ELEX

IEICE Electronics Express, Vol.* No.*,*-*

Area-delay efficient arithmetic mixed-radix conversion for Fermat moduli

Anne B. O'Donnell [a)], Chris J. Bleakley and Seamas McGettrick

School of Computer Science and Informatics, University College Dublin, Belfield, Dublin 4, Ireland

a) anneodonnell@eircom.net

Abstract: Mixed Radix Conversion is an essential feature of error detection and correction in Redundant Residue Number Systems. Fermat numbers are a popular choice as moduli in these systems. However, Fermat numbers are typically implemented using Diminished–1 arithmetic which necessitates special consideration of zero in arithmetic operations. Furthermore, the sequential nature of Mixed Radix Conversion leaves it prone to considerable delay due to carry propagation in adders at each stage. In this paper, Diminished-1 arithmetic in carry save form is used to give significant reductions in area and delay compared to previously proposed hardware architectures. The percentage area reduction was found to range from 13% to 41% and that of delay from 19% to 49% for bit widths from 8 to 28 bits.

Keywords: Fermat numbers, Mixed-Radix Conversion, Error Detection

Classification: Integrated circuits

References

[1] N.W. Szabo, and R.I. Tanaka, Residue Number Arithmetic and its Application to Computer Technology, McGraw Hill, New York, 1967.

[2] W.K. Jenkins, and E.J. Altman: "Self-checking properties of residue number error checkers based on Mixed Radix Conversion," Circuits and Systems, IEEE Transactions on, vol. 35, no. 2, pp. 159–167, Feb., 1988.

[3] A.B. O'Donnell, and C.J. Bleakley, "Area efficient fault tolerant convolution using RRNS with NTTS and WSCA," Electronics Letters, vol. 44, no. 10, pp. 648–649, May, 2008.

[4] L.M. Leibowitz, "A simplified binary arithmetic for the Fermat number transform," Acoustics, Speech and Signal Processing, IEEE Transactions on, vol. 24, no. 5, pp. 356–359, Oct., 1976.

[5] G.A. Jullien, "Residue number scaling and other operations using ROM arrays," Computers, IEEE Transactions on, vol. C-27, pp. 325–336, Apr., 1978.

[6] S.G. Li, and J. Zhang, "Area-delay efficient parallel architecture for Fermat number transform, " IEICE Electronics Express, vol. 6, no. 14, pp. 449-455 Apr., 2009.

[DOI: **.****/**********]
Received September **, 2008;
Accepted September **, 2008;
Published in October **, 2008 issue

[7] J. Mathew, D. Radhakrishnan and T. Srikanthan, "New area efficient residue-to-weighted number system converters," Proc. IEEE Int. Conf. Electronics, Circuits and Systems, ICECS, Pafos, Cyprus, vol. 2, pp. 945–948, Sept., 1999.

[8] H.T. Vergos, D. Bakalis, C. Efstathiou and D. Nikolos, "Efficient modulo 2n+1 multi-operand adders," Proc. IEEE Int. Conf. Electronics, Circuits and Systems, ICECS, St. Julien's, Malta, pp. 694 – 697, Sept., 2008.

[9] C. Efstathiou, H.T. Vergos, and D. Nikolos, "Fast parallel-prefix modulo 2n+1 adders," IEEE Trans. Computers, vol. 53, no. 9, pp. 1211–1216, Sep., 2004.

1. Introduction

A Residue Number System (RNS) is a non-weighted number system that allows fast, parallel, carry-free arithmetic [1]. A Mixed Radix System (MRS) is a weighted number system that is derived from the set of moduli in a RNS and is commonly used to perform error detection in a Redundant Residue Number System (RRNS) [2]. A RRNS error checker based on Mixed Radix Conversion (MRC) is attractive in that it is self correcting for errors (or faults) both in the RRNS and in the MRC hardware [2]. The use of Fermat numbers as moduli for RRNSs is desirable since the residues can be efficiently represented and processed using Diminished 1 (Dim1) arithmetic [3], but with the drawback that zero detection is required at the beginning and end of Dim1 arithmetic operations [3]. MRC is a sequential algorithm with subtraction and multiplication at each stage [1]. Therefore zero detection logic has a significant area cost in cell based architectures for Fermat number MRC. Furthermore, carry propagation at each stage leads to significant delay overhead. Look Up Table (LUT) based architectures avoid the zero detection problems of cell based architectures but have significantly larger area [4]. Herein, a cell based architecture for Fermat number MRC is proposed which provides reduced area and delay compared to previous proposals.

2. Background

A Residue Number System (RNS) consists of a set of pairwise relatively prime moduli, $\{m_1,\ldots\ldots,m_N\}$. An integer, X, is represented in RNS by a set of residue digits $\{r_1,\ldots\ldots,r_N\}$, where $r_i = X \bmod(m_i)$ is the remainder when X is divided by m_i and N is the number of relatively prime moduli in the system [1]. This results in a non-weighted number system that allows for parallelism in calculations. However, its non-weighted nature makes it unsuitable for overflow detection, sign detection and magnitude comparison. The Mixed Radix System is a weighted number system which is derived from the set of moduli in the RNS and can perform these operations as well as error detection in a RRNS. The two systems have the same range of values. The number X is expressed as:

$$\text{RNS: } X = \{r_1, r_2, ..., r_n\}; \qquad \text{MRS: } X = a_N \prod_{i=1}^{N-1} m_i + ... a_3 m_1 m_2 + a_2 m_1 + a_1$$

where m_i are the moduli and a_i are the mixed radix digits (MRD). The values of a_i

[DOI: **.****/**********]
Received September **, 2008;
Accepted September **, 2008;
Published in October **, 2008 issue

are calculated sequentially as follows [1]:

$$a_1 = r_1$$
$$a_2 = \left|(r_2 - a_1)\left|m_1^{-1}\right|_{m_2}\right|_{m_2} \ldots \quad (1)$$
$$a_n = \left|\left((r_n - a_1)\left|m_1^{-1}\right|_{m_n} - a_2\right)\left|m_2^{-1}\right|_{m_n} - \ldots . - a_{n-1}\left|m_{n-1}^{-1}\right|_{m_n}\right|_{m_n}$$

This algorithm involves $N(N\text{-}1)/2$ subtractions and $N(N\text{-}1)/2$ multiplications, where N is the number of moduli. Fermat numbers are of the form $2^{2^t}+1$ where t is an integer [3]. In weighted binary arithmetic, $n+1$ bits are needed to represent a Fermat number of the form 2^n+1. Dim1 arithmetic [3] works by decreasing the weighted binary representation by one. This requires a special treatment of zero at the input and output of arithmetic operations, making Dim1 problematic for arithmetic intensive operations such as MRC. A carry save implementation of Dim1 is described in [6] which overcomes the zero detection problem. This concept has been used herein to build an MRC architecture with less area and delay than existing cell based implementations. Traditionally, MRC has been implemented using (LUTs). In [7] an MRC is introduced for general moduli sets where LUTs are replaced entirely by arithmetic units with a significant reduction in area. The architecture uses 2's complement arithmetic and multi-operand modular adders(MOMA). Herein, an MRC architecture without LUTs is established for Fermat moduli, using Dim1 carry save arithmetic. The architecture is of lower area and is also faster than that which would be achieved for Fermat moduli using the architecture described in [7].

3. Proposed Architecture

Herein, we present an area-delay efficient MRC for sets of Fermat moduli in a RNS. Fig. 1 shows the architecture, based on Dim1 carry save arithmetic, for the proposed system for three Fermat moduli. The Add/Conv unit in Fig. 1 is made up a parallel-prefix modulo 2^n+1 adder [9], followed by logic to convert to ordinary binary. The Add_Inv unit is used to get the additive inverse of a residue in one Fermat modulus, m_i modulo another Fermat modulus, m_j. The 4:2 Comp unit represents a 4:2 compressor [6] which takes four Dim1 inputs and produces two outputs which in this case are the carry save equivalent of r_j-r_i modulo m_j. The Mult_Inv unit performs multiplication by the inverse of m_i modulo m_j. The first MRD a_1 is produced by adding c_1 and s_1 in the Add/Conv unit. The additive inverses (Add_Inv) of c_1 and s_1 are fed into a modulo 4:2 compressor [6] along with c_2 and s_2 to produce $\left|(r_2 - r_1)\right|_{m_2}$. The Mult_Inv unit performs multiplication by $\left|m_1^{-1}\right|_{m_2}$ and finally a_2 is produced in the Add/Conv unit. Similarly, a_3 is calculated according to (1), where all operations are carried out using Dim1 carry save representation. Carry propagate addition is needed only in the final Add/Conv in Fig. 1. If the residue inputs, r_i, are in weighted binary form they can be represented in Dim1 carry save form as s_i and c_i where s_i is the ordinary binary

[DOI: **.****/**********]
Received September **, 2008;
Accepted September **, 2008;
Published in October **, 2008 issue

form of the input and c_i is a vector consisting of n_i 1s. For example, if the input, r_i is the number 13 represented in 4-bit 2's complement with a carry-in as 01101, then s_i=1101(=r_i) and c_i =1111 which is the carry save Dim1 representation of 13mod17. An execption is 16mod17, where the input is 1 0000 and in this case, s_i= 0000 and c_i =1110. The only logic required for this is a NOT gate because the least significant bit of c_i is always the negation of the carry-in and the remaining bits of c_i can be set to the value 1.

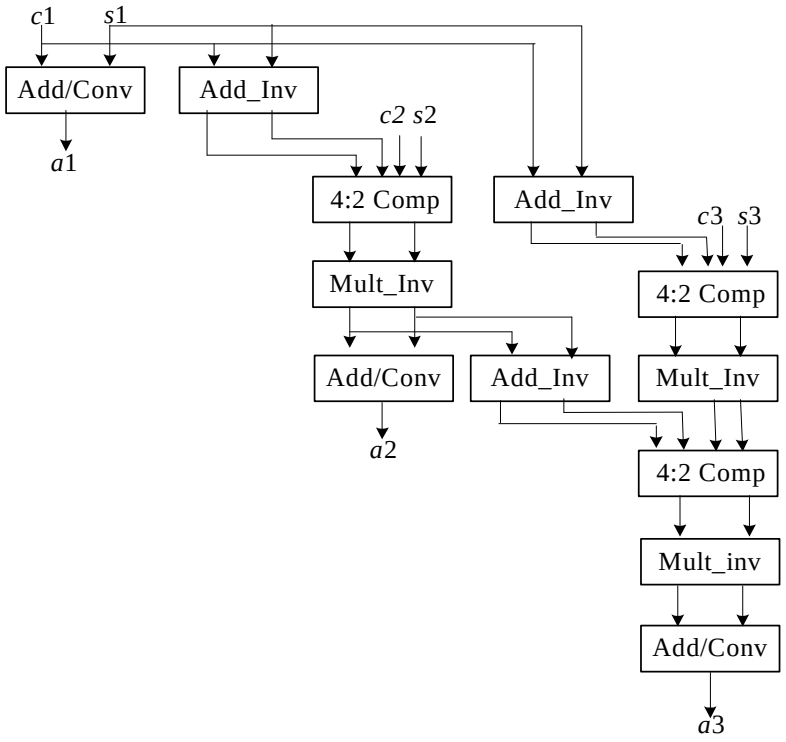

Fig 1. The proposed MRC System using Dim1 carry save arithmetic.

4. Results

The architecture described in [7] and the proposed architecture were implemented and verified in Verilog at the gate level for the Fermat moduli sets given in Table I. The gates in the verilog implementation were counted and in most cases formulae were derived for the area and delay of the circuits. The area and delay are expressed as gate equivalents (GE) based on [8], where each two-input gate, excluding XOR, accounts for one GE for both area and delay. An XOR gate accounts for two GEs and also, herein an inverter is taken as a half GE for both area and delay. Both systems consist of subtractions and multiplication by modular inverses. In [7], these are implemented in 2's complement arithmetic, whereas in the proposed system they are implemented in diminished 1 carry save arithmetic. In [7] Multioperand Modular Adders are used whereas the proposed system uses modulo 4:2 compressors. Diminished 1 arithmetic, typically requires zero detection logic at the input and output of arithmetic operations. However, in the proposed architecture, the use of carry save means that this is required only at the final output. Details of the area and delay comparison for the two implementations are

[DOI: **.****/**********]
Received September **, 2008;
Accepted September **, 2008;
Published in October **, 2008 issue

now given in GE, for the set of three Fermat moduli, $2^{n_i}+1,2^{n_j}+1,2^{n_k}+1$ where r_i, r_j and r_k refer to residues modulo these Fermat numbers respectively.

4.1 Area Comparison

The area of the system in [7] is calculated as follows: combinational logic apart from adders has a total area of $7n_i+6n_j+5n_k+18$, the area of a k-operand multi-operand adder modulo 2^n+1, MOMA(k,n)[8] is $17\lceil k/2\rceil n+6\lceil k/2\rceil+9n/2\log_2(n)-11n/2+5$. The three moduli system needs one MOMA(n_j+1, n_j) and two MOMA(n_k+1,n_k). This gives a total area of $\frac{1}{2}[(34n_j+12)\lceil(n_j-1)/2\rceil+(68n_k+24)\lceil(n_k-1)/2\rceil+(9\log_2(n_j)+35)n_j+(18\log_2(n_k)+56)n_k+14n_i+102]$ for the implementation of the architecture in [7]. For the proposed system, shown in Fig. 1, the implemented area is as follows: the Add/Conv unit is $9n/2\log n+10n+8.5$, the Add_Inv is $3.5n_i+n_j+1.5$ and the 4:2 Comp is $12n+1$GE. It is not possible to get a general formula for the area of the Mult_Inv units as they depend on the individual moduli and inverses. For the moduli sets in Table I, the Mult_Inv areas in GEs are 156, 615, 2301, 2103 and 1445. The area of the remaining units in Fig. 1 for the moduli set $\{2^{n_i}+1,2^{n_j}+1,2^{n_k}+1\}$ is $(17+9/2\log_2(n_i))n_i+(53/2+9/2\log_2(n_j))n_j+(36+9/2\log_2(n_k))n_k+33$. The areas for both systems, based on these formulae and verified by a count of gates in the implemented verilog, are given in Table I for various sets of three Fermat moduli covering bit widths from 8 to 28 bits. This table also gives the percentage savings achieved by the proposed system compared to that given in [7].

Table I. Area results based on formulae and verified by implementation

Fermat Moduli set	Bit width	[7]Area (GE)	Proposed System Area(GE)	%saving
{3,5,17}	8	622	448	28
{5,17,257}	14	1,895	1,229	35
{5,17,65537}	22	5,791	3,383	42
{5,257,65537}	26	6,353	3,363	47
{17,257,65537}	28	6,367	2,766	57

4.2 Delay Comparison

The delays for the various cells in the implemented architecture for both systems are calculated for the same moduli set, $2^{n_i}+1,2^{n_j}+1,2^{n_k}+1$. The delay of the implemented n-bit parallel prefix Diminished 1 (PPdim1) adder based on [9] is $7/2\log_2 n+5.5$. The MOMA which includes a PPdim1 adder has a delay of $4\theta+7/2\log n+9$, where θ is the number of stages in the Dadda tree in the MOMA. The delay of the surrounding logic is a constant 3GE. The delays for the units in the proposed carry save diminished 1 system are $7/2\log n+8$ for Add/Conv; $\log_2 n_i+4.5$ for Add_Inv; a constant delay of 8 gates for 4:2 Comp. As in the analysis of area, it is not possible to represent the delay of the Mult_Inv unit of the carry save system by a formula as the delay depends on the moduli involved and their relationships. The critical path was determined in the case of the moduli set $\{3,5,17\}$, with $n_1=1$, $n_2=2$ and $n_3=4$, for the system in [7] and for the proposed system. The path in the case of [7] contains two multiplications by modular inverses and two MOMAs which gives a delay of $8\theta+7\log_2 n_3+24$,=70GE where $\theta=4$. The path of the proposed system also has two multiplications by modular inverses as well as one additive inverse. The MOMAs are replaced by Modular 4:2 compressors and there is a final parallel prefix diminished 1

[DOI: **.****/**********]
Received September **, 2008;
Accepted September **, 2008;
Published in October **, 2008 issue

adder followed by a conversion to ordinary binary. The delay of the proposed system is $\log_2 n_1+7/2\log_2 n_3+53.5=60.5$GE. This gives a saving of 13.57% for the delay of the proposed system compared to that presented in [7].

4.3 Synthesis Results

The circuits were synthesised using Synopsys Design Compiler Vc2009.06 SP3. The designs were targeted at standard cell 130 µm technology. The post synthesis area and delay results for the two systems are shown in Table II for various sets of three Fermat moduli covering bit widths from 8 to 28 bits.

Table II. Synthesis results for area and delay and % reductions

Fermat Moduli set	Bit width	[7] Area (μm^2)	Proposed System Area (μm^2)	% saving	[7] Delay (ns)	Proposed System Delay (ns)	% Saving
{3,5,17}	8	1,953	1,694	13	14.6	11.8	19.2
{5,17,257}	14	6,584	5,081	23	24.7	16.0	35.2
{5,17,65537}	22	18,514	13,769	26	38.8	21.3	45.1
{5,257,65537}	26	19,327	13,404	31	39.4	20.0	49.2
{17,257,65537}	28	19,697	11,607	41	38.5	19.7	48.8

4.4 Discussion of Results

The area and delay of the proposed system are significantly less than that of the previous system [7]. This is due to the inclusion of following improvements. Firstly, Dim1 carry save arithmetic is used. This replaces MOMAs with modulo 4:2 compressors whereby carry propagation addition is required only at the output of the MR digits. Secondly, the additive inverse of a Fermat number in a given base, m_i is formed as its additive inverse in a different Fermat base, m_j by adding a known constant equal to $m_j - m_i$. This is highly suitable for Dim1 carry save arithmetic. Thirdly, Mult_Inv in Fig. 1 involves multiplication by a known constant which is input in ordinary binary form and the multiplication area is reduced by Booth encoding. The resulting partial products are in Dim1 carry save form and are combined using modulo 4:2 compressors with outputs also in carry save form thus eliminating carry propagation at the end of the multipliers. Fourthly, the intermediate outputs in the calculation of the third digit in the proposed implementation can be left in carry save form thus avoiding an addition before multiplication by the inverse. This is a major advantage of the proposed implementation over [7] in terms of reduction in both area and delay and the improvement becomes greater as the number of moduli in the system increases.

5. Conclusion

An area and delay efficient architecture for MRC is introduced for Fermat moduli giving significant area and delay reductions compared to the most efficient known hardware implementation. The percentage area reduction ranges from 13% for the conversion of an 8 bit number to 41% reduction for a 28 bit number. Reductions of up to 49% were found in the delay of the system. This was achieved by using Dim1 arithmetic in carry save form which avoids special zero detection logic and requires carry propagation addition only at the final output of the Mixed Radix digits.

[DOI: **.****/**********]
Received September **, 2008;
Accepted September **, 2008;
Published in October **, 2008 issue