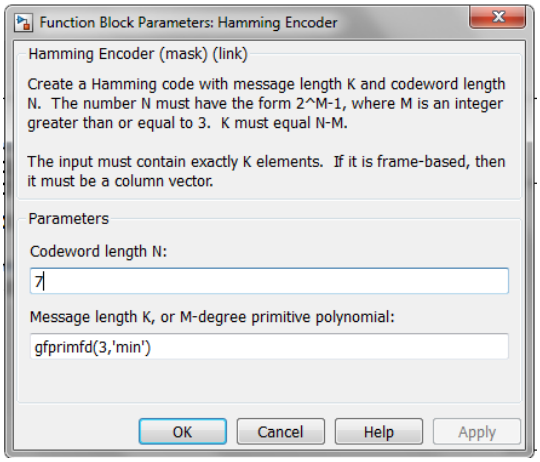
Hamming Encoder



Hamming Encoder

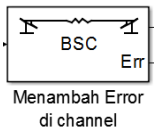
Merupakan bagian/sub dari librari Error Detection and Correction.

Hamming encoder mengkodekan data sebelum data tersebut dikirim melalui kanal.



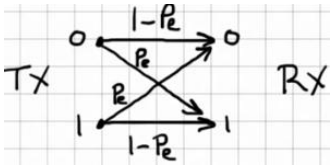
Secara default, simulasi hamming code pada simulink matlab adalah [7,4], yang artinya mengkodekan data dengan panjang 4 menjadi sepanjang 7, sebagaimana saya jelaskan pada dasar teori ENCODING HAMMING CODE. Hasilnya panjang data yang dikirim adalah 7. Fungsi dari kode yang ditambahkan adalah untuk memperbaiki satu error pada data yang ditransmisikan. Panjang input untuk hamming encoder adalah 4, maka pada bernoulli binnary diseting frame based output dengan sample per frame 4.

Binary Symmetric Channel



Menambah Error di channel

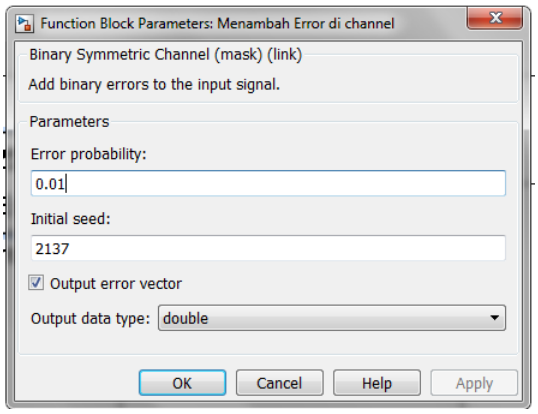
Fungsi dari blok ini adalah untuk memberikan error pada data yang dikirim melalui kanal. Cara kerja blok ini adalah sebagai berikut:



Jadi jika blok ini mendapat input 0 atau 1 maka outputnya adalah acak, bisa 0 bisa 1 sesuai konsep probabilitas. Blok ini menggunakan rumus matematis sebagai berikut:

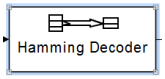
$$= \binom{n}{k} P_e^k (1-P_e)^{n-k}$$

Nilai Pe dapat kita tentukan dengan memasukan nilainya pada setting blok BSC, tepatnya pada bagian *error probability*



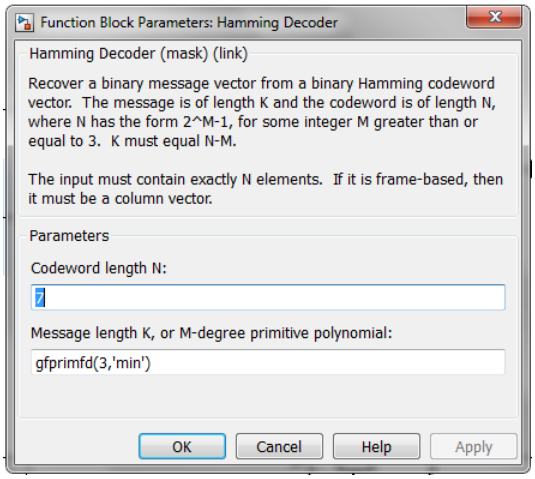
Ceklis pada bagian Output error vector agar nilai error dari blok BSC dapat diproses untuk keperluan ditampilkan pada bagian scope.

Hamming Decoder

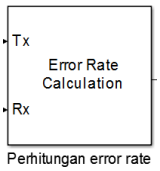


Hamming Decoder

Merupakan pasangan dari hamming encoder. Cara kerjanya sesuai dengan yang saya jelaskan pada dasar DECODING HAMMING CODE (DETEKSI ERROR). Setting disamakan dengan blok hamming encoder.

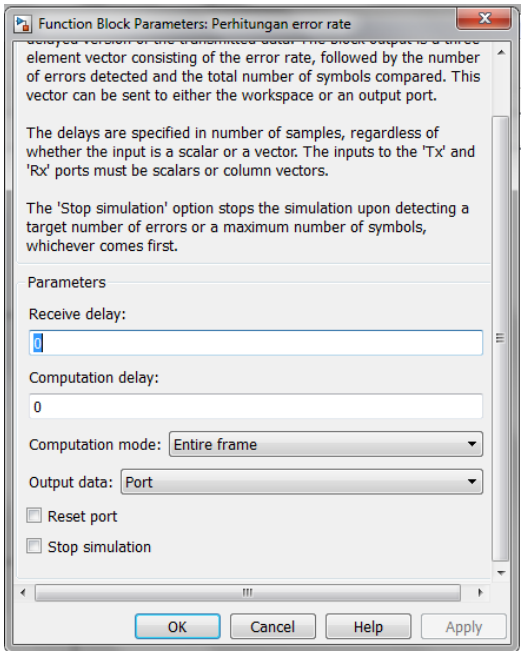


Error Rate Calculation



Perhitungan error rate

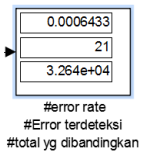
Blok ini berfungsi menghitung error rate dengan cara membandingkan data yang diterima dengan data yang dikirim. Untuk dapat melihat output dari blok ini diperlukan blok Display. Selanjutnya mengatur paramater Error rate calculation.



Output data: Port tujuannya agar nilai output dapat dimasukan ke blok display. Hasilnya nanti diperoleh tiga data yaitu: error rate, jumlah error yang ditemukan dan jumlah simbol yang dibandingkan.

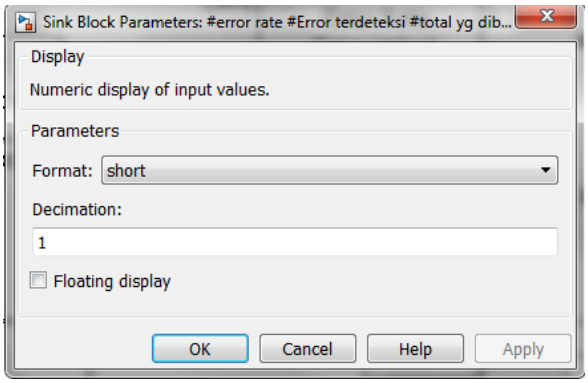
Hapus ceklis **Stop Simulation** agar simulasi tetap berjalan meskipun ditemukan error pada data.

Display

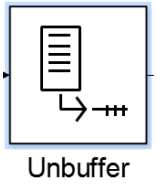


Blok ini berfungsi untuk menampilkan data dari blok error rate calculation.

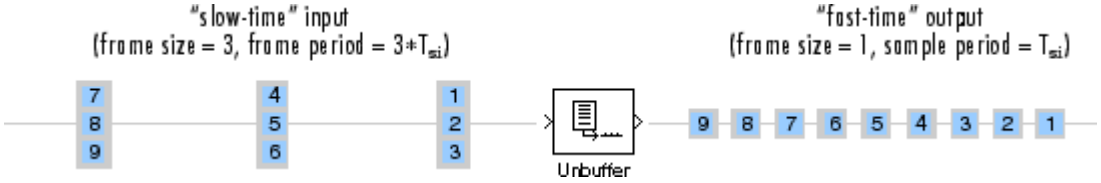
Blok ini akan menampilkan tiga data sesuai dengan masukannya, yaitu Error rate, jumlah error yang terdeteksi, dan jumlah simbol yang dibandingkan. Format data diatur dibagian settingnya.



Unbuffer

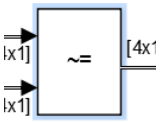


Berfungsi untuk mengkonversi data frame-based menjadi skalar.

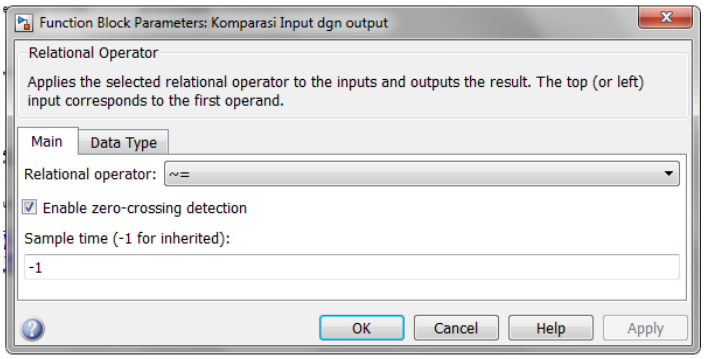


Tujuannya, agar data dapat ditampilkan pada scope.

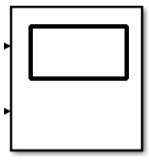
Relational Operator



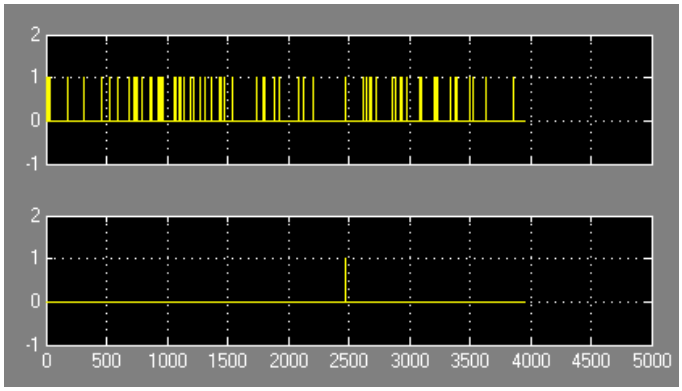
Merupakan operator hubungan dua data. Dalam hal ini diset untuk membandingkan data asli yang dikirim (sebelum di encode) dengan data yang diterima (hasil decode) dengan cara mencari bedanya. Oleh sebab itu dipilih operator "tidak sama dengan" **~=**



Scope



Blok ini berfungsi untuk melihat grafik dari data, dalam hal ini adalah data asli dengan data yang diterima. Output dari scope:



Referensi:

[1] https://www.academia.edu/9155180/Hamming_Code

[2] http://elib.unikom.ac.id/files/disk1/468/jbptunikompp-gdl-sindrianil-23380-9-10-error-g.pdf

[3] http://www.mathworks.com/help/index.html?s_cid=doc_ftr