REDUNDANT LINEAR CODING FOR ACCELERATING COUNTING AND COMPARISON OPERATIONS

I. Elhanany[†], *O. Arazi*
Electrical and Computers Engineering Department
Ben-Gurion University
Beer-Sheva 84105, ISRAEL
Tel: 972-7-6289415; Fax: 972-7-6467956
[†]itamar@ieee.org

ABSTRACT

A novel binary number system, called Redundant Linear Coding (RLC), is presented. By eliminating the need for carry handling, a feasible tradeoff between speed and area yields short computation time for up/down counting and comparison operations. The scheme can efficiently be applied to a wide range of high-speed digital applications.

1. INTRODUCTION

High-speed up/down counters and comparison operations constitute the basic building blocks of many custom logic designs [1]. In various applications, such as fast digital correlators, counters that relate to different combinational logic blocks are recurrently compared. A common design goal is to shorten the critical path as much as possible. Incrementing (up-counting), decrementing (down-counting) and comparison operations require some form of carry propagation treatment [2-3] that becomes more complex as the counter size increases, hence limiting speed. Even when employing advanced carry manipulation techniques [3-6], the propagation delay of such operations is non-negligible. In an effort to accelerate the design, a tradeoff between speed and area is acceptable. In this paper, we introduce a novel binary number system, called Redundant Linear Coding (RLC), which offers high-speed incrementation, decrementation and comparison operations at the expense of larger area. Taking into consideration current VLSI technology, employing such a design tradeoff is pragmatic. It is further shown that the timing required to perform these operations is weakly affected by the counter size.

This paper is organized as follows. In section 2 the RLC concept is introduced. In section 3 counting and comparison arithmetic operations are described. Section 4 focuses on an RLC-to-binary conversion scheme and in section 5 the main conclusions are drawn.

2. REDUNDANT LINEAR CODING

In conventional binary number representation each digit is assigned a weight denoting a power of two, which corresponds to the digit's position. Consequently, 2^M combinations may be represented using an M-bit word. In RLC, the weight of each bit is the value of its position index. In such a representation it is quite evident that some values may be encoded in more than one binary combination. To avoid such ambiguity, we define the following coding discipline:

$$\underbrace{11\ldots1}_{L}\overset{N}{}\overset{R}{}00\ldots1\overset{1}{}00\ldots0 = \sum_{i=0}^{L-1}(N-i)+R \qquad (1)$$

A sequence of set bits is located at the most significant positions, while at most a single set bit is located in the remaining region. For example, using RLC the value 110010 equals 13 (6+5+2), similarly the value 111000 equals 15 (6+5+4). The word 110101 is prohibited over RLC since there are two set bits occupying the least significant region.

It is apparent that as a result of the coding scheme only a subset of the 2^N binary representations is permitted. Moreover, the number of values that are represented using such a technique is bounded by the sum of the sequence 1,2,…N (i.e. $N(N+1)/2$) which accounts for the sum of bit weights. Hence, the efficiency of the coding technique may be evaluated by determining the number of bits (N) required to represent an M-bit word coded in conventional binary coding:

$$\frac{N(N+1)}{2}+1 = 2^M \qquad (2)$$

The solution for N from the quadratic expression in (2) is:

$$N = \sqrt{2^{M+1}-7/4}-1/2 \approx 2^{\frac{M+1}{2}} \qquad (3)$$

Alternatively, if B represents the maximal counted value then the number of bits, N, required using RLC counter is approximately:

$$N \approx \sqrt{2}B^{1/2} \qquad (4)$$

Table 1 presents a comparison between binary and RLC coding with respect to word length, area and speed. As M increases, the efficiency of the method decreases. For example, an 8-bit

0-7803-5482-6/99/$10.00 ©2000 IEEE

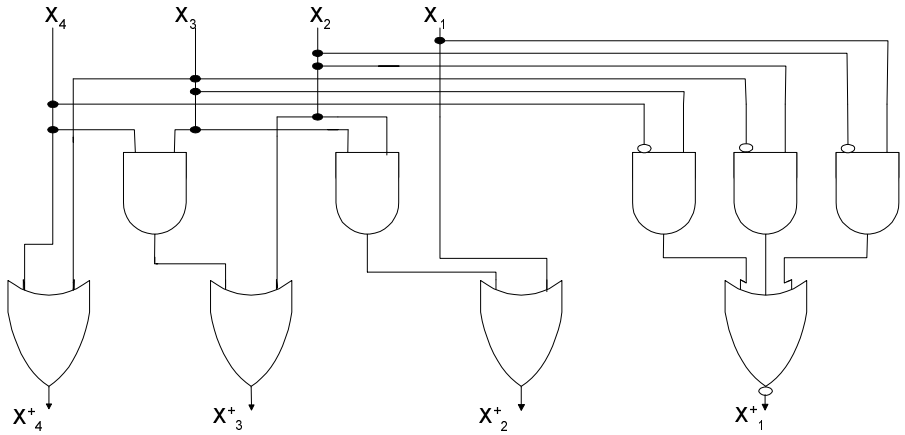

Figure 1. A 4-bit RLC Up-Counter.

counter requires 23 bits using RLC.

Binary	RLC
1	1
2	2
3	4
4	5
5	8
6	11
7	16
8	23

(a)

Method	Area	Speed
RLC	*O*(N)	*O*(1)
Binary	*O*(N)	*O*(logN)

(b)

Table 1: A comparison between RLC and conventional binary coding with respect to (a) word length and (b) area/speed requirements

3. RLC ARITHMETIC OPERATORS

3.1 Up-Counter

In conventional binary coding, up-counting (incrementing) consists of adding the value 1 to the current value. Many techniques for minimizing the logic complexity of counting with respect to addition have been proposed [1-3]. Down-counting (decrementing) introduces similar complexity. The rationale behind a comparison operation is that of subtracting the two compared values and examining the resulting sign bit. In general, the above operations all require a form of carry-propagation treatment that grows more complex with the counter size, thus limiting speed. Using RLC, these operations are easily obtained with no need for carry handling. There are two possible counter states that need to be addressed when incrementing over RLC:

(1) Locating a '01' sequence, if exists, and modifying it to '10'.

(2) When the sequence '01' does not exist, the least significant bit must be set.

Accordingly, the following equations are implemented in order to calculate the incremented value, X^+, of an N-bit RLC value:

$$\rho = \overline{\sum_{i=1}^{N-1} X_i \overline{X_{i+1}}}$$

$$X_i^+ = \begin{cases} X_{i+1}X_i + X_{i-1} & 2 < i < N\text{-}1 \\ \rho & i = 1 \\ X_N + X_{N-1} & i = N \end{cases} \qquad (5)$$

Since there are no dependencies between the resulting bits, they are all calculated concurrently. The critical path of the increment expressions is approximately that of 2 gate levels – an AND level and an OR level. It should be noted that the above counter is a non-modulo counter. Modulo up-counting can easily be accomplished by detecting the maximal value (all bits set) as a reset condition for all the bits.

As can be derived from (5), the expression for ρ, which indicates the existence of a '01' pattern, consists of *N-1* AND gates and a wide NOR gate which may be implemented using wired-OR CMOS technology to obtain high-speed. In figure 1, a 4-bit RLC counter is illustrated.

3.2 Down-Counter

Following a similar logic to up-counting, an RLC down-counter must address two possible counter states:

(1) Locating a single '10' sequence, if exists, and modifying it to '01'.

(2) When there are two '10' sequences the one positioned in the least significant locations must be altered to '01'.

The above is established by the following expressions:

$$X_i^- = \begin{cases} X_N\left(X_{N-1} + \overline{\rho}\right) & i = N \\ X_{N-2}\left(X_{N-1} + \rho\right) + X_N\left(\rho \oplus X_{N-1}\right) & i = N-1 \\ X_2\overline{X_1} & i = 1 \\ \rho\left(X_{i-1} + X_{i+1}\overline{X_i}\right) + \overline{\rho}X_{i+1}\left(X_i + \overline{X_{i+2}}\right) & 2 < i < N\text{-}2 \end{cases} \quad (6)$$

where ρ is defined in (5).

As with up-counting, the binary expressions for each of the bits are calculated concurrently. Modulo down-counting is achieved by detecting the minimal value (all bits valued '0') as a set condition for all the bits.

3.3 Comparison

In RLC, finding the larger of two values, A and B, is easily accomplished by performing the following steps:

(1) Calculate $C_i = A_i$ XOR B_i, where i denotes the bit's index.

(2) Obtain the binary vector containing at most a single set bit corresponding to the most significant bit set in C_i. Intuitively, the result is equivalent to a priority encoder denoting the first most-significant bit which is set in either A or B.

Let D_i be defined as:

$$D_i = \begin{cases} \left(\sum_{j=i}^{N-1}\overline{C_j}\right) A_i \overline{B_i} & 1 < i < N\text{-}1 \\ C_N & i = N \end{cases} \quad (7)$$

The comparison result becomes:

$$\begin{cases} A > B & \sum_{i=1}^{N} D_i = 1 \\ A \le B & else \end{cases} \quad (8)$$

Notice that A equals B if the following condition is met:

$$\overline{\sum_{i=1}^{N} C_i} = 1 \quad (9)$$

For example, let A=1101 and B=1010. We first calculate C as A XOR B = 0111 followed by calculating D to obtain D=0100. Since D is not a null vector we conclude that A>B.

4. CODE CONVERSION

In order to comply with common digital arithmetic circuits, RLC conversion to and from binary coding is required. Although it may be assumed that the occurrence rate of such a conversion is lower than that of counting or comparing operations, it should be kept efficient.

Since RLC is efficient for short counters, converting from binary to RLC may be attained using a simple lookup table or a ROM having M address lines and an N-bit word. It is apparent that an RLC word is comprised of a sequence of set bits positioned at the MSB locations, and optionally a single set bit located at an arbitrary position at the least significant bits. By extracting a constant term for each of the two components and adding the terms up using a conventional binary adder, the conversion is attained.

Let us define expressions for indicating the least significant term, α_i, and most significant term, β_i,as:

$$\alpha_i = \begin{cases} \overline{X_{i+1}}X_i & 1 < i < N\text{-}1 \\ X_i\overline{X_{i-1}} & i = N \end{cases} \quad (10)$$

$$\beta_i = \begin{cases} X_{i+1}X_i\overline{X_{i-1}} & 2 < i < N\text{-}1 \\ X_{i+1}X_i & i = 1 \end{cases}$$

Since at α_i and β_i are one-hot (i.e. contain a single set bit) these lines can act as decoding signals for extracting the two constant terms. By adding these two terms the conversion result is obtained. Although this method requires a relatively large area, the critical path is that of three gate levels and an M-bit adder. Figure 2 illustrates the conversion scheme.

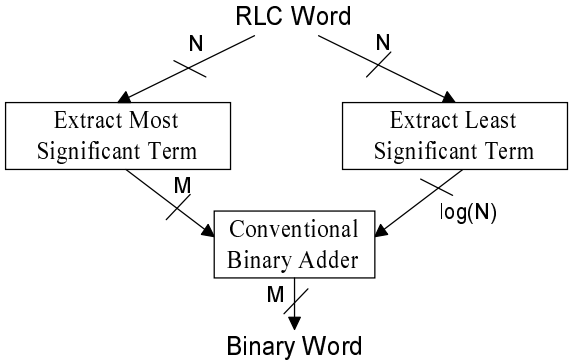

Figure 2. Block diagram of RLC-to-binary conversion.

5. SUMMARY

A new approach for accelerating arithmetic functions using a redundant binary number system, called Redundant Linear Coding (RLC) has been presented. By entirely eliminating the necessity for carry manipulation, optimized implementation for counting and comparison operations is achieved. In addition, a scheme for converting from RLC to binary has been illustrated. The method can easily be applied in order to accelerate a wide range of digital designs.

6. REFERENCES

[1] M. R. Stan, A. F. Tenca and M. D. Ercegovac, "Long and Fast Up/Down Counters," *IEEE Trans. Computers*, vol. 47, pp. 722-735, July 1998.

[2] D. C. Hendry, "Sequential Lookahead Method for Digital Counters," *Electronics Letters*, vol. 32, pp. 160-161, Feb. 1996.

[3] T.Y. Chang and M. J. Hsiao, "Carry-Select Adder Using Single Ripple-Carry Adder," *Electronics Letters*, vol. 34, pp. 2101-2103, Oct. 1998.

[4] P. Larsson and J.R. Yuan, "Novel Carry Propagation In High Speed Synchronous Counters and Dividers," *Electronics Letters*, vol. 29, pp. 1457-1458, 1993.

[5] I. Koren, *Computer Arithmetic Algorithms*. Englewood Cliffs, N.J.:Prentice Hall, 1993.

[6] D.R. Lutz and D.N. Jaysimha, "Programmable Modulo-k Counter," *IEEE Trans. Circuits and Systems,* voil. 43, pp. 939-941, Nov. 1996.