Constraint Algorithm in Double-Base Number System for High Speed A/D Converters

Minh Son Nguyen*, Man-Ho Kim* and Jongsoo Kim*

Abstract – In the paper, an algorithm called a Constraint algorithm is proposed to solve the fan-in problem occurred in ADC encoding circuits. The Flash ADC architecture uses a double-base number system (DBNS). The DBNS has known to represent the multi-dimensional logarithmic number system (MDLNS) used for implementing the multiplier accumulator architecture of FIR filter in digital signal processing (DSP) applications. The authors use the DBNS with the base 2 and 3 to represent binary output of ADC. A symmetric map is analyzed first, and then asymmetric map is followed to provide addition read DBNS to DSP circuitry. The simulation results are shown for the Double-Base Integer Encoder (DBIE) of the 6-bit ADC to demonstrate an effectiveness of the Constraint algorithm, using 0.18μ m CMOS technology. The DBIE's processing speed of the ADC is fast compared to the FAT tree encoder circuit by 0.95 GHz .

Keywords: Double-Base number system, Flash ADC, Asymmetric DBIE and constraint algorithm

1. Introduction

1.1 A Double Base Number System

In [1] and [4], a new double-base redundant number system was proposed with the following number representation,

$$X = \sum_{i,j}^{k,m} d_{ij} 2^i 3^j \qquad (1)$$

where $d_{ij} \in \{0,1\}$, $i \in (0,k)$ and $j \in (0,m)$ are independent integers we will refer to as binary and ternary exponents respectively. A 2-dimensional table will be used to illustrate the representation scheme. Since i and j are independent, we can represent any number as indices in 2 dimensions. The base 2 is represented as the x-axis and the base 3 as the y-axis as in Fig. 1. The summation of non-zero squares is equal to the given number X. In reference [1, 6], the representation by the Greedy algorithm was suggested as the name of Near Canonic Double Base Number Representation (*NCDBNR*) for real time application due to NP characteristics. Representation of a number that has smallest number of non-zero squares is called the Canonical Double Base Number Representation (CDBNR). As an example, generating a DBNS representation for $X = 41$ of left side can be converted to the right side by the greedy algorithm as in Fig 1. More details about DBNS arithmetic operations can be founded in [1, 4, 5, 7] and [8]. Two consecutive non-zero squares can be replaced by the non-zero square depending on its pattern after shifting downward or right. Therefore, addition and multiplication can be done easily.

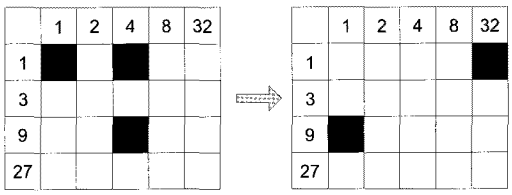


Fig. 1 Greedy algorithm and the canonic DBNS

In this paper, a new algorithm is presented to process the DBNS with base 2 and 3, since the DBNS has the fan-in problem in ADC. The remainder of this paper is organized as follows. In Section II, the basic idea of DBNS and the proposed Constraint algorithm are introduced after reviewing the flash ADC architecture briefly in this section. Section III gives the experimental results of the Constraint algorithm for the 6-bit ADC. In Section IV, we conclude with a summary and future works.

1.2 Flash ADC

There were many efforts to design and implement low power and chip area consumption of the fast ADC in [11, 12, 13] and [14]. After converting analog signals into digital signals, a DSP does arithmetic operations heavily. Therefore, we give a new flash ADC with DBNS to solve such problem in DSP circuit.

† Corresponding Author: All authors are with the School of Electrical Engineering, University of Ulsan. Ulsan, Korea. (sonlu@ulsan.ac.kr, jamesmhkim@mail.ulsan.ac. kr, jongsoo@ulsan.ac.kr)
Received 10 October, 2007 ; Accepted 11 May, 2008

Fig. 2 describes the architecture of a flash ADC using DBIE to convert analog signal to double base integer system for real time DSP application. In this architecture, the comparator and 0-1 generator circuits were well referred in [2]. The only difference from the typical architecture is the last stage. In this paper, the authors are only interested in designing the DBIE circuits. The implementation of DBIE circuits depends on the result of the Constrain algorithm used to represent by DBNS.

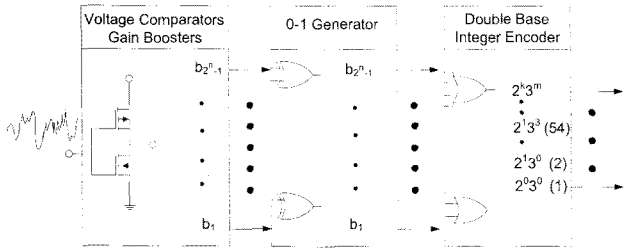


Fig. 2 Flash ADC with DBNS

2. DBNS-Map and Constraint Algorithm

With n-bit ADC, the output is represented binary n-bits. But, every number X ϵ (0, 2^n -1) must be represented by equation (1) in DBNS format. In [4], the computational complexity to find the Canonical value was high. Thus, instead of getting CDBNR, they tried to represent all Xs on a NCDBNR. However, our goal is to find the minimum size and the minimum number of addition in DBNS-map to solve the fan-in problem in ADC encoding circuit [15].

In this section, logical method is presented to calculate the number of bits and additions to represent any number Xs with the following definition and lemma.

Definition: *DBNS-map is a symmetric if number of columns and rows are equal, or if numbers of columns (rows) are greater than number of rows (columns) by 1.*

Assume that DBNS-map has k columns and m rows. $|k - m| = l_0$ is called asymmetric level of DBNS-map, where l_0 = positive integer such as $1 < l_0 < m, k$. We try to find $|k - m| = l_0$ with $l_0 = 1, 0$. Assume X ϵ (0, 2^n -1) is binary integer number, thus $X < 2^n$.

Besides, we always have $\exists X \in (0, 2^n - 1)$,

$$X = 2^k 3^m + \sum_{i,j}^{k-1,m-1} d_{ij} 2^i 3^j < 2^n \qquad (2)$$

Therefore,

$$2^k 3^m < 2^n \quad \text{Or,} \quad k\ln 2 + m\ln 3 < n\ln 2 \qquad (3)$$

If we suppose that DBNS-map is symmetric, k = m.
In equation (3) gives:

$$k = m < \frac{n\ln 2}{(\ln 2 + \ln 3)} \qquad (4)$$

If we suppose that DBNS-map is asymmetric,

$$k = m - l_0, \text{ or} \qquad m = k - l_0.$$

Besides, $1 < l_0 < k$ or m. Assume that l_0 = k or m is called maximum asymmetric level of DBNS-map, then: m = k/2 or m = 2k.

Substitution in (4), given by:

$$k < \frac{n\ln 2}{(\ln 2 + \frac{1}{2}\ln 3)} \qquad (5)$$

or:

$$k < \frac{n\ln 2}{(\ln 2 + 2\ln 3)} \qquad (6)$$

Compare equation (4) with equation (5) and (6). In order to find k and m of DBNS-map with symmetric figure, we should refer k in

$$\left(\frac{n\ln 2}{(\ln 2 + 2\ln 3)}, \frac{n\ln 2}{(\ln 2 + \frac{1}{2}\ln 3)} \right) \qquad (7)$$

Algorithm finds m and k to be able to get best DBNS-map represented for 2^n –1 numbers used for DBIE of ADC.

Input: number of bits of ADC – n,
FAN-IN constraint parameter - n_f
Output: number of columns of DBNS-map – k, number of rows of DBNS-map – m,
number of stages of DBInt-Encoder - S_{min}

PSEUDO algorithm

Procedure of DBIE Architechture (n)
1. Calculate k_{min} and k_{max} from boundary values of (7).
2. Find number of stage s responding with every pair (k_o, m_o), where k_o and m_o ϵ (k_{min}, k_{max}).
3. With each of s, check whether S is minimum.
4. If S is minimum, return S_{min}, m and k; else jump to step 2

Algorithm to find number of stages and the presentation of 2-integer number.

Input: number of bits of ADC – n,
number of columns – k, number of rows - m.

Output: number of stages of DBIE – s.

We knew that number X ϵ (0, 2^n -1) with the following formula,

$$X = 2^{i_1}3^{j_1} + 2^{i_2}3^{j_2} + \cdots + 2^{i_s}3^{j_s} \quad (8)$$

where, $i_1 \neq i_2 \neq \ldots \neq i_n$; $j_1 \neq j_2 \neq \ldots \neq j_n$; $i_i < m$, $j_j < k$

And number of addition terms in formula (8) are S. Let call S_1, S_2,…, S_i, …, S_s are stage to generate every number 2-integer X.

So, every S_i stage will give the set of 2-integers in the following:

$$S_i = \left\{2^0 3^0, 2^1 3^1, 2^1 3^0, 2^0 3^1, \ldots, 2^k 3^m\right\}$$

Thus, number X = $2^0 3^1 + 2^i 3^j + \ldots + 2^k 3^m$ can be represented as distinguishable sub-graph line (connected paths) of G graph representation. Use the Handshaking lemma in graph theory in [3]. Therefore equation (9) is given,

$$n_f * (k*m) * s = 2 * m_e \quad (9)$$

where m_e is the number of edges and s is the number of stages. Note that the graph G must include all connected paths are represented by number X ϵ (0, 2^n -1) => $m_e > 2^n$. So, substituting in (7) gives $n_f*(k*m)*s > 2*2^n$. Or,

$$s > \frac{2^{n+1}}{n_f * k * m} \quad (10)$$

PSEUDO algorithm

Procedure Find_DBInt_Stages(n, k, m)

1. Calculate $S_{temp} = round\left(\frac{2^{n+1}}{n_f * k * m} + 0.5\right)$
2. Call set V include integer number X ϵ (1, 2^n -1). Find V' = V – {X_i ϵ set S_i}.
3. For every number of stage i (i < S_{temp}), find set U = {X_j has (i –1) additions and every term ϵ set S_i}. V" = V' – U.
4. If V" is empty, stop and return S_{min} = i. Else increase i by 1 and jump in step 3.

From above algorithm derivation, we were interested in reducing the output as minimum number. With symmetric DBNS-map definition, the author can get results that sufficiently satisfy the proposed requirements. However, the above algorithms didn't solve the problem of reducing two contiguous non-zero squares by rules 3 and 4 to decrease number of additions in every 2- integer representation [4,9]. With these symmetric DBNS-map, the Constraint algorithm can't yield addition ready DBNS (ARDBNS). The ARDBNS means there are not consecutive squares in the DBNS-map, since the consecutive squares can be reduced into one square by just shifting downward by 1 or right side by 2. If a DSP circuitry needs ARDBNS, the DBNS-map must be asymmetric. Thus, the numbers of encoder output increase more.

To solve this problem, we should append columns or rows from the previously defined symmetric DBNS-map to asymmetric DBNS-map to represent all numbers in (0, 2^n - 1). It is simply to know that the maximum number of column of asymmetric DBNS-map will not be greater than n-number of ADC bits. The pseudo algorithm will describe to find asymmetric DBNS-map in the following.

PSEUDO algorithm

Procedure Find_Asym DBIE

1. Find number of rows and stages of Asymmetric DBNS-map from symmetric DBNS-map
2. Call set V include integer number X ϵ (1, 2^n -1). Find V' = V – {X_i ϵ set S_i}.
3. For every number of stage i (i < number of stages of Asymmetric DBNS-map), set U = {X_j has (i –1) additions, every term ϵ set S_i and sum of terms is not ϵ set S_i}. V" = V' – U.
4. If V" is empty or i = number of stages of Asymmetric DBNS-map, stop and exit. Else increase i by 1 and jump in step 3.

To check accuracy of above algorithms, we implemented these algorithms with n = 6 and shown the result in next Section.

3. DBIE Results

Appling above procedures with 63 numbers X from 1 to (2^6 –1), we get 4x4 DBNS-map and the maximum number of additions of DBIE is 2 in case of symmetric map, while getting 4x6 DBNS-map with the same number of addition 2 in case of asymmetric map. After comparing with the result of the Greed algorithm in [1], the number of additions was 3 in Greedy algorithm when the input is 53_{10}. Due to the increasing of the input terms, the asymmetric map provided less fan-in problem, even though it needs more chip area and output pins. For example, twelve among 63 inputs are required 2 additions in symmetric map, which mean there are 3 cells (non-zero square). The

asymmetric map can process 5 inputs as 2 additions.

As shown in table 1, the number of fan-in of DBIE circuits must satisfy electrical design requirements of DBNS. However, the Greedy algorithm approach can't meet the requirement of the number of fan-in in encoder circuits.

To prove DBNS representation of the Constraint algorithm, we designed DBIE circuit with Mentor CAD tool as shown in Fig. 3. The simulation of DBIE circuits was done by H-spice software. With the simulation result we verified that the designed circuit worked properly. In designing DBIE circuit, we used Magna 0.18um CMOS technology. The simulated output waveforms are shown in Fig.4. Table 2 is the summary of timing information between the input and output waveform for the Constrain algorithm in *ns* unit. We supplied the 2.5GHz input waveform shown in the table. From the tables, the asymmetric algorithm shows better performance than the symmetric case with the delay time.

Table 1. The fan-in numbers

Greedy		Constrain Symmetric		Constrain Asymmetric	
Cell	Fan in	Cell	Fan IN	Cell	Fan in
54	10	54	10	54	9
				48	5
36	18	36	11	36	11
				32	6
27	9	27	12	27	6
24	3	24	10	24	3
18	6	18	7	18	6
				16	4
12	11	12	8	12	5
9	7	9	8	9	7
8	4	8	10	8	5
6	8	6	7	6	5
4	14	4	10	4	9
3	7	3	8	3	11
2	10	2	12	2	11
1	21	1	12	1	12

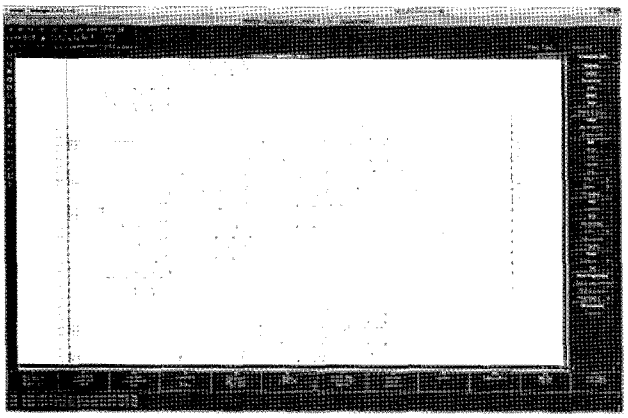
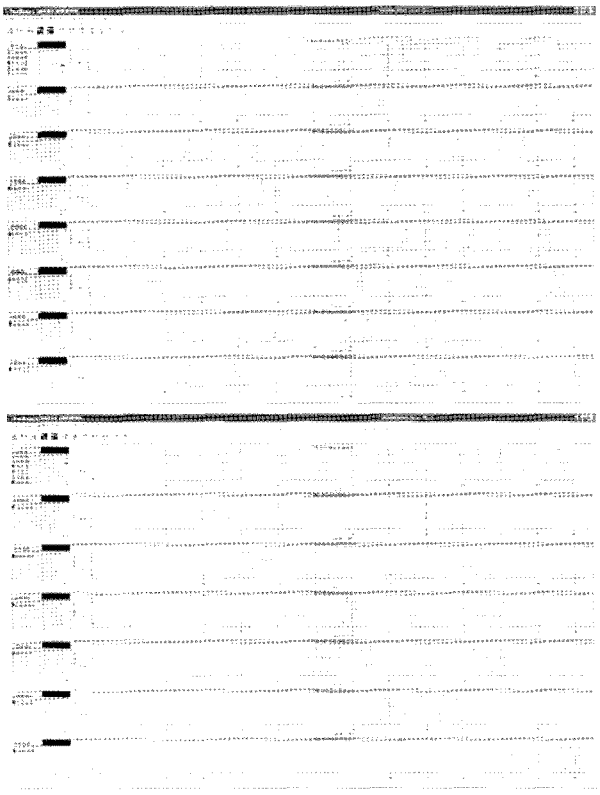
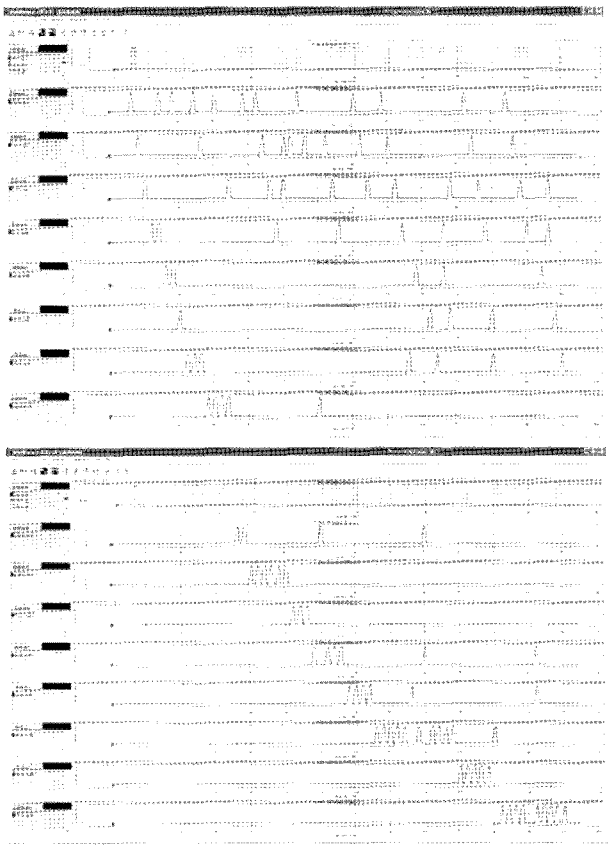


Fig. 3 Schematic circuit of DBIE



(a) Output waveforms symmetric map



(b) Output waveforms asymmetric map

Fig. 4 H-spice output waveforms

Table 2. Timing information

		delay	rising	falling	high	low	period
Symm	Input		0.05	0.05	0.175	0.125	0.4
	Output	0.171	0.09	0.082	0.09	0.114	0.376
Asym	Input		0.05	0.05	0.175	0.125	0.4
	Output	0.13	0.087	0.063	0.127	0.096	0.373

Table 3. Summarize three encoders used for ADC from Hspice simulation with CMOS 0.18um

HSpice results is simulated by technology of CMOS 0.18um			
	Symmetric DBIE	Asymmetric DBIE	FAT
Max speed (Ghz)	6.26	6.66	5.71
Number of transistors	380	424	498
Max power (mW)	21.49	17.26	12.44
Output bits	13	16	6

In reference [10], the FAT encoder is faster than the ROM encoder for the 6-bit ADC. In this paper, the author simulated the FAT encoder at schematic level with Hspice. Table 3 described some parameters of three encoders, two are proposed encoders, and one is the FAT encoder. The asymmetric DBIE has 6.66GHz maximum speed. The FAT tree encoder shows only 5.71GHz, while the symmetric DBIE encoder yields 6.26GHz. Thus, the asymmetric DBIE encoder can enhance up to 0.95GHz(16.64%) in case of 6-bit ADC compared with the FAT tree, which is known as the fastest encoder now.

4. Conclusions

In this paper, we presented a new algorithm to implement Double Base Integer Encoder that will be used in ADC. With the new Constraint algorithm, it was possible to support more fast ADC speed because the Constraint algorithm reduced addition stages in DBIE circuits for DSP. The comparison between two algorithms may not be reasonable, since the Greedy algorithm is not developed for encoder circuits. In this work, the goal of comparison is just to show that the minimum approach is not appropriate sometimes. The asymmetric DBIE yielded more speed gain than any other types of encoders. However, the real advantage of processing speed may be obtained in DSP circuitry at the cost of more power consumption.

To improve this algorithm, we can apply the depth first search with some parameters to find good double base number system used for our design. Last, the DSP arithmetic circuits will be embedded with ADC as future works.

References

[1] V. S. Dimitrov and G. A. Jullien, "A new number representation with applications," *IEEE Circuits and Systems Magazine*, 2003.

[2] J. Yoo, "A TIQ based CMOS flash A/D converter for SoC applications," PhD Thesis, Department of Computer Science and Engineering of the Pennsylvania State University, 2003.

[3] http://en.wikipedia.org/wiki/Handshaking_Lemma.

[4] V. S. Dimitrov and G. A. Jullien, and W. C. Miller, "Theory and applications of the double-base number system," *IEEE Transactions on Computers*, vol. 48, pp. 1098 – 1106, 1999.

[5] G. Gilbert and J. M. P. Langlois, "Multipath greedy algorithm for canonical representation of numbers in the double base number system," *IEEE-NEWCAS Conference*, pp. 39-42, 19 – 22, June 2005.

[6] V. Berthe, L. Imbert, and G. A. Jullien, "More on converting numbers to the double-base number system," Research Report LIRMM – 0403 1, Montpellier France, Oct. 2004.

[7] Y. Ibrahim, W. C. Miller, Graham A. Jullien, and Vassil S. Dimitrov, "DBNS addition using cellular neural networks," *IEEE International Symposium on Circuits and Systems*, vol. 4, 2005, pp. 3914 – 3917.

[8] M. Pankaala, A. Paasio, and M. Laiho, "Implementation alternatives of a DBNS adder," *9th International Workshop on Cellular Neural Networks and Their Applications*, 2005, pp. 138 – 141.

[9] C. Doche and L. Imbert, "extended double-base number system with applications to elliptic curve cryptography," LNCS 4329, pp. 335–348, Springer–Verlag Berlin/Heidelberg, INDOCRYPT 2006.

[10] K. Wangjitman and A. Surarerks, "Addition transducer for double base number system," *ISCIT'06*, 2006, pp. 994-999.

[11] J. Yoo, D. Lee, K. Choi and J. Kim, "A power and resolution adaptive flash analog-to-digital converter," *ISLPED'02*, 2002, pp. 233-236.

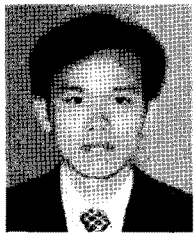
[12] D. Lee, J. Yoo, K. Choi, and J. Ghaznavi, "Fat tree encoder design for ultra-high speed flash A/D converters," *The 45th Midwest Symposium on Circuits and Systems*, vol. 2, pp. 87-90, Aug 2002.

[13] D. Lee, J. Yoo, and K. Choi, "Design method and automation of comparator generation for flash A/D converter," *IEEE International Symposium on Quality Electronic Design*, 2002, pp. 138-142.

[14] D. P. Dimitrov and T. K. Vasileva, "8-bit semi-flash A/D converter," *Proceedings of the International Conference Mixed Design of Integrated Circuits and System*, 2006, pp. 171-174.

[15] J. Kim, M.-H. Kim, and E.-H. Jang, "A new flash A/D

converter adopting double base number system," *KISPS Journal of Signal Processing and Systems*, vol. 9, no. 1, pp. 54-61, 2008.



Minh Son Nguyen
He received B.E and M.E degree in computer engineering from HCMcity University of Technology, Vietnam, in 2001 and 2005 respectively, and is working toward the Ph. D degree in electrical engineering at the University of Ulsan. His research interests are embedded system, VLSI, ASIC and signal processing.

Man-Ho Kim
He received his Ir degree from Delft University of Technology in Holland (1986-1989) and his Ph.D. from Leicester University in the UK (1992-1995). His research interests are the CCD/CMOS image sensor and Optical laser transmission system.

Jongsoo Kim
He received his Ph.D. from University of Alabama in Hunstville in 1994. Currently, he is a professor in the School of Electrical Engineering, University of Ulsan, Ulsan, Korea. His current research includes low power circuits design, mixed signal circuit design, and A/D converter design.