2nd International Conference on Engineering of Tarumanagara (ICET 2015)
Faculty of Engineering, Tarumanagara University, Jakarta-Indonesia, 22-23 October 2015
ISBN 978-602-71459-1-7

ANALYSIS, SIMULATION AND IMPLEMENTATION OF LINEAR BLOCK CODES USING A MICROCONTROLLER

Joni Fat[1)]

[1)]Tarumanagara University, Jl. Let. Jend. S. Parman No. 1, Jakarta Barat - 11460, Indonesia

e-mail: jonif@untar.ac.id

Abstract

Linear Block Code (LBC) is a methode which is used to detect and correct an error in data transmission through communication channel. In this research, LBC was simulated by using two software simulators and then was implemented in a hardware which was based on microcontroller. We have tested these simulators and device. We concluded that they could run well. The bit transfer rate in the hardware device is 2 bit/s. Since this is a low rate, but it is needed in order that user could manipulate error manually in the time of data transfer. This device also had another limitation, i.e. the type of data error. The type of data error that could be made during data transfer limited only to the flipping from 1 to 0. Implementation by using ATMEGA 8 gave another limitation which was the maximum size of G matrix. So, user could only input the size of G matrix as large as m=10 and n=10. This limitation was caused by the size of data memory in ATMEGA 8 which was 2 KB. We could solve this particular problem by using external memory or using a higher microcontroller specification.

Keywords: ATMEGA 8, error correction and detection, LBC, microcontroller, simulator.

INTRODUCTION

The discovery of a vacuum tube marked a new era in modern computing. John Mauchly started the design of a general-purpose computer using the vacuum tubes. The next era was marked by the discovery of a transistor. The advantages of transistor are smaller, cheaper and more efficient in power usage compared to the vacuum tube. It can be said that invention of the transistor was the one that started modern era in the design of computing machines. The invention of the transistor allowed an integrated circuit (IC) to emerge, because such components can be fabricated directly from a semiconductor material such as silicon. As a result, the components could be developed from a very thin wafer form. This technology allowed the integration of the components in a very small size and also compact. This is the IC.

In this current era and the incoming era, IC design challenges still remain in the same issues, such as size, speed, and density. Only because of the limitations of the most fundamental — in this case — the matter itself, then by the development of fabrication techniques which have reached the size of nanometer, will meet its own limitations. This limitation will lead to the issue of reliability. Energy consumpption will reduce, but intrinsically become unreliable. Therefore, we need a new error handling method, effective and efficient. This new error handling method is generally known as Linear Block Codes (LBC).

INFORMATION THEORY

Information theory was first introduced by Claude E. Shannon in The Bell System Technical Journal under the title of *A Mathematical Theory of Communication* in 1948. Shannon is one of the engineers in Bell Telephone Laboratories, hence, this work, although was focused on information, it was understandable that it was meant to telephony system. By Weaver, this concept was developed that it could be applied to all communication system.

Furthermore, with the development of today communication technology and the development of computing technology, the boundaries between communication and computing itself has been converged. Thus, a form of communication that looks like using telephone lines, computer networks, cellular networks, and so on, has been distorted. It is like communication that occurs in a chip or electronic circuit. This form of communication is simpler, but requires a higher accuracy. Because the interpretation is done in the lowest level. However, Shannon's thoery is still relevant because it provides some rules and restrictions that can be followed.

In some cases, this theory requires adjustment, such as the need for a feedback. Based on Shannon and Weaver's paradigm for point to point communication[1], changed the paradigm of public communication at that time. The paradigm adds an observer which is capable of providing data correction when an error occurs. Observer in this case could be codec mechanism with a correction capability. This data correction capability needs to be done with a method that is concise and precise and do not overload the system. The candidate methods that can be used further - but requires further research - is Low Density Code (LPDC)[2]. This code is part of a block of code that is widely used in channel coding. With a low data density, but could achive Shannon's limit.

LBC

GENERATOR MATRIX

Linear code is a vector space, where each code word is a vector. Thus, the set of vectors with n-length is called LBC if and only if the set is a subspace of a vector space n-tuples. Matrix representation of this code is an ideal way in order to describe it. A linear code with a size of (n, k) is expressed by the matrix generator, **G** with dimension of k x n. Each line **G** is an **n**-tuple, and each column is a **k**-tuple.

Therefore, row space of matrix **G** is a set of base vectors for **k**-dimensional subspace. Every code word, **c** is a linear combination of the rows of **G** based on information data $\mathbf{d} = (d_0, d_1, \ldots, d_{k-1})$, then

$$\mathbf{c} = \mathbf{d}_0\mathbf{g}_0 + \mathbf{d}_1\mathbf{g}_1 + \ldots + \mathbf{d}_{k-1}\mathbf{g}_{k-1} \quad (1)$$

with $\mathbf{d}_i$ $(0 < i < k-1)$, represent bits of information; and $\mathbf{g}_i$ $(0 < i < k-1)$, is a row vector of **G**. Encoding procedure can be represented in matrix form as follows:

$$\mathbf{C} = \mathbf{d} \, . \, \mathbf{G} \quad (2)$$

with $\mathbf{G} = [\mathbf{P}_{k \times (n-k)} \mid \mathbf{I}_k]$. The matrix **G** is called a generator matrix of a systematic code.

PARITY-CHECK MATRIX

Matrix **I** is the parity-check matrix for the generator matrix, **G**. Matrix **H** is a matrix with a dimension of (n-k) x n, such that $\mathbf{c.H}^T = \mathbf{0}$ with 0 indicating all zeros with a membership of n-k.

This equation can be used to prove the validity of the vector **G**, namely $\mathbf{G.H}^T = 0$ with 0 is a zero-dimensional matrix membered k x (n-k). For the matrix **G** such as in the above equation, the matrix **H** can be formulated as follows:

$$\mathbf{H} = [\, \mathbf{I}_{n-k} \mid \mathbf{P}^T_{(n-k) \times k} \,] \quad (3)$$

$\mathbf{P}^T$ is the matrix transpose of the submatrix **P** from **G**.

SYNDROME

Let $\mathbf{c} = (c_0, c_1, \ldots, c_{n-1})$ is a code word that is transmitted and $\mathbf{r} = (r_0, r_1, \ldots, r_{n-1})$ word that is received at the demodulator output. Word **r** can be the same or different from **c**, depending on the noise in the channel. If $\mathbf{r} \neq \mathbf{c}$, it can be corrected by using equation error, with **e** = **r** + **c** = (e0, e1, ..., en-1). Word **e** is called error.

After receiving **r**, the decoder starts counting syndrome in order to locate errors, and then corrects them. Syndrome is denoted by **s**:

$$\mathbf{s} = \mathbf{r} \,.\, \mathbf{H}^T = (s_0, s_1, \ldots, s_{n-k-1}) \tag{4}$$

Therefore, the sum of the vector **c** and **e**, then the equation can be replaced by:

$$\mathbf{s} = (\mathbf{c} + \mathbf{e})\, \mathbf{H}^T = \mathbf{c} \,.\, \mathbf{H}^T + \mathbf{e} \,.\, \mathbf{H}^T \tag{5}$$

with **c** . $\mathbf{H}^T$ = **0**, thus the equation can be written simply be:

$$\mathbf{s} = \mathbf{e} \,.\, \mathbf{H}^T \tag{6}$$

This equation shows the relationship between syndrome and error. So, if **s** = **0**, then it is certain that **e** = **0** or no error. But if **s** ≠ **0**, then **r** ≠ **c**, which means there error.[3]

SIMULATOR

HARDWARE

For this simulator, there are two hardware design, i.e. transmitter and receiver. The transmitter contains program that could process matrix **G** and user's input and then transmits matrix **c** to the other. The receiver receives matrix **G**, process it into matrix **H** and then receives matrix **c**, and also performs error detection and correction. Both of these devices have an identical specification for hardware, but different in the program. Another difference is the position of wireless transmitter and receiver modules. The position of the modules on each device is placed in an order that the wireless transmitter module in one device is faced to wireless receiver module in another device, and vice versa.

Processing module consists of an ATMEGA8 microcontroller. This module uses an [illegible] MHz crystal oscillator. For ATMEGA 8, the available ports are port A, B, C, and D. But the ports that can be used for I/O are port B, C and D. This microcontroller also provides 1 KB Read Access Memory (RAM), and 8 KB Read Only memory (ROM).

The design of this processing module can be seen in Figure 1. External power supply which is used is an adapter that produces an output voltage of 5 VDC with a source of 220 VAC.

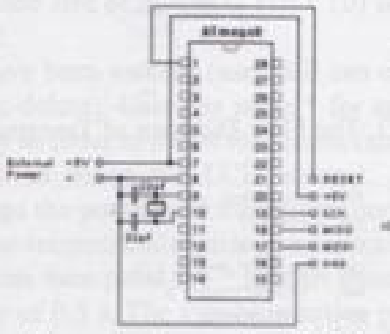


Figure 1. Processing Module

SOFTWARE

TRANSMITTER

Figure 2 shows a flow diagram for transmitter. When the transmitter is turned on, transmitter will wait for input. First, user should input the size and the values of the matrix G. Figure 3 shows part of program that runs in the transmitter. It appears that there are some subroutines that have been defined in advance, such as subroutines in row **100** ([illegible] ()), the subroutine in row **160** (**Konversi_Biner (m, M)**), the subroutine in row **230** ([illegible] ()), and subroutine in row **250** (**Tampil_Matrix (m, n, g, G)**). These subroutines are named in accordance with their respective functions. Thus, with just reading the subroutine names, their functions could be understood. For example, subroutine **Input_MatrixG ()** is a procedure which is requesting input from the user for the values of the matrix **G**.

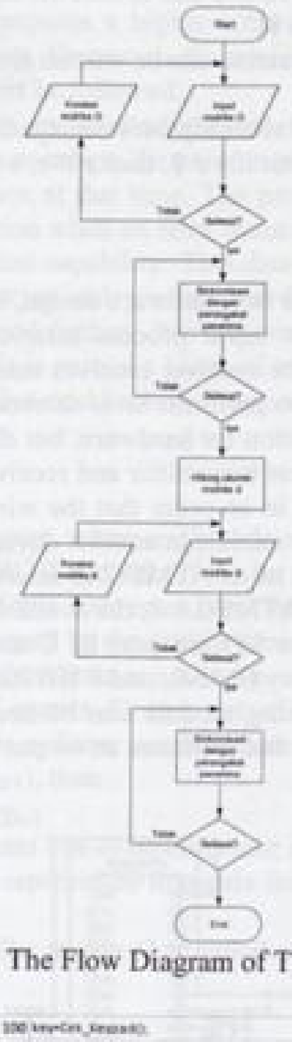
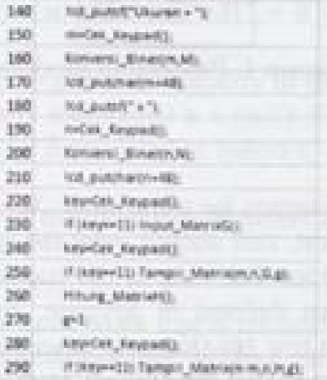


Figure 2. The Flow Diagram of Transmitter

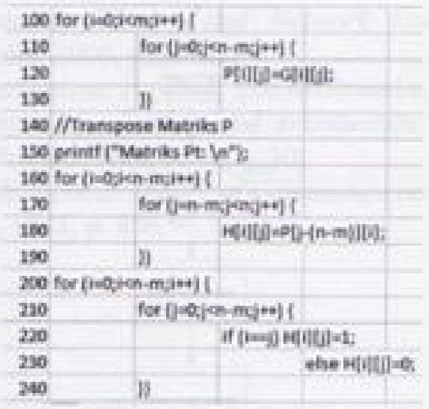


```
100 key=Cek_Keypad();
110   if (key==11) {
120       lcd_clear();
130       lcd_puts("  MATRIKS  G  ");
140       lcd_puts("Ukuran = ");
150       m=Cek_Keypad();
160       Konversi_Biner(m,M);
170       lcd_putchar(m+48);
180       lcd_puts(" x ");
190       n=Cek_Keypad();
200       Konversi_Biner(n,N);
210       lcd_putchar(n+48);
220       key=Cek_Keypad();
230       if (key==11) Input_MatrixG();
240       key=Cek_Keypad();
250       if (key==11) Tampil_Matriks(m,n,G,g);
260       Hitung_MatrixH();
270       g=1;
280       key=Cek_Keypad();
290       if (key==11) Tampil_Matriks(n-m,n,H,g);
```

Figure 3. Program that Runs in the Transmitter

RECEIVER

After synchronization, the values of matrix **G** could be found in array variable **G [10][10]**. Matrix **G** will be converted into the matrix **H** in accordance with rules that has been defined. In this program, the conversion process can be read in Figure 4.



```
for (i=0;i<m;i++) {
        for (j=0;j<n-m;j++) {
                P[i][j]=G[i][j];
        }}
//Transpose Matriks P
printf ("Matriks Pt: \n");
for (i=0;i<n-m;i++) {
        for (j=n-m;j<n;j++) {
                H[i][j]=P[j-(n-m)][i];
        }}
for (i=0;i<n-m;i++) {
        for (j=0;j<n-m;j++) {
                if (i==j) H[i][j]=1;
                        else H[i][j]=0;
        }}
```

Figure 4. Matrix G to H Conversion Program

Figure 5 shows flow diagram for receiver. Whe matrix H is formed, system will form Coset table.

RESULTS AND DISCUSSION

Once the program is downloaded to the transmitter and receiver, on the transmitter LCD screen will appear "LINEAR BLOCK CODE" in the first line and "TRANSMITTER MODULE" on the second line. System will be in stand-by state until the user presses the * button.

Then system will prompt user to enter size of the matrix **G**. When entered, LCD screen will go blank. In this position, user should input values of matrix **G** per line. Array is used in this program to define size of matrix **G** (10 x 10) so it defines the maximum size of matrix **G** which is allowed.

When these values have been entered (user still can edit them using arrow keys on the keypad and press the # to delete). User can press * for entering values and rechecking the matrix values with arrows in order to move the screen (size LCD screen is only 2 lines, therefore arrows are needed to move the LCD screen). After checking procedure is completed, user should arrange the position of transmitter and receiver modules so they are in the line of sight (LOS). The recommended distance for transmitter and receiver modules are between 5-10 cm. User can then press the * to start synchronization process. Each bit will be sent with a time delay of 0.5 s. The synchronization process is completed when the LCD screen shows an instruction to enter values of matrix **d**, and LCD screen in receiver shows "MODULE RECEIVER."

User can enter the value of matrix **d** and then press * in transmitter to initiate the process of encoding and sending data. When user wants to make an error, it could be done by presenting an object between transmitter and receiver modules LOS for an interval approximately 0.5 s. Once the data is received by receiver, system will calculate matrix **H**, coset table and matrix **s**.

Detection and correction process will be done accordingly, so that when there is an error, it will be immediately corrected by system. Testing without error shows that system could be perform well. For testing with error, for some carefully selected cases, shows that

ISBN 978-602-71459-1-

system also well perform. Therefore it can be concluded that for reception conditions with or without an error, the system has also been able to work well.

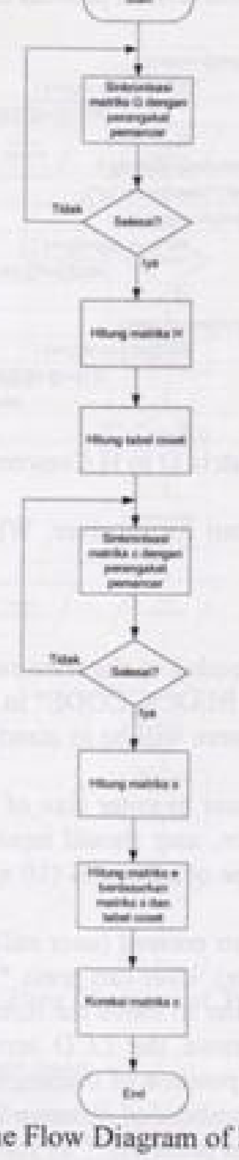

Start
Sinkronisasi matriks G dengan perangkat pemancar
Tidak
Selesai?
Ya
Hitung matriks H
Sinkronisasi matriks x dengan perangkat pemancar
Tidak
Selesai?
Ya
Hitung matriks s
End

Figure 5. The Flow Diagram of Receiver

CONCLUSIONS

ATMEGA8 microcontroller is a microcontroller which is produced by AVR. For ATMEGA family, ATMEGA 8 has the lowest specification. From the tests carried out in this research, it can be concluded that microcontroller with limited capacity is also able to handle the process of sending, receiving, detection and correction of data properly. Even with a memory that only 2 KB for data, system is able to handle various kinds of data which are necessary for process, although some adjustments are necessary.

In the transmission process, data are sent wirelessly between transmitter and receiver modules. Each bit of data is sent with the interval of 0.5 s. This means that the data transfer rate for this process is 2 bps. For data transfer, a value of 2 bps is very low. Although, test showed that system was capable of handling all processes (send, receive,

ISBN 978-602-71459-1-7

detection and correction) with no errors.

REFERENCES

[1] C.E. Shannon and W. Weaver, *The Mathematical Theori of Communication*, Urbana: The University of Illinois Press, 1964, pp. 34, 68.
[2] R.W. Yeung, *Information Theory and Network Coding*, New York: Springer Science+Business Media, 2008, p. 166.
[3] *Help Matlab: Communications Toolbox*, The Mathworks, 2004.
[4] W. Stallings, *Computer Organization and Architecture Designing for Performance*, Upper Saddle River: Prentice Hall, 2010, p. 30.
[5] C. E. Shannon, *A Mathematical Theory of Communication*, The Bell System Technical Journal, Vol. 27, July, 1948, p. 406.