A Novel Formulation of Hamming Code

Conference Paper · May 2009

DOI: 10.1109/ECTICON.2009.5137169

2 authors:

Arash Ahmadpour
LIAU
11 PUBLICATIONS **21** CITATIONS
SEE PROFILE

A. Ahadpour Shal
Islamic Azad University of Lahijan
17 PUBLICATIONS **32** CITATIONS
SEE PROFILE

A Novel formulation of Hamming Code

[1]A. Ahmadpour and [2]A. Ahadpour Shal, [3]M. Ziabari
Faculty of Electrical Engineering, Islamic Azad University Lahijan Branch
[1]ar.amp@iau-lahijan.ac.ir, [2]alirezaahadpoor@yahoo.com, [3]81811121@ee.iust.ac.ir

***Abstract*– Nowadays, usually data is transmitted through a communication channel in the form of binary codes, which undesirably their data bits can be masked and changed by noise. Detecting and correcting these errors is important. In this paper, a general formula of Hamming Code to recognize a single bit error in composite data word-parity bits is presented and an application of our model is introduced for logic circuit design using unique capability of Quantum Dot Cellular Automata (QCA). Simplicity and ability of using this new formula to find out parity bits, check bits and error bit position is more suitable than other current methods.**

***Keywords*- Hamming Code, Parity bits, Check bits, Quantum Dot Cellular Automata (QCA).**

I. INTRODUCTION

In the late 1940's Richard Hamming recognized that the further evolution of computers required greater reliability, in particular the ability to detect and correct errors, which at the time, parity checking was being used to detect errors, but was unable to correct any errors. He created the Hamming Codes, perfect 1-error correcting codes and the extended Hamming Codes 1-error correcting and 2-error detecting codes. The Hamming codes use multiple parity checks to locate and correct single-bit errors. If some parity bits are correct and others are not, the bit in error can be deduced. Each check is now a sum only over bits in selected positions. In the simplest case, message words of length $(2^k - \mathrm{k} - 1)$ bits, where k is any integer, are to be sent together with k check bits, so that each code word (message bits plus check bits) contains $(2^k - 1)$ bits. The positions of the code word are numbered from left to right. The first check bit is in position 1, and is a parity check over the positions that have a 1 as the least significant bit of their binary representations (that is, positions 1, 3, 5, 7, ...). The second check bit is in position 2, and is a parity check over the positions that have a 1 as the second least significant bit of their binary representations (that is, positions 2, 3, 6, 7, ...), and so on. If no parity checks fail, the code word is assumed to be correct. Also, if one bit of the code word is in error, the error is at the location whose binary representation equals the pattern of the failed parity checks. Hamming Codes are still widely used in computing, puzzles, and turbo codes. In addition, it's provides an error correction mechanism that is used by many applications such as Quantum Dot Cellular Automata (QCAs), electronic chips and satellite communications devices, and has been proposed for deeply faded wireless asynchronous transfer mode networks [1-5].

Having an algorithm, which is able to define all previous calculations and can give us parity bits, check bits and the exact position of error occurred bit, all in a single formula, can be suitable for better and more accurate design of circuits in future.

In this paper, a Hamming code formulation with related computations is proposed in section II. Also, there is an overview of QCA and designing of Modular Chip Schematic in section III and IV, respectively. Then it is followed by a conclusion and a proposal of our hamming code formula application in their design.

II. HAMMING CODE

This kind of binary code normally consists of a n bit binary word and k parity bits which are added to n bit word to form a new $(\mathrm{n} + \mathrm{k})$ bits word. Of course the word can have any length, if we suppose our primary binary n bit word is $\mathrm{b}_0, \mathrm{b}_1, \ldots \mathrm{b}_{\mathrm{n}-1}$, we can define $\mathrm{k} = (1 + \log_2^{\mathrm{n}})$ parity bits $\mathrm{p}_1, \mathrm{p}_2, \ldots \mathrm{p}_{2^{\mathrm{k}-1}}$, and composing them together will result a $(\mathrm{n} + \mathrm{k})$ bits word [5]. It will be convenient if we localize them in the format of Table I with position numbers of $1, 2, 3, \ldots, \mathrm{n} + \mathrm{k}$.

We define "Set" as an array of bit position in equation (1) in order to drive check bits in equation (2). In which $C_1, C_2, \ldots C_{2^{k-1}}$ and $BP_1, BP_2, \ldots, BP_i$ are check bits and value of bit position number " i ", respectively.

$$\mathrm{Set}(\mathrm{BP_i})|_{\mathrm{i}=1}^{\mathrm{M}} \equiv (\mathrm{BP_1}, \mathrm{BP_2}, \ldots, \mathrm{BP_M});\ \mathrm{M} \le \mathrm{n} + \mathrm{k} \qquad (1)$$

$$\mathrm{C}_{2^{\mathrm{L}-1}} = \mathrm{XOR}\left(\mathrm{Set}\left(\mathrm{Set}\left(\mathrm{BP}_{(2\mathrm{m}+1).2^{\mathrm{L}-1}+\mathrm{S}}\right)|_{\mathrm{S}=0}^{2^{\mathrm{L}-1}-1}\right)\right)|_{\mathrm{m}=0}^{\mathrm{M}} \qquad (2)$$

In mention above, L a natural number is between 1 to k. It is obvious that if $\mathrm{n} \neq 2^{\mathrm{i}}$ in which is i an integer number, k will be an irrational number. In this case the formula of k should be modified as follow:

$$\mathrm{k} = 2 + [\log_2^{\mathrm{n}}] \qquad (3)$$

As an example, if $\mathrm{n} = 12$ then $\mathrm{k} = 2 + [\log_2^{12}] = 5$, it means the number of parity bits is 5 which can be shown by $\mathrm{p}_1, \mathrm{p}_2, \mathrm{p}_4, \mathrm{p}_8, \mathrm{p}_{16}$ and can be arranged as a Table II. Now we can use formula (1), (2), (3) and Table II in order to obtain $C_1, C_2, C_4, C_8, C_{16}$ for instance putting $\mathrm{L} = 1$ in formula (2), gives C_1 as follow:

TABLE I
THE POSITIONS OF PARITY BITS AND DATA BITS IN A COMPOSITION (n + k) BIT WORD

Bit position	1	2	3	4	5	6	7	8	…	n+k
Bits	P_1	P_2	b_0	P_4	b_1	b_2	b_3	P_8	…	BP_{n+k}

TABLE II
THE POSITIONS OF PARITY BITS AND DATA BITS IN A COMPOSITION (n + k) = 17 BIT WORD

Bit position	1	2	3	4	5	6	7	8	…	16	17
Bits	P_1	P_2	b_0	P_4	b_1	b_2	b_3	P_8	…	P_{16}	$BP_{17}=b_{11}$

TABLE III
THE POSITIONS OF PARITY BITS AND DATA BITS IN A COMPOSITION (n + k) = 17 BIT WORD

Bit position	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Bits	1	0	1	0	0	1	1	0	0	1	1	1	0	0	1	1	1

TABLE IV
THE POSITIONS OF PARITY BITS AND DATA BITS IN A COMPOSITION (n + k) = 17 BIT WORD

Bit position	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
Bits	1	0	1	0	0	1	1	0	0	1	1	1	1	0	1	1	1

$$C_{2^{1-1}} = C_1 = \mathrm{XOR}\left(\mathrm{Set}\left(\mathrm{Set}\left(BP_{(2m+1)+S}\right)|_{S=0}^{S=0}\right)\right)|_{m=0}^{M} \quad (4)$$

The inner parenthesis consist of only one sentence (BP_{2m+1}) in which increasing m from zero up to M produces different bits of BP_i for outer parenthesis which are as follow in C_1:

$$C_1 = \mathrm{XOR}(BP_1, BP_3, BP_5, \dots, BP_{M=2m+1}) \quad (5)$$

In the above mentioned equation, M is the largest allowed number less than or equal $(n + k)$ which in this special example is 17. In the next step, to estimate C_2 the same procedure should be followed by using $L = 2$. It means:

$$C_{2^{2-1}} = C_2 = \mathrm{XOR}(\mathrm{Set}\left(\mathrm{Set}\left(BP_{2(2m+1)+S}\right)|_{S=0}^{S=1}\right))|_{m=0}^{M} \quad (6)$$

Again the inner parenthesis consists of two sentences (BP_{4m+2}, BP_{4m+3}). Changing the value of m from zero to M, will produce different bits of BP_i of outer parenthesis as follow:

$$C_2 = \mathrm{XOR}(BP_2, BP_3, BP_6, BP_7, \dots, BP_{4m+2}, BP_{4m+3}) \quad (7)$$

As we mentioned $M \leq n + k$ and in equation (7) inner parenthesis after the third repeat of the loop counter 'm' the value of M will be 15 and we can not have the forth repeat of m in which M will be 19 that is not acceptable. With the same procedure C_4, C_8, C_{16} can be obtained by following relations.

$$\begin{aligned} C_{2^{3-1}} &= C_4 \\ &= \mathrm{XOR}\left(\mathrm{Set}\left(\mathrm{Set}\left(BP_{4(2m+1)+S}\right)|_{S=0}^{S=3}\right)\right)|_{m=0}^{m=M} \\ &= \mathrm{XOR}(BP_4, BP_5, BP_6, BP_7, BP_{12}, BP_{13}, BP_{14}, BP_{15}) \end{aligned} \quad (8)$$

$$\begin{aligned} C_{2^{4-1}} &= C_8 \\ &= \mathrm{XOR}\left(\mathrm{Set}\left(\mathrm{Set}\left(BP_{8(2m+1)+S}\right)|_{S=0}^{S=7}\right)\right)|_{m=0}^{m=M} \\ &= \mathrm{XOR}(BP_8, BP_9, BP_{10}, BP_{11}, BP_{12}, BP_{13}, BP_{14}, BP_{15}) \end{aligned} \quad (9)$$

$$\begin{aligned} C_{2^{5-1}} &= C_{16} \\ &= \mathrm{XOR}\left(\mathrm{Set}\left(\mathrm{Set}\left(BP_{16(2m+1)+S}\right)|_{S=0}^{S=15}\right)\right)|_{m=0}^{m=M} \\ &= \mathrm{XOR}(BP_{16}, BP_{17}) \end{aligned} \quad (10)$$

The next step which normally should be taken before the above mentioned computation is to estimate p_1 to p_{16}. We don't need to present independent formulas, and we can omit the lowest BP from each check bits C_1 to C_{16} i. e. p_1 to p_{16} will be:

$$p_1 = \mathrm{XOR}(BP_3, BP_5, \dots, BP_{15}, BP_{17}) \quad (11)$$

$$p_2 = \mathrm{XOR}(BP_3, BP_6, BP_7, BP_{10}, BP_{11}, BP_{14}, BP_{15}) \quad (12)$$

$$p_4 = \mathrm{XOR}(BP_5, BP_6, BP_7, BP_{12}, BP_{13}, BP_{14}, BP_{15}) \quad (13)$$

$$p_8 = \mathrm{XOR}(BP_9, BP_{10}, BP_{11}, BP_{12}, BP_{13}, BP_{14}, BP_{15}) \quad (14)$$

and

$$p_{16} = \mathrm{XOR}(BP_{17}) = BP_{17} \quad (15)$$

As a final numerical example we suppose the 12-bit binary word is 101101110011. Inserting the bit position value in formula (11) to (15), results following parity bits.

$$p_1 = 1, p_2 = 0, p_4 = 0, p_8 = 0, p_{16} = 1$$

Thus, we have 17 bits word has Table III as transmitted data. If there is a single bit error in bit position 13, the received actual data will be as Table IV, so that the check bits C_1 to C_{16} will be computed as follow:

$$C_1 = 1, C_2 = 0, C_4 = 1, C_8 = 1, C_{16} = 0$$

The result is $C_{16}C_8C_4C_2C_1 = 01101$.

It is obvious that the five bit binary number formed by check bits equals 13 and gives the position of erroneous bit. Up to now, the word with arbitrary length and the needed parity bits was discussed. As it is shown the check bits in mentioned order can give us the single bit error position so that it can be corrected simply. As equation (2) shows the hardware at the check bits function can be easily implements by XOR or NAND gates.

Our special aim of proposing the formulation of Hamming code is its application in quantum dot cellular automata (QCAs) in which we can accomplish hamming code hardware by quantum cells [9]. The QCA is attracting a lot of attention due to its extremely small feature size (at the molecular even atom level) and ultra low power consumption [1], [6].

III. QUANTUM DOT CELLULAR AUTOMATA

Quantum dot cellular automata (QCA) which has been proposed by Lent, et al, in 1993 can be a convenient substitution for silicon/CMOS switch in near future. This promising pattern which has been actually fabricated in 1997 principally consists of four charge container or "dot" which should be located in the corners of a square which is schematically shown in Figure 1.a. As can be seen in this figure QCA can be supposed as a novel nano device which stores binary information in the form of position of individual electrons.

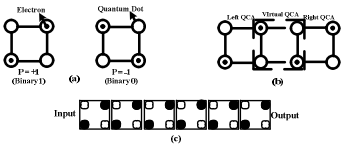
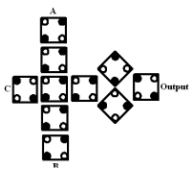


Figure 1: (a) QCA cell polarizations, (b) Interaction between two QCAs, (c) QCA wire

Having a careful look in localization of electrons in the QCA will reveal that two neighboring QCAs like figure 1.b will interact with each others. It means that setting the left QCA in "1" position forces the right QCA through Columbic repulsion and absorption forces to the same "1" position. This criterion i. e. controlling the position of a single electron in a QCA by position of another single electron in a neighboring QCA has been actually demonstrated in university of Notre dame [7]. Extending the above mentioned interacting property to a row of QCAs like figure 1.c shows that, Coulomb forces causes all the cells to adopt the same polarity. This configuration can be regarded as a QCA wire [8].

Whereas the logic circuits are based on NAND gate, a sample of this gate is shown in figure 2. It is obvious that when C = 0 the logic element is equivalent to a NAND gate and when C = 1 it operates as a NOR gate.



C	A	B	Output
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

Figure 2: A logic element equivalent to NAND/NOR gate and its truth table [9]

It is well known that every programmable device consist of vertical and horizontal lines which according to specified programming pattern should be interconnected. In a normal ROM these interconnections are shown by cross signs which show existing of intact fuses between lines. Overlapping lines without cross signs means no interconnections which show existing of blown fuses between lines. These interconnections provide the path for information flow and using them are inevitable in all programmer devices such as ROM, PROM, EPROM, EEPROM, PLA and PLA. Figure .3 shows two QCAs wires. One composed of cells oriented at 90 degree and the other from cells oriented at 45 degree. It is obvious that the two wires can cross each other without any interfering. But, in this figure there is not any mechanism by which the signals from 45 degree wire can jump to 90 degree wire and the signal in this configuration would not be able to turn corner.

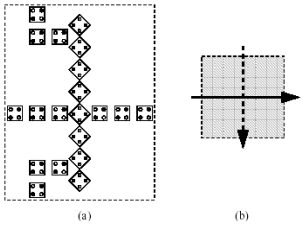


Figure 3: (a) crossing of two signals (b) Ultimately ineffective as a routing element [10]

To improve this routing element some configurations have been introduced [10], which had some problems so that it is proposed to use schematic shown in Figure .4. This proposed routing element will optimize delay, latching, chip occupation area and manufacturing cost.

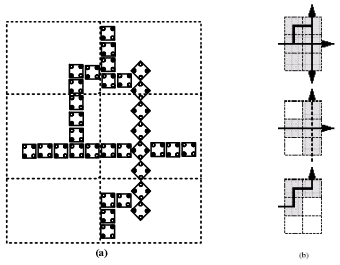


Figure 4: (a) QCA routing element with $3 \times 2 = 6$ zones, (b-up) routing element acts as fan out, (b-middle) signal propagate in two perpendicular QCA lines, (b-down) signal turns corner upward and is buffered from the down side [10]

Now we are in position to give a modular pattern composed of logic and routing elements. These modules can be called QCA chips. The fabrication of these chips needs many sophisticated technologies such as two layer grid routing techniques. The final pattern is severely dependent on ultimate precision of channel depth fabrication.

IV. MODULAR CHIP SCHEMATIC

This pattern can be drawn as figure .5. In this figure, circles represent logic block and squares represent the possible path from one gate to another gate. These paths

are programmable and controllable by clocking zones [9]. To accomplish the desired routing a same mechanism which is already used in traditional CMOS, is usable.

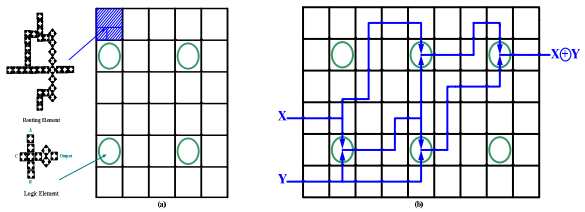


Figure 5: (a) proposed pattern for a block of modular chip, (b) Mapping of NAND gates in NAND-based QCA chip to simulate the XOR-gate[9]

According to figure .6, as mentioned earlier the logic of XOR-gate can be converted to NAND logic. The next step is placing this logic in QCA NAND-based chip [9].

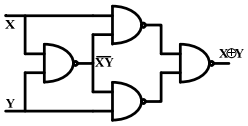


Figure 6: Exclusive-OR constructed with NAND gates

V. CONCLUSION

In this paper, a new formulation of Hamming Code to identify a single bit error in composite data word-parity bits has been presented. Simplicity and ability of using this algorithm to find out parity bits, check bits and error bit position is more suitable than other current methods. Also, using special capability of QCAs a design work has been proposed. This was actually an application of hamming coder/decoder for logic networks. In feature, this algorithm can be used for better and simpler nano scale IC design with any desirable length and also this proposed system can be used in digital and satellite communication. In order to small size and very low power consumption; it will be unavoidable using of this proposed system.

ACKNOWLEDGMENT

The author would like to appreciate Dr. A. Ghadimi and R. Fouladi and for their support and encouragement on this paper.

REFERENCES

[1] Kyosun Kim, Kaijie Wu and Ramesh Karri, "Quantum-Dot Cellular Automata Design Guideline," *IEICE Trans. Fundamental,* vol.E89-A, NO.6 June 2006.

[2] Abuelyman E. and Daniel S., "SPEC: Single-Packet Error Control Protocol for WATM Networks," *Proceedings of the 2003 ICWN*, pp 504-509, Las Vegas, USA, 2003.

[3] G. A. Aderounmu et al, "Performance Comparison of Data Link Control Protocols for wireless ATM Networks," *International Journal of Computers and Applications*, vol 24, No 3, pp 144-152, September, 2002.

[4] Eltayeb S. Abuelyaman and Abdul-Aziz S. Al-Sehibani, "Optimization of the Hamming Code for Error Prone Media," *IJCSNS International Journal of Computer Science and Network Security*, vol 8, No 3, March 2008.

[5] R. W. Hamming, "Error Detection and Error Correction Codes," *Bell System Tech. Journal*, vol 29, pp 147-160, April, 1950.

[6] C.S. Lent, P.D. Tougaw, W. Porod, and G.H. Bernstein, "Quantum cellular automata," *Nanotechnology*, vol.4, pp.49–57, 1993.

[7] A. Orloy, I. Amlani, G. Bernstein, C. Lent and G. Snider, "Realization of a functional cell for quantum-dot cellular automata, " *Science,* 277:928-930, 1997.

[8] A. Orloy. I. Amlani, C. Lent, G. Bernstein and G. Snider, "Experimental demonstration of a binary wire for quantum-dot cellular automata," *Appl. Phys. Lett.,* 74; 2875-77, 1999.

[9] A. Ahmadpour and M. Ziabari, "Designing a Hamming Coder/Decoder using QCAs," *The 8th Conference of Science and Research, Faculty of Electrical Engineering, University of Guilan*, pp 200-203, Rasht, IRAN, March, 2008.

[10] T. N. Michael, A. F. Rodrigues and P. M. Kogge, "A Potentially Implementable FPGA for Quantum Dot Cellular Automata, *"Published in workshop on Non-Silicon Computation (NSC), in Conjunction with Int. Symp. On High Performance Computer Architecture*, HPCA, 2003.