Faculty of Electrical Engineering
Universiti Teknologi Malaysia

VOL. 9, NO. 2, 2007, 49-55

ELEKTRIKA
http://fke.utm.my/elektrika

Another Perspective in Generating and Using Gray Code-word

A. Ahmad

Department of Electrical and Computer Engineering, College of Engineering, Sultan Qaboos University
P. O. Box 33, Postal Code 123; Muscat, Sultanate of Oman

*Corresponding author: afaq@squ.edu.om (A. Ahmad), Tel: 968-24415327, Fax: 968-24413454

Abstract: A newly innovated set of binary code-words of unitary Hamming distance is proposed. The design of generating the code using VHDL, hardware and algorithmic implementation procedures are embodied in this paper. Also, a comparison study is carried-out to demonstrate the edge-over improvement and added advantages over the binary Gray code-words. Further, since the existence of latches is the fact of any digital system design and by exploitation of these latches in generating the Gray type codes hence reduces the extra burden of hardware to generate the 'true' Gray code.

Keywords: Binary code, Gray code, Hamming distance, Shift register, Transition counts, VHDL

1. INTRODUCTION

The Reflected Binary Code (RBC) which was an original name of Gray Code (GC) is derived from the stem of the fact that the second half of the values are equivalent to the first half in reverse order, except for the most significant bit, which is inverted i.e. 0 changes to 1. The Reflected Binary code was later named after the researcher Frank Gray of Bell Labs., who patented this code by using it for shaft encoders in 1953 [1]. Gray codes were originated when digital logic circuits were built around vacuum tubes and electromechanical relays. And thus, counters were generating enormous power demands and noise spikes when many bits changed at the same time [2].

A Gray code is a binary code with code-words having unitary Hamming distance between each other. Table 1 lists all the possible Gray-code patterns of lengths 1, 2, 3, and 4 bits and shows the above mentioned property of unitary distance. A Gray code of length n bits is represented by all the integers from 0 to (2^n-1) in the forms of bit patterns (called "words") of length n having the adjacency property (i.e. every pattern differs in only one bit position from each of the two patterns corresponding to the two adjacent integers). Therefore, by this way, using counters based on Gray code, where any increment or decrement in the value changes only one bit, irrespective of the size of the counter, hence minimizing the effect of noise [2 -3].

Among the many other binary codes such as the Binary Coded Decimal (BCD), the Excess-3 code, the Hamming code, or the Cyclic Redundancy Code (CRC) existing in the literature, due to its attribute of unitary distance which can avoid ambiguous switching situations, the Gray code is particularly suited to handle control problems in a robust and convenient manner. Unlimited applications are accounted for the Gray codes.

Table 1. Gray code patterns of lengths1, 2, 3, and 4 bits.

Decimal	Binary code	Gray code 4-bit	Gray code 3-bit	Gray code 2-bit	Gray code 1-bit
0	0000	**0**000	**0**00	**0**0	**0**
1	0001	000**1**	00**1**	0**1**	**1**
2	0010	00**1**1	0**1**1	**1**1	
3	0011	001**0**	01**0**	1**0**	
4	0100	0**1**10	**1**10		
5	0101	011**1**	11**1**		
6	0110	01**0**1	1**0**1		
7	0111	010**0**	10**0**		
8	1000	**1**100			
9	1001	110**1**			
10	1010	11**1**1			
11	1011	111**0**			
12	1100	1**0**10			
13	1101	101**1**			
14	1110	10**0**1			
15	1111	100**0**			

Mechanical position sensors use Gray code to convert the angular position (angle-measuring devices) of a shaft to digital form. Gray codes were used in telegraphy [1-4]. The Gray code also forms a Hamiltonian cycle on a hypercube, where each bit is seen as one dimension [5-7]. In data transmission, Gray codes are capable of providing more robust communication and simultaneously play an important role in error detection and correction of errors [8-9]. Other processes where Gray codes are successfully employed include the

solution of puzzles such as the "Tower of Hanoi" [10] and "the Brain", the study of bell-ringing [11], analog-to-digital conversion [12], the classification of Venn-diagrams [13], continuous space-filling curves, resolution enhancement of spectrometry for satellite applications, and the labeling of axes of Karnaugh maps [14–18]. Gray codes are also beneficial in Genetic Algorithms due to the property of their code-words to change incrementally [19].

Also perspective uses of Gray codes while addressing the memory results in power-savings because few address lines change as the program counter advances to the next location. Furthermore, Gray codes are extensively used by digital system designers for passing multi-bit count information between synchronous logic that operates at different clock frequencies. In some numerical problems, Gray codes can be useful in situations of looping over many values of a bit. Finally, due to their unique properties Gray codes can be a good choice in the search for optimal test-sequences in digital system testing.

2. GRAY CODE APPLICATIONS - EXAMPLE

Table 2 (adapted from [20]) illustrates how the Towers of Hanoi can be solved using Gray codes. In Table 1, each bit in the three-bit binary number in the Bit string column corresponds to one of the three discs to be moved, with bit *j* representing disc *j*. The convention here is that the smallest disc is disc 1 (the least significant bit), and the largest is disc 3 (the most significant bit). The sequence is initialized to 000, and a change in the *j*th bit from one binary code to the next represents a move of disc *j*. For example, the flipping of bit 1 from 000 to 001implies that disc 1 is to be moved, whereas the flipping of bit 2 from 111 to101 implies that disc 2 is to be moved. Hence, the binary reflected Gray code 2 sequence represents a sequence of single disc moves that can be used to solve the Towers of Hanoi problem.

Table 2. Application of Gray Codes to the Towers of Hanoi Problem

Bitstring n···321	Towers of Hanoi
000	
001	
011	
010	
110	
111	
101	
100	

Considering, another practical example, of the hypothetical rotation of the device (disc) as shown in Figure 1. This rotating situation may occur as the drum in a photocopy machine or a magnetic storage device or the axle of a vehicle, both real examples of the use of this idea. The knowledge of actual physical locations is necessary as the device turns to rotate. Let the device contains a number of tracks (rings) each of which is broken down into sectors (pieces of pie). Furthermore, consider there is a fixed set of sensors, one per track. Each (track, sector) pair stores either a 0 or a 1. In the figure there are four tracks and constituting 16 sectors. In general it can be formulated that if there are n tracks then there will be a 2^n sectors. Assume a white (track, sector) pair as a 0 and a black colored shaded area as a 1. Since each sector intersects n tracks, it corresponds to some binary number in the range of 0 to $2^n - 1$, a number that should be unique in order to unambiguously identifying the sector. In what order should those binary numbers be arranged around the sectors requires help of Gray code. The main problem with binary system that using binary code in tracks are that there are many positions where several tracks change state at same time. This may result in error. Actually in Gray code only one track can change at same time during rotation. So then if error occurs, the resulting error will be only one bit. Gray code facilitates an easy correction through a microcontroller using a lookup table.

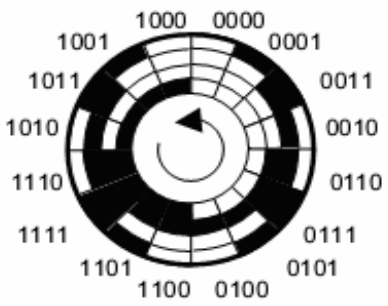

Figure 1. A 4-bit Gray code encoder disc

A more elaborated image with complete arrangement is shown below in Figure 2 where, a 5-bit rotation encoder disc with rotating arrangement via shaft, using a Gray-code pattern and an optical sensor arrangements.

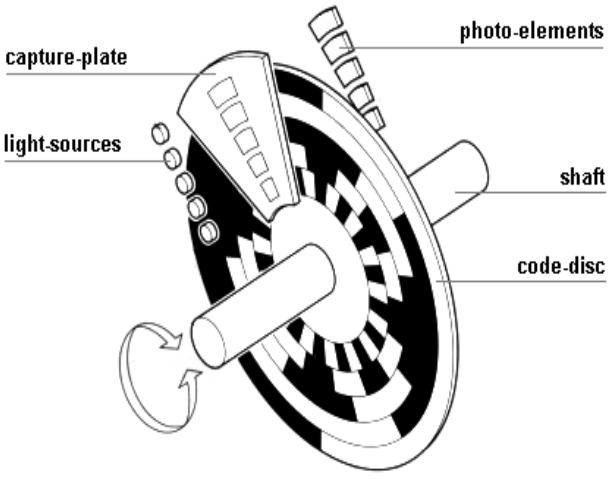

Figure 2. A 5-bit rotation encoder, using a Gray-code pattern and an optical sensor

3. ANOTHER PERSPECTIVE

Maintaining the unitary Hamming distance between each of the predecessor and successor binary code-words C(*i*) and C(*i*+1) means differing at exactly only one bit position. For example let's consider a 2-bit size such binary code-words [C(1), C(2), C(3), C(4)] are [00, 01, 11, 10].

The Gray code which is weighted and cyclic in nature came into existence at the time when digital logic circuits were built from vacuum tubes and electromechanical relays. The time at which counters dissipating enormous power and generating noise spikes when many bits changed at once.

Using binary code-words with unitary Hamming distance (Gray code counter), where the binary counter increment or decrement in the value changes only one bit, regardless of the size of the number, thereby minimizing the effect of noise and power dissipation as well. However, we further investigated the property of Gray codes and found that the code-words it self can be improved while maintaining the unitary Hamming distance between the predecessor and successor code words. Our proposed code-words have constant transition counts, reducing the power dissipation and also cutting short the noise spikes. It has been investigated [21] that the conventional method of generating Gray codes requires more time and enormous silicon area. The reason behind this is due to the methodology of generating the Gray codes where, the binary codes are first generated and then to be converted into the Gray codes using the conventional recursive equations. Therefore, the conventional Gray code generators require the 2^n x n-bit binary generator followed by a circuit of n x 2^n binary to Gray code converter. To avoid this requirement of large silicon area due to implementation of Gray code generator circuit, another idea is presented to achieve the same purpose of the Gray code applications without the requirements of any extra silicon area. This is the main thrust of this paper.

The idea of presenting this work stems from the fact that the shift registers are the built-in parts of any digital system. Just chaining the last and the first flip-flop will generate the Gray type code with better properties of transitioning hence less power dissipation and faster processing.

The ensuing section discusses about that how this idea has been grasped and modeled. Further, the idea of its implementation using the available circuits of shift registers is elaborated Section V. The faster algorithmic generation of the proposed Gray type code is also described in this section. The demonstration of the claim of better transitioning properties of these newly innovated Gray type codes is presented in Section 6.

4. PROPOSED TECHNIQUE

Through analyzing the bit-patterns of Gray code we realized that why not to write the code-words in the following patterns to minimize the transition while retaining the property of unitary Hamming distance. Let's consider the bit size of the code-word as n then the proposed pattern generation scheme can be expressed in the form of a theorem (Proposed Theorem 1) as given below.

Proposed Theorem 1:

For an *n*-bit size of code there can be 2n binary code-words where each of the predecessor and successor binary code-words has unitary Hamming distance.

Figure 3 below constructs the code-word vector C which will be of size 2*n* x *n*. The code-word pattern can be represented by the each row vector of the following matrix.

$$C = \begin{bmatrix} 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & \cdots & 1 & 1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \cdots & \vdots & \vdots & \vdots \\ 1 & 1 & 1 & 1 & \cdots & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & \cdots & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & \cdots & 1 & 0 & 0 \\ \vdots & \vdots & \vdots & \vdots & \cdots & \vdots & \vdots & \vdots \\ 1 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \end{bmatrix}$$

Figure 3. An *n*-bit size code-words generation matrix

In general, to script all the 2*n* code-words, the following equations are suggested.

The *i*th code-word (where, *i* varies from 1 to *n*) scripting is governed by Equation (1) whereas; to script the rest (*n*-1) code-words Equation (2) is to be used. The Equations (1) and (2) are as given below.

$$(2^{i-1} - 1)_{base2}; \text{ where } i \text{ varies from 1 to } n. \quad (1)$$

$$(2^n - 2^j)_{base2}; \text{ where } j \text{ varies from 1 to } n\text{-}1. \quad (2)$$

Example 1:

Let's consider n = 4, then the first five code words can be written using the Equation (1) by inserting the subsequent values of i from 1, 2, 3, 4, and 5.

$C(1) = [0\ 0\ 0\ 0]_{1x4}$
$C(2) = [0\ 0\ 0\ 1]_{1x4}$
$C(3) = [0\ 0\ 1\ 1]_{1x4}$
$C(4) = [0\ 1\ 1\ 1]_{1x4}$
$C(5) = [1\ 1\ 1\ 1]_{1x4}$

Using Equation (2) and subsequently inserting values of *j* from 1, 2, and 3 we get

$C(6) = [1\ 1\ 1\ 0]_{1x4}$
$C(7) = [1\ 1\ 0\ 0]_{1x4}$
$C(8) = [1\ 0\ 0\ 0]_{1x4}$

5. SIMULATION AND HARDWARE IMPLEMENTATION

To prevent multiple bits transitioning simultaneously, we can use a Gray code to ensure the property of unitary Hamming distance that only a single bit transitions at a time is maintained through out amongst the predecessor and its successor code-words. However, analyzing the

Gray code-words we found that the property of single bit transitioning is unavoidable in some of the code-words itself. Let's consider the conversion of binary code $(011)_2$ into the Gray code, results $(010)_{GRAY}$ which has the transition count as 2 but not the 1.

An alternative approach is suggested to meet out the objectives of Gray codes along with the additional property of maintaining the unitary transitioning amongst the bits of the code-words itself. Such types of code-words can be generated simply by using Linear Feedback Shift Register (LFSR) circuits.

The proposed implementation methodology for generating the suggested code-words of size *n* is depicted in Figure 4. The sequential machine shown in Figure 4 generates the suggested cyclic codes with its pre-determined code-word either all 0s or all 1s which can be tuned by either the RESET or the SET pins respectively.

The characteristic equation of this proposed sequential circuit which computes the successive code-words i.e. the next code vector [Q_n (t+1) Q_{n-1} (t+1) . . . Q_2 (t+1) Q_1 (t+1)] from its present code vector [Q_n (t) Q_{n-1} (t) . . . Q_2 (t) Q_1 (t)] can be given as below (see Equation 3).

$$\begin{bmatrix} Q_n(t+1) \\ Q_{n-1}(t+1) \\ \vdots \\ Q_j(t+1) \\ \vdots \\ Q_2(t+1) \\ Q_1(t+1) \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & \vdots & 0 & 0 & 0 \\ 0 & 0 & 1 & \vdots & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & \vdots & \vdots & 0 \\ 0 & 0 & 0 & 0 & 1 & \vdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & \vdots & 0 & 1 & 0 \\ 1 & 0 & 0 & \vdots & 0 & 0 & 0 \end{bmatrix} * \begin{bmatrix} Q_n(t) \\ Q_{n-1}(t) \\ \vdots \\ Q_j(t) \\ \vdots \\ Q_2(t) \\ Q_1(t) \end{bmatrix} \quad (3)$$

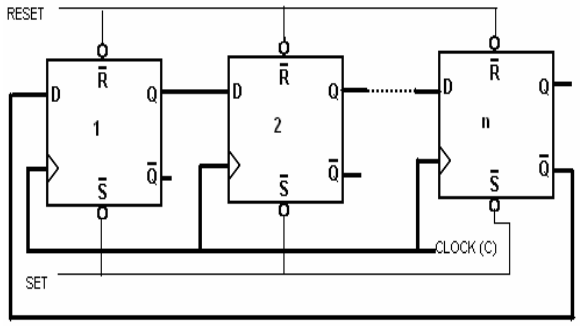

Figure 4. A proposed n-bit LFSR circuit

The simulation results for n = 2 to 9 is shown through Table 3. We can visualize from the tables that the generated code-words from the circuit of Figure 4 maintains "cyclic" and "adjacency (unitary Hamming distance)" properties of the Gray code-words while also ensuring the unitary transitioning between the bits of a code-word itself – which is an added value to the Gray type of code-words.

The implementation of the proposed code generation can be done either using the simulation or the hardware. The proposed hardware implementation for generating the code is much simpler than the Gray code generation schemes.

The following is the VHDL code used to simulate the *n*-bit code of the proposed scheme.

The VHDL code :

```
entity My Counter n is
  port(clock : in bit ; q :out bit vector (n downto 1)) ;
end ;
architecture RTL of My Counter n is
 signal My Counter : bit vector (n downto 1)
begin
 process(clock)
 begin
   if clock'EVENT and clock='1' then
     My Counter(1) <=not(My Counter (n)) ;
     LFSR(n downto 2) <= My Counter(n-1 downto 1) ;
  end if ;
  q<= My Counter,
 end process ;
end ;
```

To illustrate the shift register based hardware design of proposed code for an 8-bit size generator is considered. A chip, of 8-bit shift register with output register (IC # 74HC/HCT594) can be used having its functional circuit as shown in Figure 5. In the figure Q0 to Q7 represents the parallel data output lines while Q7' is meant for serial data output. The pin SHR (active LOW) resets shift register. Pin SHCP is the clock input of the shift register. The pin STR active (LOW) resets storage register, STCP 12 is responsible for clock input of storage register, while Ds pin is meant for serial data input. The GND (0 V), and VCC supply voltage are to be connected at pins 8 and 16 respectively. First by resetting the register, and short circuiting the pins 9 and 15, we obtain the following data (see Table 5) through the pins of Q0 to Q7.

6. ANALYSIS OF THE PROPOSED CODE-WORDS

We simulated the binary Gray-code words and the proposed binary code-words for large values of *n*. The Tables 4 and 5 show respectively the binary Gray code-words and proposed binary code-words for n = 2 to 6. Since there are 2*n* code-words can be generated using the proposed methodology and thus, only 2*n* Gray code-words are considered in the Table 4.

Transition count is one of the measures which reflect the noise spike, power dissipation and the propagation delay in the code-word signal itself. The Transition count is referred to number of times a signal changes from 0 to 1 or from 1 to 0 during the propagation of the signal. Refer to the Figure 3, where it can be seen that the each row has the constant transition count as only one or zero but certainly not more than one. Since each row vector represents the separate code-words where each of the subsequent code-words maintains the property of the unitary Hamming distance as of the property of the Gray code-words. Therefore, the transition counts for any code-word will be computed due to the recording of the changes from 0 to 1 or from 1 to 0 in row vector of Figure 3.

For a sequence vector, R = [r_1, r_2, … , r_m], the associated Transition count can be given by

$$TC(R) = \sum_{i=1}^{m-1} (r_i \oplus r_{i-1}) \quad (4)$$

where $\oplus$ denotes modulo-2 addition.

Table 3. The generated Gray type code-words for n = 2-9.

2-bit size

Q_1	Q_o
0	0
0	1
1	1
1	0

3-bit size

Q_2	Q_1	Q_o
0	0	0
0	0	1
0	1	1
1	1	1
1	1	0
1	0	0

5-bit size

Q_4	Q_3	Q_2	Q_1	Q_o
0	0	0	0	0
0	0	0	0	1
0	0	0	1	1
0	0	1	1	1
0	1	1	1	1
1	1	1	1	1
1	1	1	1	0
1	1	1	0	0
1	1	0	0	0
1	0	0	0	0

6-bit size

Q_5	Q_4	Q_3	Q_2	Q_1	Q_o
0	0	0	0	0	0
0	0	0	0	0	1
0	0	0	0	1	1
0	0	0	1	1	1
0	0	1	1	1	1
0	1	1	1	1	1
1	1	1	1	1	1
1	1	1	1	1	0
1	1	1	1	0	0
1	1	1	0	0	0
1	1	0	0	0	0
1	0	0	0	0	0

7-bit size

Q_6	Q_5	Q_4	Q_3	Q_2	Q_1	Q_o
0	0	0	0	0	0	0
0	0	0	0	0	0	1
0	0	0	0	0	1	1
0	0	0	0	1	1	1
0	0	0	1	1	1	1
0	0	1	1	1	1	1
0	1	1	1	1	1	1
1	1	1	1	1	1	1
1	1	1	1	1	1	0
1	1	1	1	1	0	0
1	1	1	1	0	0	0
1	1	1	0	0	0	0
1	1	0	0	0	0	0
1	0	0	0	0	0	0

8-bit size

Q_7	Q_6	Q_5	Q_4	Q_3	Q_2	Q_1	Q_o
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1
0	0	0	0	0	0	1	1
0	0	0	0	0	1	1	1
0	0	0	0	1	1	1	1
0	0	0	1	1	1	1	1
0	0	1	1	1	1	1	1
0	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	0
1	1	1	1	1	1	0	0
1	1	1	1	1	0	0	0
1	1	1	1	0	0	0	0
1	1	1	0	0	0	0	0
1	1	0	0	0	0	0	0
1	0	0	0	0	0	0	0

9-bit size

Q_8	Q_7	Q_6	Q_5	Q_4	Q_3	Q_2	Q_1	Q_o
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1
0	0	0	0	0	0	0	1	1
0	0	0	0	0	0	1	1	1
0	0	0	0	0	1	1	1	1
0	0	0	0	1	1	1	1	1
0	0	0	1	1	1	1	1	1
0	0	1	1	1	1	1	1	1
0	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	0
1	1	1	1	1	1	1	0	0
1	1	1	1	1	1	0	0	0
1	1	1	1	1	0	0	0	0
1	1	1	1	0	0	0	0	0
1	1	1	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0

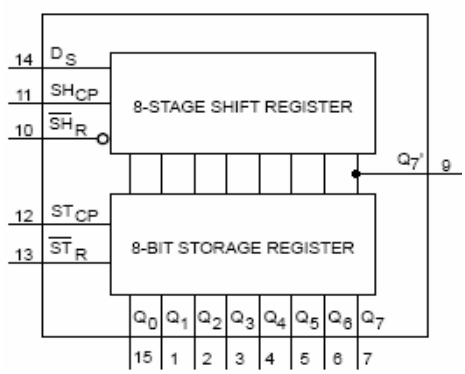

Figure 5. Hardware design of proposed code-words generator of 8-bit size

Since, the Transition detector needs a counter with [$\log_2$ m] stages, and thus the maximum transition count can be (n-1). The Transition counts using the Equation (4) are also computed and shown in Tables 4 and 5 for the binary Gray code-words and proposed binary code-words respectively. The Transition counts variations of both types of the code-words are shown in Figure 6.

Table 4. Binary Gray code-words (n = 2:6)

s. no.	6-bit		5-bit		4-bit		3-bit		2-bit	
	A	C	A	C	A	C	A	C	A	C
1	001011	3	00010	2	0000	0	000	0	00	0
2	001001	3	00110	2	0001	1	001	1	01	1
3	001000	2	00111	1	0011	1	011	1	11	0
4	011000	2	00101	3	0010	2	010	2	10	1
5	011001	3	00100	2	0110	2	110	1		
6	011011	3	01100	2	0111	1	111	1		
7	011010	4	01101	3	0101	3				
8	011110	2	01111	1	0100	2				
9	011111	1	01110	2						
10	011101	3	01010	4						
11	011100	2								
12	010100	5								

A: Code; C: Transition count.

Table 5. Proposed binary code-words (n = 2:6)

s. no.	6-bit		5-bit		4-bit		3-bit		2-bit	
	A	C	A	C	A	C	A	C	A	C
1	000000	1	00000	1	0000	1	000	1	00	1
2	000001	1	00001	1	0001	1	001	1	01	1
3	000011	1	00011	1	0011	1	011	1	11	1
4	000111	1	00111	1	0111	1	111	1	10	1
5	001111	1	01111	1	1111	1	110	1		
6	011111	1	11111	1	1110	1	100	1		
7	111111	1	11110	1	1100	1				
8	111110	1	11100	1	1000	1				
9	111100	1	11000	1						
10	111000	1	10000	1						
11	110000	1								
12	100000	1								

A: Code; C: Transition count.

7. RESULTS

To prevent multiple bits transitioning simultaneously, one can use a gray code to ensure that only a single bit transitions at a time between two subsequent code-words.

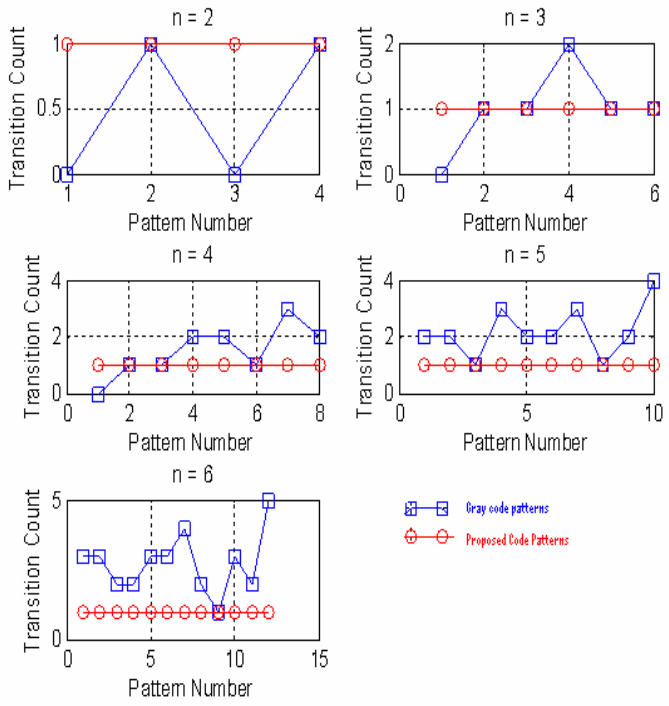

Figure 6. Transition counts variation graph

In the physical world, there is no way to ensure that all the bits will transition at exactly the same time, so a system may actually pass through a sequence of 000, 001, 011,…….

To demonstrate the practical approach of our proposed binary code-words we consider an example of an angular rotational disk of 60° slots. Therefore, six slots are required to rotate completely to 360°. Thus, n = 3 code generation scheme will be required to implement the system. On some parts of a track on a disk have metal contacts (contacts 1, 2 and 3), corresponding to a "1" (on), and other parts have insulator, corresponding to a "0" (off). The switching transitioning of the proposed code-words and the binary Gray code-words are shown in Tables 6 and 7 respectively. It is noted from Tables 6 and 7 that the proposed scheme has better transitions in comparison to binary Gray codes while switching from sector 4 to 5. The proposed scheme switches from on → on → on to on → on → off while the Gray code switches from off → on → off to on → on → off.

Table 6. A 60° angular rotational scheme using the proposed binary code-words

Proposed binary code-words				
Sector	Contact 1	Contact 2	Contact 3	Angle
1	off	off	off	0° to 60°
2	off	off	on	60° to 120°
3	off	on	on	180° to 240°
4	on	on	on	180° to 240°
5	on	on	off	240° to 300°
6	on	off	off	300° to 360°

Table 7: A 60° angular rotational scheme using binary Gray code-words

Proposed Gray code-words				
Sector	Contact 1	Contact 2	Contact 3	Angle
1	off	off	off	0° to 60°
2	off	off	on	60° to 120°
3	off	on	on	180° to 240°
4	off	on	off	180° to 240°
5	on	on	off	240° to 300°
6	on	on	on	300° to 360°

8. CONCLUSIONS

A novel technique for generating a subset of binary code-words with unitary Hamming distance is proposed. The generated code-words have better transitioning than the binary Gray code-words. The proposed scheme requires less hardware and reduces the propagation delays because it does not require binary code converter stage. Thus, it's a cost and time effective scheme of generating the code-words of unitary Hamming distance. The propagation delays further reduces due to the better transitioning attributes of the proposed codes. Also, less power dissipation and noise spike cuts are provided due to the constant transitioning.

Thereby, this paper presented an approach of generating the set of unique code-words which maintains the unitary Hamming distance property of the Gray code-words while providing tremendous attributes while generated using software or hardware. Since the built-in registers of the digital circuits are exploited to generate the proposed code-words hence, no burden of extra hardware. Generation of these code-words are much faster than the generations of the Gray code-words since the proposed scheme avoids the computation processes of the recursive equations of generating the Gray code-words. Further, providing the better transitioning of the proposed code-word signals ensures the faster actions and less power dissipation.

As far as the practical application of these proposed code-words the number of the rotor plates (see Figure 2) will certainly be increased but the sectors will be less on each plate in comparison to while applying the Gray code-words. For example if 16 code-words required for certain application then using the Gray code-words application requires 4 rotary plates with 16 sectors each whereas, the proposed code-words need 8 rotary plates with 8 sectors each. Since the cost of the design of the rotary plates are only once invested in the system design and it will not cost much in the long run while saving much more of the running cost due to the various attributes of the proposed code-words. Further, none of the applications uses the Gray code-words of size more than n = 8, as it is evident through the literature survey thus, the cost of the rotary plates is not much important. Also, using proposed code-words an added attribute of the reduction of the sectors on each plate will reduce complexity of switching stages.

ACKNOWLEDGMENT

The acknowledgements are due to authorities of Sultan Qaboos University (Sultanate of Oman) for providing research support grant (SQU-PGSR/ IG/ENG/ECED/07/04).

REFERENCES

[1] F. Gray, "Pulse Code Communication", United States Patent Number 2 632 058, March 17, 1953.

[2] F. G. Heath, "Origins of the Binary Code", *Scientific American*, vol. 227, no. 2, August 1972, p.76.

[3] W. M. Goodall, "Television by Pulse Code Modulation", *Bell Systems Tech. Journal*, vol. 30, 1951, pp. 33- 49.

[4] I. Flores, "Reflected number systems", *IRE Trans. Electron. & Computer*, vol. 5, no. 2, June 1956, pp. 79-82.

[5] E. N. Gilbert, "Gray Codes and Paths on the *n*-Cube", *Bell System Tech. Journal*, vol. 37, 1958, pp. 815-826.

[6] Alan A. Bertossi and Alessandro Mei, "Time and work optimal simulation of basic reconfigurable meshes on hypercubes", *Journal of Parallel and Distributed Computing*, vol. 64, no. 1, January 2004, pp.173 - 180

[7] Jywe-Fei Fang and Kuan-Chou Lai, "Embedding the incomplete hypercube in books", *Information Processing Letters*, vol. 96 no.1, 16 October 2005, pp.1-6.

[8] K. W. Cattermole, *Principles of Pulse Code Modulation*, New York: American Elsevier Publishing Company, Inc., 1969.

[9] Paul E. Black, "Gray code", *NIST National Institute of Standards and Technology*, 25 February 2004

[10] Martin Gardner, *Knotted Doughnuts and Other Mathematical Entertainments*, New York: W. H. Freeman, 1986

[11] White, "Ringing the cosets", *American Math. Monthly*, vol. 94, 1987, pp.721-746

[12] Dah-Jyu Guan, "Generalized Gray Codes with Applications",. *Proceeding National Science Council of Republic of China*, vol. (A) 22**,** 1998, pp. 841-848.

[13] F. Ruskey, "A Survey of Venn Diagrams", *Electrical Journal of Combinatorics*, 1997.

[14] Martin Gardner, "Mathematical Games", *Scientific American*, vol. 227, no.2, August, 1972, p.106.

[15] A. Nijenhuis and H. Wilf, "Combinatorial Algorithms for Computers and Calculators, New York: Academic Press, 1978.

[16] M. C. Er, "On generating the N-array reflected Gray codes", *IEEE Transactions on Computers*, vol. 33, 1984, pp. 739-741.

[17] J. Conway, N. Sloane, and A. Wilks, "Gray codes and reflection groups", *Graphs and combinatorial*, vol. 5, 1989, pp. 315-325.

[18] S. Skiena, *Gray Code, Computational Discrete Mathematics: Combinatorics and Graph Theory with Mathematica*, Reading, MA: Addison-Wesley, 1990, pp. 42-43 and 149.

[19] B. Hollstien, "Artificial Genetic Adaptation in Computer Control Systems", PhD thesis, University of Michigan, 1971.

[20] A. Ahmad and M. M. Bait Suwailam, "Design of an Efficient and Less Memory Requiring Algorithmic Procedure for Computing Gray Codes", *Proceedings IEEE / IEE int'l conf. communication, ICCCP'07*, 2007, pp. 172-176.