A SINGLE FAULT TOLERANT MULTIPLIER FOR RNS APPLICATIONS

A.P. Preethy and D. Radhakrishnan

Division of Computing Systems, School of Applied Science, Nanyang Technological University, Nanyang Ave., Singapore 639798
e-mail: asdrkrishnan@ntu.edu.sg

Abstract: With the ever-increasing complexity of VLSI chips more component failures are bound to occur. Devising test patterns to detect these failures and taking preventive measures for correcting them is becoming increasingly difficult. In this regard, RNS exhibits error detection and correction capabilities at the expense of only a few redundant channels. The area overhead is found to be reasonable when applied to a large circuit such as a multiplier, but is generally quite high for an adder. In this paper we present the design of a 36 bit multiplier with single error correction capability by taking advantage of the logarithmic properties of both prime and powers of prime moduli while limiting the hardware overhead for error correction to a few adder units.

1. INTRODUCTION

Despite how fast a system is, a single error in any one of the units can disable the whole system, and in some instances it may result in a catastrophic failure. So it is highly desirable to include a fault detection and correction mechanism in failure critical systems. RNS (Residue Number System) has unique capabilities with respect to error detection and correction which the conventional number systems lack [1]. In this paper, a novel and elegant error correction mechanism using an n-bit duet \((2^n,2^n-1)\) is proposed which is then successfully incorporated into an RNS multiplier using a 5-bit moduli set.

2. RNS OVERVIEW

An RNS is defined by a set of relatively prime integers (moduli) \(m_1, m_2, \ldots, m_n\). Each integer \(X\) in the range 0 to \(M-1\) for \(M = \prod m_i\) is uniquely represented by an \(n\)-tuple \((x_1, x_2, \ldots, x_n)\), where each residue \(x_i = X \mod m_i\) is defined as the least remainder when \(X\) is divided by the modulus \(m_i\), [1]. \(M\) is defined here as the range of the number system. It follows from the Chinese Remainder Theorem (CRT) that, for any given \(n\)-tuple satisfying the above relationships, there exists one and only one integer \(X\) such that \(0 \leq X < M\). The number \(X\) can be evaluated from the \(n\)-tuple \((x_1, x_2, \ldots, x_n)\) using the equation:

\[
X = \left( \sum_{i=1}^{n} \frac{m_i}{m_j} x_i \right) \mod M, \quad \text{where } \frac{m_i}{m_j} x_i \equiv 1 \mod m_j.
\]

Arithmetic operations on two operands \(X\) and \(Y\) are defined as: 
\(Z = X \circ Y\), where \(X = (x_1, x_2, \ldots, x_n)\), \(Y = (y_1, y_2, \ldots, y_n)\), \(Z = (z_1, z_2, \ldots, z_n)\), and \(z_i = x_i \circ y_i\) for \(i = 1, \ldots, n\). The symbol \(\circ\) denotes any of the operations of addition, subtraction or multiplication. From the above definition, it is seen that these operations are performed all in parallel in each of the residue channels, independent of one another.

Inherent parallelism and the carry-free arithmetic between different residue channels provide speed up during arithmetic processing.

3. MULTIPLICATION USING INDEX CALCULUS TECHNIQUES

Fast multiplication is one of the primary design goals in many of today’s processors. In RNS, efficient multiplication requires a choice of smaller moduli. For further speed improvement, logarithmic properties of prime fields are used, thus transforming multiplication into addition. But the choice of moduli limiting to primes restricts the dynamic range, and it necessitates the addition of extra moduli which are non-prime, without sacrificing the logarithmic properties. In this respect, an appropriate choice of moduli will be of the form \(p^m\), where \(p\) is a prime and \(m\) is any integer. The residue sets based on these moduli can be directly mapped onto integer rings modulo \(p^m\), denoted by \(Z(p^m)\), for odd and even values of \(p\). A procedure for finding an index set for the elements of \(Z(2^{m2})\) is given in [2,3]. Using this, any integer \(X \in \{1, 2^{m2}-1\}\) can be coded using a triplet index code \(<\alpha, \beta, \gamma>\) with the relationship

\[
X = 2^{\alpha \beta} \left(1\right)^\gamma \mod 2^{m2}, \quad \text{where } \alpha \in \{0,1,\ldots,m-1\}, \beta \in \{0,1,\ldots,\left(2^{m2}-1\right)\}, \gamma \in \{0,1\}.
\]

Multiplication of two integers can now be carried out as follows: let \(X, Y \in Z(2^{m2}), X \neq 0, Y \neq 0\), and

\[
X = 2^{\alpha X} \left(1\right)^{\gamma X}, \quad Y = 2^{\alpha Y} \left(1\right)^{\gamma Y} \mod 2^{m2},
\]

then the product

\[
XY \mod 2^{m2} = 2^{\alpha X+\alpha Y} \left(1\right)^{\gamma X+\gamma Y} \mod 2^{m2}.
\]

These indices are added subject to the following constraints:

\(\beta_1\) and \(\beta_2\) are added mod \(2^{m2}\), \(\gamma_1\) and \(\gamma_2\) are added mod 2, and \(\alpha_1\) and \(\alpha_2\) are added in normal binary mode. When \(\alpha_1 + \alpha_2\) equals \(m-1\) the corresponding \(\beta\) and \(\gamma\) are made zero, and when it exceeds \(m-1\), the final result is made zero. Thus, by storing the index and inverse index tables, multiplication of the nonzero elements can be replaced by index addition.
A solution to the general problem of finding an index code for each and every element of \( \mathbb{Z}/(p^m) \) for odd prime \( p \) is by using an index pair coding. The generation of these index pairs is given in [4,5]. In the case of \( \mathbb{Z}/(p^m) \), where \( p \) is odd, an index pair coding \((\alpha, \beta)\) is used, where \( X \) is given by \( X = (g^\alpha p^\beta) \mod p^m \) [4,5]. The product of two numbers \((X_1X_2) \mod p^m \) can now be calculated as follows:

Let \( X_1 = (g^\alpha_1 p^{\beta_1}) \mod p^m \) and \( X_2 = (g^\alpha_2 p^{\beta_2}) \mod p^m \), then their product \((X_1X_2) \mod p^m \) is given by:

\[
[X_1X_2]_{p^m} = (g^\alpha_1 p^{\beta_1} g^\alpha_2 p^{\beta_2}) \mod p^m
\]

The indices are added subject to the following constraints: \( \alpha_1 \) and \( \alpha_2 \) are added mod \( \phi(p^m) \), and \( \beta_1 \) and \( \beta_2 \) are added in normal binary mode. When \( \beta_1 + \beta_2 \) exceeds \( m-1 \), the final result is made zero. It may be noted that more than one index pair \( \langle \alpha, \beta \rangle \) generated during index addition may correspond to the same residue value.

From the above it can be seen that prime and powers of prime moduli emerge as ideal candidates for the design of index transform based multipliers. For the design of a 36 bit multiplier we decided to go with moduli of up to 5 bits in length. The prime moduli of length 5 bits include 17, 19, 23, 29, and 31, and powers of prime moduli include 25, 27 and 32. All these moduli together give an overall range of 36 bits.

### 4. ERROR DETECTION AND CORRECTION

Reliability in computation is very essential for the proper functioning of any system. RNS lends itself to reliable computation due to its concurrent error detection and correction properties. The addition of \( r \) redundant moduli to an otherwise non-redundant RNS makes it a redundant residue number system (RRNS). Thus RRNS has \( n+r \) relatively prime moduli \((m_1, m_2, ..., m_n, r)\). The moduli \((m_1, m_2, ..., m_n)\) are the fundamental or non-redundant moduli. The additional \( r \) moduli \((m_{n+1}, m_{n+2}, ..., m_{n+r})\) are called the redundant moduli. Thus in RRNS a number is represented by \( n+r \) residue numbers \((x_1, x_2, ..., x_n, r)\) mod \( m \) can now be calculated as

\[
X = (x_1, x_2, ..., x_n, x_{n+r}) \mod m
\]

or non-redundant moduli. The total range \((m_1, m_2, ..., m_n)\) is divided into two adjacent intervals - Legitimate Range, where \( 0 \leq x < m \), and Illegitimate Range, where \( x \geq m \). This total range is divided into two adjacent intervals - Legitimate Range, where \( 0 \leq x < m \), and Illegitimate Range, where \( x \geq m \).

Figure 1. The most significant field will be \( m \) mod \( l \) bits wide. It was shown in [7] that since \( b^{l-1} \equiv 1 \mod 2^{l-1} \), for a nonnegative integer value of \( t \)

\[
[X]_{b^{l-1}} = \sum_{i=0}^{b^{l-1}} b_i 2^{l-1}
\]

where \( b_i \) is the decimal value of \( B_i \), \( 0 \leq b_i \leq 2^{l-1} \), and the computation
of $|X|_{2^{l-1}}$ can be accomplished with a tree of $b-1$ modulo adders mod ($2^l-1$) (i.e. 1's complement adders).

5. SINGLE FAULT TOLERANT 36-BIT MULTIPLIER USING 5-BIT MODULI

From the discussion given in Section 3 it is apparent that prime and powers of prime moduli form ideal candidates for the design of index transform based multipliers, and a moduli set $\{7,11,13,17,19,23,25,27,29,31,32\}$ is chosen in the proposed design. Thus, in all the cases of prime moduli ($7,11,13,17,23,29,31$), and powers of prime moduli ($25,27,32$) multiplication can be easily carried out by using index and inverse index look-up tables. These moduli taken together give an overall range of 36 bits. 31 and 32 are used as redundant moduli.

The two redundant channels (31 and 32) in our proposed design make the system capable of single digit error correction. These redundant moduli are chosen as an n-bit duet ($2^n$ and $2^n-1$). Residues are generated in non-redundant channels as well as in the redundant channels as usual. But to reconstruct the output using CRT, only the non-redundant residues are made use of. Further, a new set of redundant residues are also generated from the reconstructed output using a forward conversion circuitry as shown in Figure 1. An error syndrome is obtained out of the two sets of redundant residues and the error is easily corrected using a small error correction look-up table implemented based on the error correction procedure given in Section 4.

The 36 bit multiplier incorporating single error correction mechanism is shown in Figure 2. The forward converters for both the operands X and Y generate the corresponding residues and feed them to the multiplier channels whose outputs (redundant channels excluded) are passed to a reverse converter to get the output Z. A new residue $|Z|_{31}$ is generated using a forward converter. This converter requires only a few 5 bit modulo adders. The 5 least significant bits of Z represent $|Z|_{32}$. Hence additional hardware is needed only for generating modulo 31 residue. These residues are compared with the residues $|Z|_{31}$ and $|Z|_{32}$ from the redundant channels to generate the syndromes $S_{31} = |Z|_{31} - |Z|_{31}$ and $S_{32} = |Z|_{32} - |Z|_{32}$.

The syndrome pairs $(S_{31}, S_{32})$ address the error table. The complementary error output '-e' from the error table adds to the direct output from the reverse converter and produces the correct output.

6. CONCLUSIONS

An efficient and elegant scheme for single error correction using an n-bit duet ($2^n$ and $2^n-1$) has been proposed, to incorporate fault tolerance into a 36-bit RNS-based multiplier using 5-bit moduli set. The extra hardware cost is only a few binary adders, and the design is apparently less complex compared to the conventional schemes using MRC based base extension approaches [6].

7. REFERENCES


