IMPLEMENTASI *ENCODER* SANDI *REED SOLOMON* PADA *CONTROLLER AREA NETWORK*

Wisnu Kartika , I Wayan Mustika , Agus Bejo
Jurusan Teknik Elektro dan Teknologi Informasi, Fakultas Teknik, Universitas Gadjah Mada
Jl. Grafika no. 2 , Yogyakarta, Indonesia
wisnukartika_te07@mail.ugm.ac.id, mustika@gmail.com, agusbj@ugm.ac.id

Abstrak

EMI (Electromagnetic Interference) banyak ditemui pada sistem otomotif dan industri yang menggunakan kabel untuk menghubungkan antar device. Masalah utama saat ini ialah rentan terjadinya interferensi pada komunikasi antar device pada Controller Area Network (CAN). Maka akan diusulkan suatu skema rancangan untuk mengatasi burst error. Akan digunakan metode Reed Solomon Code dengan panjang kode (31, 27). Penelitian ini dapat membantu untuk mengurangi electromagnetic interference yang sering terjadi pada industri dan otomotif. Hasil dari penelitian ini ialah perhitungan bit paritas dan informasi yang akan dikirim melalui encoder sandi Reed Solomon.

Kata kunci- ARQ, Controller Area Network, CRC, Reed Solomon codes.

I. PENDAHULUAN

Pada saat ini komunikasi data digital pada suatu sistem jaringan baik *point to point*, *point to multipoint* atau *broadcast* menggunakan proses pemulihan konvensional seperti ARQ, FEC dan CRC. *Automatic Repeat Request* (ARQ) adalah teknik mengirim ulang data yang tidak sampai pada penerima dengan sistem *Acknowledgement* (ACK). *Forward Error Correction* (FEC) adalah teknik pengiriman dan penerimaan data dalam komunikasi *point to multipoint*. Jika pada penerima terkena *error* maka data akan dipulihkan kembali. Sedangkan *Cyclic Redundancy Check* adalah teknik memulihkan data yang terkena *error* pada penerima dengan menggunakan runtun bit CRC.

Pada penelitian ini digunakan FEC karena dapat menghemat waktu dan digunakan pada komunikasi *broadcast*.

Penelitian difokuskan pada CAN karena perkembangan teknologi pada saat ini banyak menggunakan CAN terutama pada bidang otomotif dan industri.

Permasalahan yang sering terjadi pada CAN adalah adanya EMI yang menyebabkan *burst error* pada suatu jaringan. EMI ini disebabkan oleh *inductive load* pada sistem. Ada suatu metode pemulihan *error* (*error correcting code*) untuk mengatasi *burst error* dan *random error*. Metode tersebut adalah *channel coding*. *Channel coding* adalah suatu metode untuk mengatasi *error* dengan melakukan proses penambahan *parity bit* dan kemudian dengan adanya *parity bit* ini dapat dipulihkan kembali informasi yang terkena *error*. Salah satu jenis *channel coding* adalah sandi Reed Solomon. Dengan metode sandi RS dapat mengatasi *burst error* yang disebabkan oleh EMI.

Channel coding dengan sandi Reed Solomon diterapkan dalam berbagai macam sistem jaringan. Sandi Reed Solomon ini digunakan pada jaringan komunikasi data. Jaringan yang sering menggunakan *channel coding* ini adalah jaringan *multi cast* atau *broadcast* atau *point to multipoint*. *Broadcast* adalah transmisi data dari satu sumber ke beberapa *node* penerima. Jika terjadi *error* pada media (dalam hal ini media udara) maka pada penerima akan dilakukan proses koreksi error oleh sandi Reed Solomon. *Broadcast* digunakan pada komunikasi AM, FM, jaringan TV.

Peneliti (Shabour, 2013) melakukan penelitian tentang bagaimana mengatasi EMI pada CAN *bus*. EMI ini sangat mengganggu dalam hal proses pengiriman data karena EMI termasuk *burst error*. Pada paper (Shabour, 2013) ini memiliki kelebihan yaitu peneliti menggunakan CRC untuk mengatasi *burst error*. Tetapi memiliki kelemahan yaitu masih ada *error* yang belum dapat terkoreksi. Hal ini disebabkan jumlah *error* melebihi batas *error* yang dapat diatasi oleh sandi RS.

Pada paper ini akan dilakukan proses perhitungan dengan sandi RS. Karena sandi RS dapat melakukan koreksi *error* pada penerima. Sehingga dapat menghemat dan efisiensi waktu. Sandi RS yang akan dihitung adalah RS (31, 27). Dengan *codeword* n adalah 31 simbol dan panjang informasi k adalah 27 simbol. Perhitungan dimulai dengan memasukkan 27 simbol dengan setiap simbol terdiri atas 5 bit. Perhitungan menggunakan operasi XOR. Perhitungan dibagi menjadi 3 tahap. Menghitung *generator polynomial* g(x), pembagian g(x) dan informasi dan menambahkan sisa hasil bagi pada informasi. Proses menambahkan sisa hasil bagi ini dapat juga disebut dengan

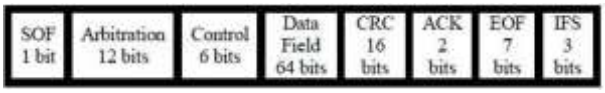
ISBN 978-602-99334-4-4

proses penambahan *parity bit*. *Parity bit* ini yang akan melengkapi simbol yang akan dikirim melalui media. Secara rinci proses perhitungan adalah sebagai berikut. Dari 27 simbol informasi tersebut dikalikan dengan penggeseran pangkath x^{n-k}. Kemudian dilakukan proses perkalian untuk menghitung g(x). Kemudian 27 simbol yang telah dikalikan tadi dibagi dengan polinomial g(x). Dari hasil pembagian akan didapatkan hasil bagi dan sisa hasil bagi. Sisa hasil bagi inilah yang kemudian ditambahkan ke LSB (*Least Significant Bit*) dari 27 simbol tersebut. Maka data siap dikirim melalui media.

Paper ini akan dibagi menjadi beberapa bagian. Penjelasan mengenai CAN akan dibahas pada Bab II. Desain dan Perancangan dibahas pada Bab III. Hasil dan Pembahasan akan dibahas pada bab IV. Kesimpulan akan dibahas pada Bab V.

II. CONTROLLER AREA NETWORK

Controller Area Network adalah sebuah sistem yang menggunakan topologi *bus* untuk menghubungkan antar *device*. Dapat juga digunakan untuk proses transfer data. *Layer physical* yang menggunakan *differential transmission* pada *twisted pair wire*. Sebuah *non destructive bit wise arbitration* yang digunakan untuk mengendalikan akses ke bus. Pesan yang dikirim kecil 8 bytes dan dilindungi oleh *checksum*.



Gambar 1 Format Frame CAN Standard (Shabour, 2013)

Saat ini CAN digunakan untuk beberapa aplikasi pada kendaraan seperti transfer data antara *sensor* dan *actuator* pada *automobile*. Kemampuan CAN saat ini sedang diuji untuk menjadikan aman dan efisien serta *latency*, *data rate* yang tinggi, kekebalan terhadap *noise* dan kemampuan memperbaiki *error*. Beberapa faktor yang mengurangi efisiensi CAN sistem *bus* ialah *error* yang disebabkan oleh EMI, *stop and wait retransmission* yang tidak efisien, *bit overhead* yang tinggi, sebuah sistem *broadcast* yang membutuhkan *retransmission* jika ada *node* yang mengalami *flag error*, dan panjang CAN *bus*.

Burst error pada CAN disebabkan oleh EMI. EMI ini disebabkan oleh *inductive load* dan interferensi radio eksternal pada *wiring*. Kemudian dengan adanya EMI ini akan menyebabkan *burst error* pada proses pengiriman data yang menggunakan suatu media baik itu kabel maupun udara. Kemudian salah satu kemampuan sandi RS adalah dapat melakukan koreksi *error* pada data yang terkena *burst error* dengan kemampuan mengatasi dibatasi.

Saat ini hanya ada tiga metode yang dapat digunakan untuk mendeteksi *error* pada CAN yaitu CRC, *bit monitoring* dan *bit stuffing*. Ketika *error* terdeteksi maka sistem akan melakukan ARQ dan mengirimkan kembali *frame* yang *error*. Efisiensi untuk *high speed* CAN mencapai 30 %.

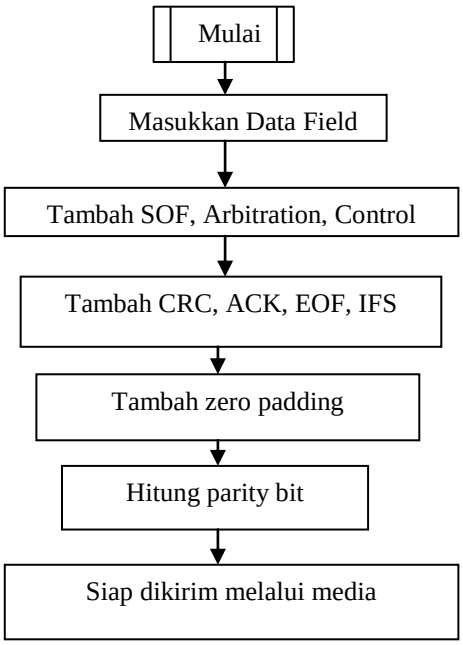
III. DESAIN PERANCANGAN

Gambar 1a menunjukkan tentang karakteristik sandi RS.

Gambar 1a. Karakteristik Simbol dalam sandi RS

Pada penelitian (Shabour, 2013) untuk RS (31,27) hanya disimulasikan dengan MATLAB, maka pada paper ini akan dilakukan contoh perhitungan matematis. Untuk skenario 1 digunakan m adalah 5 dan n adalah 31 (n merupakan panjang *codeword*), maka dipilih k = 27 (k merupakan panjang kata pesan atau informasi). Maka (n, k) = (31, 27). Karena akan di *encode* menggunakan k sebanyak 27 simbol maka akan diproses 27 simbol x5 bit = 135 bit. Maka pada CAN *frame* (yang memiliki panjang bit standard 111) ditambah 24 bit zero (dikenal dengan teknik *zero padding*) agar sama dengan 135 bit. Pada skenario ini akan diproses setiap satu simbol dari LSB (*Least Significant Bit*). Maka banyaknya simbol yang dapat dikoreksi ialah (31-27)/2 = 2 symbol (t=2) (Shabour, 2013)

.



Gambar 4. Diagram Alir

Dari diagram alir pada Gambar 4 dapat dijelaskan bahwa proses perhitungan dimulai dengan memasukkan *data field*. Kemudian ditambah *Start Of Frame* (SOF), *Arbitration*, dan *Control bit*. Pada bagian akhir *data field* ditambahkan *Cyclic Redundancy Check* (CRC), *Acknowledgement* (ACK), *End of Frame* (EOF) dan *Inter Frame Space* (IFS) bit. Kemudian ditambah *zero padding*. Penambahan *zero padding* ini mengikuti banyaknya simbol dari RS *code* yang digunakan. RS *code* yang digunakan pada penelitian ini adalah 27 simbol. Pada penelitian ini, skema yang diusulkan untuk mengurangi proses retransmisi ialah menggunakan sandi RS (31, 27). Sandi ini dapat mengkoreksi *error* hingga 4 *symbol* atau setara dengan 4x5 = 20 bit. Sehingga diharapkan dapat membantu mengurangi proses retransmisi. Jika proses retransmisi dikurangi maka akan menghemat daya pada pengirim. Pada CAN memiliki total bit 111 bit. Maka akan di encoding dengn 27x5 bit = 135 bit.

IV. HASIL SIMULASI DAN PEMBAHASAN

Untuk RS Code (31, 27) sudah ada hasil hingga perhitungan *syndrome*. Dengan nilai *integer* informasi ialah 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2. Hasil perhitungan pembagian antara $u(x).x^{n-k}$ dengan g(x). Notasi $u(x).x^{n-k}$ menunjukkan informasi yang akan dikirim. Sedangkan g(x) adalah *generator* polinomial. Untuk $u(x).x^{n-k} = \alpha^0 x^{30} + \alpha\, x^{29} + \alpha^{18} x^{28} + \alpha^2 x^{27} + \alpha^5 x^{26} + \alpha^0 x^{25} + \alpha\, x^{24} + \alpha^{18} x^{23} + \alpha^2 x^{22} + \alpha^5 x^{21} + \alpha^0 x^{20} + \alpha\, x^{19} + \alpha^{18} x^{18} + \alpha^2 x^{17} + \alpha^5 x^{16} + \alpha^0 x^{15} + \alpha\, x^{14} + \alpha^{18} x^{13} + \alpha^2 x^{12} + \alpha^5 x^{11} + \alpha^0 x^{10} + \alpha\, x^9 + \alpha^{18} x^8 + \alpha^2 x^7 + \alpha^5 x^6 + \alpha^0 x^5 + \alpha\, x^4$.

Dan nilai

$g(x) = (x-\alpha)(x-\alpha^2)(x-\alpha^3)(x-\alpha^4) = x^4 - (\alpha^{24})x^3 - (\alpha^{19})x^2 + (\alpha^{29})x + \alpha^{10}$

Maka didapatkan hasil pembagian yaitu

$\alpha^0 x^{26} + \alpha^{13} x^{25} + \alpha^{23} x^{24} + \alpha^7 x^{23} + \alpha^{22} x^{22} + \alpha^{24} x^{21} + \alpha^0 x^{20} + \alpha^{20} x^{19} + \alpha^{29} x^{18} + \alpha^{13} x^{17} +$

ISBN 978-602-99334-4-4

$\alpha^6 x^{16} + \alpha^{27} x^{15} + \alpha^{13} x^{14} + \alpha^{12} x^{13} + 0\, x^{12} + \alpha^8 x^{11} + \alpha^{24} x^{10} + \alpha^{12} x^9 + \alpha^5 x^8 + \alpha^{16} x^7 + \alpha^{13} x^6 + \alpha^5 x^5 + \alpha^6 x^4 + \alpha^{21} x^3 + \alpha^{19} x^2 + \alpha^9 x + \alpha^{23} x^0$.

Kemudian didapat sisa hasil bagi yaitu

$b(x) = \alpha^{15} x^3 + \alpha^9 x^2 + \alpha^{24} x + \alpha^2$.

Kemudian ditambah *error* pada *integer* 1 dan 2 paling kiri dari *integer* informasi (atau yang dicetak tebal). Informasi setelah terkena error **1 2** 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5 1 2.

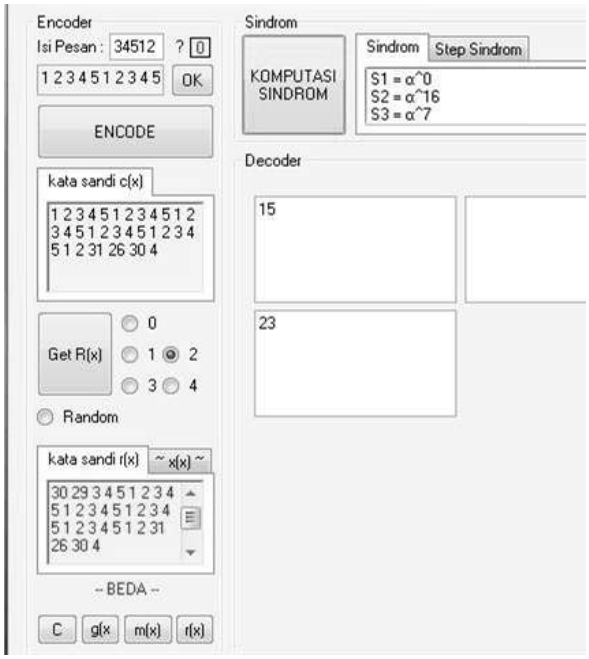
Kemudian perhitungan dilanjutkan dan didapatkan nilai *syndrome*.

$S_i(x) = u(x) \,.\, x^{n-k} + b(x)$.

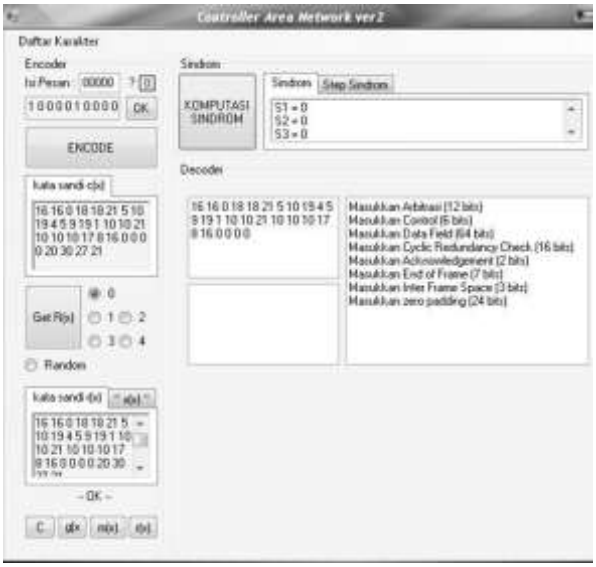
$\alpha^0 x^{26} + \alpha^{13} x^{25} + \alpha^{23} x^{24} + \alpha^7 x^{23} + \alpha^{22} x^{22} + \alpha^{24} x^{21} + \alpha^0 x^{20} + \alpha^{20} x^{19} + \alpha^{29} x^{18} + \alpha^{13} x^{17} + \alpha^6 x^{16} + \alpha^{27} x^{15} + \alpha^{13} x^{14} + \alpha^{12} x^{13} + 0\, x^{12} + \alpha^8 x^{11} + \alpha^{24} x^{10} + \alpha^{12} x^9 + \alpha^5 x^8 + \alpha^{16} x^7 + \alpha^{13} x^6 + \alpha^5 x^5 + \alpha^6 x^4 + \alpha^{21} x^3 + \alpha^{19} x^2 + \alpha^9 x + \alpha^{23} x^0 + \alpha^{15} x^3 + \alpha^9 x^2 + \alpha^{24} x + \alpha^2$.

Syndrome yang akan dihitung ialah $S_1(\alpha^1)$, $S_2(\alpha^2)$, $S_3(\alpha^3)$ dan $S_4(\alpha^4)$.

Penjelasan Gambar 5 adalah sebagai berikut. Pada kolom isi pesan dimasukkan 27 simbol dengan rentang 1 sampai 5. Kemudian tekan tombol OK. Maka akan ditampilkan nilai yang dimasukkan. Ketika tombol Encode ditekan maka akan dilakukan proses pembagian informasi yang dimasukkan dengan *generator polynomial* yang kemudian ditampilkan pada *box* kata sandi c(x). Kemudian dipilih angka 2. Maksudnya akan ada 2 error yang ditambahkan pada data yang dikirim yaitu pada posisi 1 dan 2. Kemudian ditekan tombol Get r(x). Maka akan muncul *codeword* yang telah ditambah dengan 2 *error* pada *box* kata sandi r(x). Kemudian setelah ditekan tombol C yang berarti tombol *compare* maka akan ada tulisan BEDA. Hal ini maksudnya adalah bahwa untuk membandingkan data kata sandi sebelum terkena *error* dan kata sandi yang telah terkena *error*. Kemudian setelah ditekan tombol KOMPUTASI SINDROM maka akan muncul hasil perhitungan sindrom yang akan ditampilkan pada *box* sindrom.



Gambar 5. Simulasi Encoder



Gambar 6. Simulasi Encoder dengan Input CAN

Penjelasan Gambar 6 adalah sebagai berikut. Pada kolom isi pesan dimasukkan 135 simbol dengan rentang 0 sampai 1. Untuk memasukkan 135 simbol ini perlu memperhatikan perintah yang ada di kolom kanan. Sehingga dapat diketahui bit apa yang akan dimasukkan. Kemudian tekan tombol OK. Maka akan ditampilkan nilai yang dimasukkan dan telah diubah ke simbol dengan tiap bit menjadi 1 simbol.

V. KESIMPULAN

Dari kedua penelitian diatas dapat disimpulkan bahwa ada beberapa metode untuk melakukan perbaikan *error*. Metode yang telah digunakan oleh peneliti terdahulu ialah metode ARQ, HARQ, CRC, *Cyclic code* dan RS *Code*. Pada analisis matematis ini dapat diambil kesimpulan bahwa pada keluaran dari *Encoder* adalah sindrom. Dengan masukan 27 simbol maka akan didapatkan 31 simbol pada keluaran *encoder* yang siap dikirim. Keluaran encoder inilah yang akan digunakan untuk menghitung sindrom. RS code (31, 27) ini dapat mengatasi error 2 simbol. Sehingga akan dapat mengatasi burst error. EMI juga akan bisa diatasi.

VI. DAFTAR PUSTAKA

Bleichenbacher, D., Kiayias, A. & Yung, M., 2007. Decoding interleaved Reed – Solomon codes over noisy channels. *Elsevier*, 379, pp.348–360.

Chang, H. & Shung, C.B., 1999. New Serial Architecture for the Berlekamp Massey Algorithm. *IEEE Communications, IEEE Transactions on Volume: 47 , Issue: 4*, 47(4), pp.481–483.

Chent, I. et al., 2011. An Error-Correction Scheme with Reed-Solomon Codec for CAN Bus Transmission. *International Symposium on Intelligent Signal Processing and Communication Systems*, pp.7–11.

Emani, K.C. et al., 2005. IMPROVEMENT OF CAN BUS PERFORMANCE BY USING ERROR-CORRECTION CODES. *Proceedings of IEEE Region 5 Technical Conference*, pp.205–210.

Fenn, S.T.J., Taylor, D. & Benaissa, M., 1995. The design of Reed-Solomon codecs over the dual basis. , 26, pp.383–391.

Jie, M., Min, S. & Min, Z., 2009. New Application of Reed-Solomon Codes in China Mobile Multimedia Broadcasting System. *IEEE Information Technology and Applications, 2009. IFITA '09. International Forum on*, pp.2–5.

Kumar, S., 2011. Bit Error Rate Analysis of Reed-Solomon Code for Efficient Communication System. *International Journal of Computer Applications*, 30(12), pp.11–15.

ISBN 978-602-99334-4-4

Ng, W.L. et al., 2010. Home Appliances Controller using Wireless Controller Area Network (WCAN) System. *International Conference on Computer and Communication Engineering*, (May), pp.11–13.

Rassouli, B. & Olfat, A., 2012. Spectrum sensing with energy detection under fast and slow multipath fading. *2012 19th International Conference on Telecommunications (ICT)*, (Ict), pp.1–5. Available at: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6221304.

Shabour, H.M., 2013. Performance Enhancement of the Controller Area Network Protocol Using Reed-Solomon Codes. *International Conference on Computing, Electrical and Electronic Engineering IEEE*, pp.512–517.

Smith, S., Taylor, D. & Benaissa, M., 1998. Design automation of Reed-Solomon codecs using VHDL. *Elsevier*, 29, pp.977–982.

Thale, S., Agarwal, V. & Ieee, S.M., 2011. Controller Area Network (CAN) based Smart Protection Scheme for Solar PV , Fuel Cell , Ultra-Capacitor and Wind Energy System based Microgrid. *IEEE Photovoltaic Specialists Conference (PVSC), 2012 38th*, (Ll), pp.580–585.