PIPELINED HOGENAUER CIC FILTERS USING FIELD-PROGRAMMABLE LOGIC AND RESIDUE NUMBER SYSTEM

A. Garcia

Department of Electronics and
Computer Technology
University of Granada, 18071 Granada, Spain
e-mail: grios@ditec.ugr.es

U. Meyer-Baese and F. Taylor

High Speed Digital Architecture Laboratory
University of Florida
Gainesville 32611-6130, U.S.
e-mail: {uwe,fjt}@alpha.ee.ufl.edu

ABSTRACT

Field-Programmable Logic (FPL) is on the verge of revolutionizing digital signal processing (DSP) in the manner that programmable DSP microprocessors did nearly two decades ago. While FPL densities and performance have steadily improved to the point where some DSP solutions can be integrated into a single FPL chip, they still have limited use in high-precision high-bandwidth applications. In this paper it is shown that in such cases, the residue number system (RNS) can be an enabling technology. The design of a high-decimation rate digital filter is presented which demonstrates the RNS-FPL synergy.

1. INTRODUCTION

There are currently three classical viable DSP ASIC technology development paths (custom VLSI, standard cells, and gate arrays) and FPLs. Each represent a different performance and economic statement. FPLs come in two flavors: field programmable gate arrays (FPGAs) and complex programmable logic devices (CPLDs). FPGAs are fine grain devices consisting of small logic elements (LEs) (e.g., Xilinx XC4000) and various different routing canals (short, local, and long-lines). CPLDs have larger blocks and fast busses between this array blocks (e.g., Altera FLEX [1]). The historical advantage of FPLs has been their "in circuit programmability" and support of "rapid prototyping." FPLs have been promoted in custom computing machine (CCMs) applications where they have been reported to achieve speed-up-factors ranging from 10-1000 compared with conventional workstations [14, Table 1]. Recently ASICs have been used to develop DSP [7]. FPLs provide DSP arithmetic support with fast carry chains (Xilinx XC4000, Altera FLEX) which are used to implement multiply-accumulates (MACs) at relatively high speeds. A modern 200 MHz TMS320C60 DSP microprocessor with two multiplier cores, for example, can realize a 16 tap FIR filter having a throughput of $2 \cdot 200/16 = 25$ MSPS. A FPL can be designed to implement the same 16-bit FIR running at 100 MSPS. In those cases where FPL have been used to develop DSP ASIC (baseline digital filters and transforms), the designs have typically exploited latent:

- parallelism: implementing multiple MAC calls
- efficiency: zero product-terms are removed
- pipelining: each LE has a register, therefore pipeline requires no additional resources

FPLs, nevertheless, currently have limited applicability in high-bandwidth high-precision applications. The geometric increase in area requirements for constant-speed arithmetic is a very challenging FPL design issue. Unless this barrier is overcome, FPL will probably remain restricted to use in baseline DSP filter and transform applications having only modest bandwidth (100 MHz) and precision (16-bits) targets.

A. Garcia was supported by the Spanish General Direction of Higher Education under project PB96-1397, and a PNFPI research grant from the Spanish Ministry of Education.

U. Meyer-Baese was supported by a European Space Agency fellowship.

2. RESIDUE NUMBER SYSTEM (RNS)

The silicon area associated with a constant-speed fixed-point MAC unit is generally considered to geometrically increase with wordlength. The antithesis it the Residue Number System (RNS) which has a linear relationship between MAC silicon area and speed [11, 12]. The RNS therefore provides an opportunity to overcome the precision barrier in high-performance FPL applications. The RNS mechanics are well understood. RNS integer arithmetic is performed concurrently in parallel non-communicating small wordlength channels. An RNS system is defined in terms of a basis set $\{m_1, m_2, \ldots, m_L\}$ of relatively prime positive integers. The dynamic range of the resulting system is $M = \prod_{l=1}^{L} m_l$. RNS arithmetic is defined with respect to the ring isomorphism

$$\mathbb{Z}_M \cong \mathbb{Z}_{m_1} \times \mathbb{Z}_{m_2} \times \cdots \times \mathbb{Z}_{m_L} \tag{1}$$

Specifically, $\mathbb{Z}_M = \mathbb{Z}/(M)$ which corresponds to the ring of integers modulo M. The mapping of an integer X into the RNS is defined to be the L-tuple $X = (x_1, x_2, \ldots, x_L)$ where $x_l = X \mod m_l$, for $l = 1, 2, \ldots L$. Defining $\Box$ to be either the algebraic operation $+, -$ or $*$, it follows that if $0 \leq Z < M$, then

$$Z = X \Box Y \mod M \tag{2}$$

is isomorphic to $Z = (z_1, z_2, \ldots, z_L)$ where

$$z_l = x_l \Box y_l \mod m_l \qquad l = 1, 2, \ldots, L \tag{3}$$

Here it is evident the RNS arithmetic is performed in parallel within channels whose wordwidth is bounded by $w_l = \lceil \log_2(m_l) \rceil$ where typically $w_l \leq$ 8-bits. In practice, most RNS arithmetic systems use small RAM or ROM tables to implement the modular mappings $z_l = x_l \Box y_l \mod m_l$.

A historical problem with the RNS implementing RNS-to-integer decoding, division, and/or magnitude scaling in hardware [6]. Two methods are commonly used to convert the RNS L-tuple into the integer X [11]. The mixed radix conversion (MRC) produces division rest zero to find a weighted representation of the RNS tuple. The Chinese Remainder Theorem (CRT) is a direct conversion given by

$$X = \left(\sum_{l=1}^{L} M_l \langle M_l^{-1} x_l \rangle_{m_l} \right) \mod M \qquad (4)$$

where $M_l = M/m_l$. Typically, the desired output dynamic range is much smaller than M. A highly efficient algorithm called the $\varepsilon-$CRT [3] can be used to implement a time and area efficient CRT.

RNS systems have been built as custom VLSI devices [10], GaAs, and LSI [12]. It has already be shown that for a small wordlengths, the RNS can provide a significant speed-ups [13] using the $2^4 \times 2$ bit tables found in a Xilinx XC4000 FPGAs. For larger moduli, the $2^8 \times 8$ bit tables belonging to the Altera FLEX CPLDs are beneficial in designing RNS arithmetic and RNS-to-integer converters. With the ability to support larger moduli, the design of high-precision FPL systems becomes a practical reality. An area of DSP which requires both high-speed and high-precision is RF-to-baseband conversion. An RF-to-baseband converter, sometimes called a channelizer, is a steep-skirt high-decimation rate filter which accepts data in the high MHz range and exports a low-frequency baseband signal. Such systems are often architected as a Hogenauer CIC-filters [5].

3. THE HOGENAUER CIC-FILTERS

The Cascade Integrator Comb (CIC) (a.k.a., Hogenauer filter) have proven to be an effective element in high decimation or interpolation systems. Examples of highly oversampled systems include channelizers and converters. Figure 1(a) shows a three stage CIC filter which consists of a three stage integrator, a sampling rate reduction by R, and a three stage three comb. The transfer function of a general CIC system of S stages is

$$H(z) = \left(\frac{1 - z^{-RD}}{1 - z^{-1}} \right)^S \qquad (5)$$

It can be seen that S poles at zero frequency (DC) are annihilated by S zeros. The result is that the transfer-function is that of an S stage "moving average" filter. The maximum dynamic range growth occurs at DC-frequency (i.e., $z = 1$) and has a value of $B_{\text{grow}} = (RD)^S$. In bits, the maximum dynamic range growth is $b = \log_2 B_{\text{grow}}$. This value, in practice, can be substantial as evidenced by the 56 bit dynamic range of the Harris HSP43220 [4] channelizer. This fact, along with a high-bandwidth requirement, motivates the use of RNS to implement CIC filters.

A 3 stage RNS CIC filter with an input wordwidth of 8 bits, along with $D = 2$, $R = 32$, or $DR = 2 \cdot 32 = 64$, would require an internal wordwidth of $W = 8 + 3\log_2(64) = 26$ bits to insure that run-time overflow would not occur. The output wordwidth would normally be a value significantly less then W, 10 bits for example. Hogenauer [5] noted by careful analysis that some of the lower significant bits form early stages can be eliminated without sacrificing system integrity. Figure 2 shows

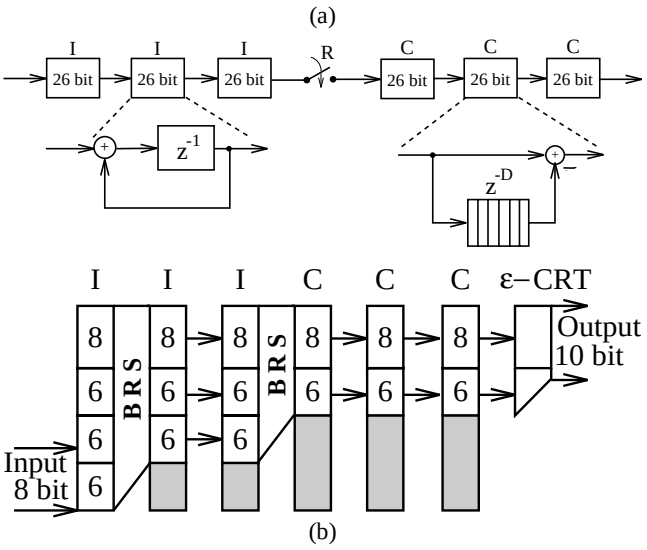

Figure 1: CIC filter. (a) Each stage 26 Bit. (b) Detail design with base removal scaling (BRS).

the transfer function for the realization with full wordwidth in all stages and by using the "pruning" suggested by Hogenauer. If the ratio of signal bandwidth to sampling frequency is, for instance $1/32$, then the aliasing suppression is 89.6 dB and the passband attenuation is 0.17 [5, Table I+II].

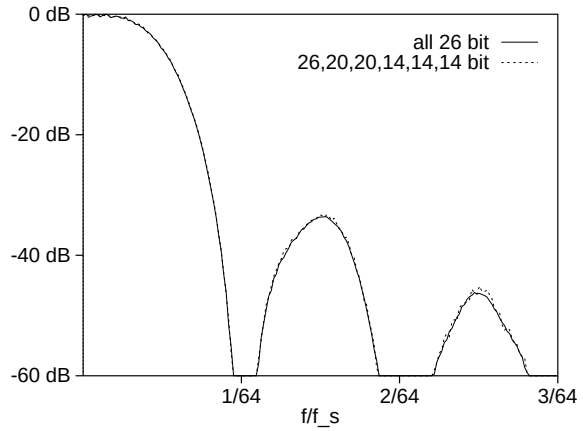

Figure 2: CIC transfer function (f_s is sampling frequency at input).

4. IMPLEMENTING (MODULO) ADDERS AND ACCUMULATORS WITH CPLDS

The implementation of DSP building blocks can be made with graphical design entry CAE tools or in VHDL. For Altera CPLD, the VHDL description provides more flexibility and better optimization results. For the VHDL design the structural (i.e. component instantiation) or behavioral description gives similar results, but the structural designs produce synthesized results which are easier to post-optimized by the design engineer.

Each logic block of a Altera FLEX 10K devices has 8 LEs,

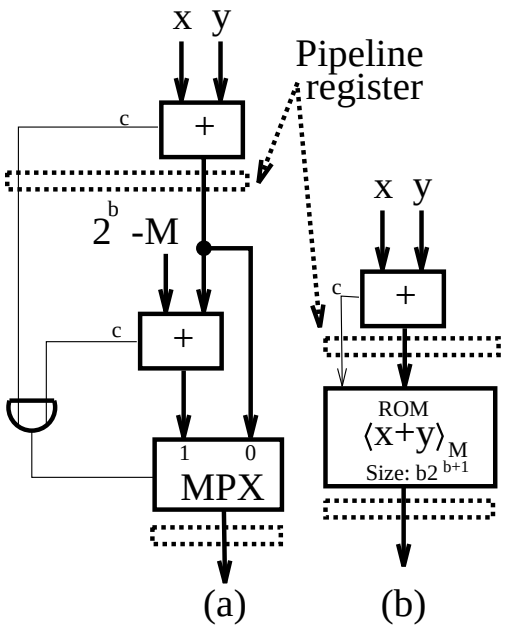

Figure 3: Modular addition with CPLD. (a) MPX-Add and MPX-Add-Pipe. (b) ROM-Pipe.

and each LE provides a $2^3 \times 1$ table and fast carry chain support in an arithmetic mode [1]. A EPF10K250A devices has for example 1520 LE blocks, 12160 LEs, and 20 2K tables (called embedded array block, EAB). For implementing high-speed adders, it should be noted that LE tables are much slower ($\approx$ 3ns) than a fast carry chain ($<$ 0.5ns). Therefore "fast adder" appearing in the literature, such as the carry save, carry select, or carry look-ahead adder [8], give *slower* bandwidth solutions in FPLs than using a ripple carry architecture with fast carry chains. The following table shows two's complement adder data for 4ns devices where speed is measured in mega samples per second (MSPS), including the pipeline register delay, and the number of required logic elements (LE).

Bits	8	16	26	32
MSPS	136.9	72.5	50.5	45.0
# LE	8	16	26	32

A wide variety of *modular* addition designs exist [2]. Using LEs only, the design of Fig. 3(a) is viable for FPLs. The Altera FLEX CPLD contains a small number of 2K Bit ROMs or RAMs (EABs) which can be configured as $2^8 \times 8, 2^9 \times 4, 2^{10} \times 2$ or $2^{11} \times 1$ tables which can be used for modulo m_l correction (Fig 3(b)). The next table shows data for 6, 7, and 8-bit modulo adder.

Bits	6	7	8
MPX-Add	41.3 MSPS 15 LE	44.6 MSPS 18 LE	33.6 MSPS 19 LE
MPX-Add-Pipe.	76.3 MSPS 16 LE	62.5 MSPS 18 LE	60.9 MSPS 20 LE
ROM-Pipe	70.4 MSPS 7 LE 1 EAB	70.4 MSPS 8 LE 1 EAB	60.6 MSPS 9 LE 2 EAB

Although the ROM shown in Fig 3 provides high-speed, the ROM itself produces a four cycle pipeline delay and the number of ROMs is limited. ROMs, however, are mandatory for the scaling schemes discussed in the next section. The multiplexed-adder (MPX-Add) has comparatively a reduced speed even if a carry chain is added to each column. The pipelined version usually needs the same number of LEs as the un-pipelined version but runs about twice as fast. Maximum throughput occurs when the adders are implemented in two blocks (where each block has 8 LEs for Altera FLEX 10K devices) within six-bit pipelined channels.

The additional pipeline delay of the modulo adder corresponds to a non-recursive transfer function $A(z) = z^{-2}$ which introduces no significant processing problem. The accumulator, however, is recursive and the addition delay introduces a second pole at half the sampling frequency (i.e., π [9, Fig. 1]), because the transfer function of the pipelined accumulator satisfies $F(z) = z^{-2}/(1 - z^{-2})$. The pole at π can be compensated by a (modulo m_l) comb with delay one (i.e., $G(z) = (1 - z^{-1})z^{-2}$). The integrator section, with pole compensation, becomes $F(z) \cdot G(z) = z^{-4}/(1 - z^{-1})$ as desired. In a high decimation CIC application it can be assumed that an anti-aliasing filter provides sufficient suppression of signal components near π, therefore a second passband at π, as introduced by the recursive pipelined accumulator, will introduce no additional aliasing. Therefore, six-bit wide pipelined accumulators are selected for use *without* pole compensation.

5. A THREE STAGE CIC DESIGN EXAMPLE

As a design example, a three stage CIC filter with 8-bit input, 10-bit output, $D = 2$, and $R = 32$ is implemented. As shown in Section 3, the total bit width is 26-bits. For the RNS implementation, a 4 modulus set $(256, 63, 61, 59)$ (i.e., one 8-bit two's complement (TC) and three 6-bit moduli), was chosen. The output scaling of the RNS system is implemented using the ε-CRT, which costs 8 tables and 3 TC adders [3, Fig. 1], or (as shown in Fig. 4) with a base removal scaling (BRS) algorithm based on two 6-bit moduli (which occur in the same fashion in the mixed radix conversion scheme [6]) and a ε-CRT of the remaining 2 moduli, which uses a total of 5 modulo adder and 9 ROM tables, or 7 tables if the multiplicative inverse ROM and the ε-CRT are combined. The following table shows speed in MSPS and used LEs and EABs for the three scaling schemes.

Type	ε-CRT	BRS-ε-CRT (Speed data for BRS m_4 only)	BRS-ε-CRT combined ROM
MSPS	58.8	70.4	58.8
#LE	34	87	87
#Table (EAB)	8	9	7

The decrease in speed to 58.8 MSPS for the scaling schemes 1 and 3 is caused by the fact that the 10 bit ε-CRT must be placed in different rows (each row has only one EAB). This introduces no *system* speed decrease because the scaling is applied at the lower (output) sampling rate. For the BRS-ε-CRT it is assumed that only the BRS m_4 part (see Fig. 4) must run at the input sampling rate, while BRS m_3 and ε-CRT run at the output sampling rate.

Some resources can be saved if the detailed scaling scheme from Hogenauer [5] shown in Fig. 1(b) is applied. With this scheme the BRS-ε-CRT scheme must be applied to reduce the bit-width in earlier sections of the filter. The early use of ROMs decreases the possible throughput from 76.3 to 70.4 MSPS which is the maximum speed of the BRS with m_4. At the output the efficient ε-CRT scheme was applied.

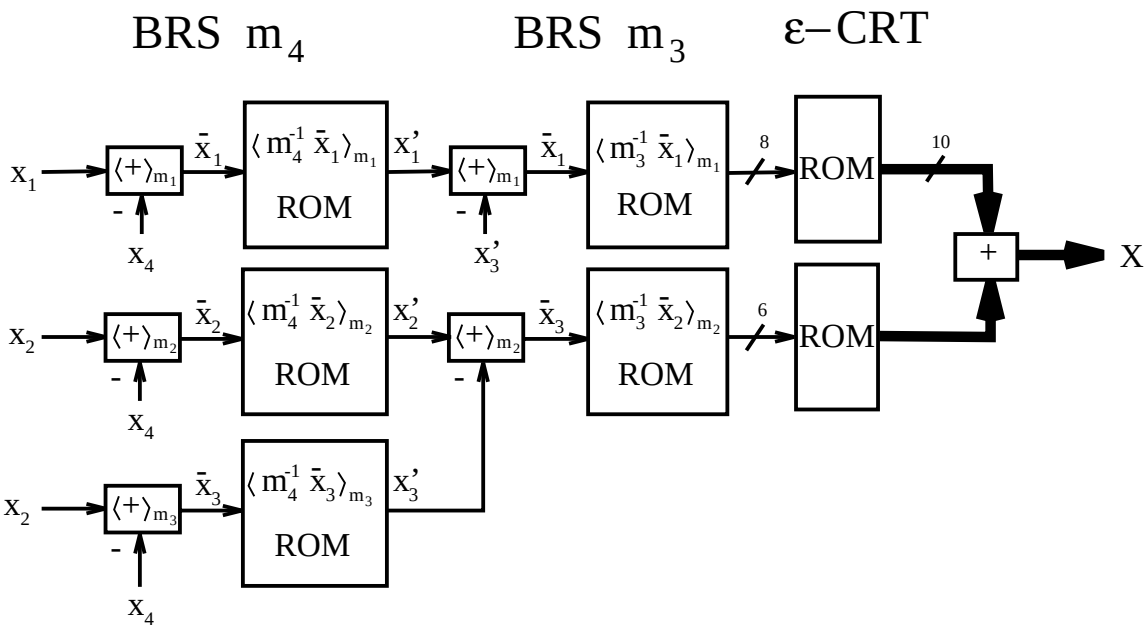

Figure 4: BRS and ε-CRT conversion steps.

The following table concludes the three implemented filter realization without including the scaling data.

Type	TC 26 Bit	RNS 8, 6, 6, 6 bit	detailed bit-width RNS design
MSPS	49.3	76.3	70.4
#LE	343	559	355

6. CONCLUSION

The Hogenauer [5] cascade integrator comb filters has been implemented with CPLDs. Compared to a two's complement design, the presented RNS-based design enjoys an improved speed advantage by approximately 54%. A detail BRS-ε-CRT scaling scheme was developed for the CIC-RNS, which runs 43% faster than the two's complement design and reduced the effort by 36% compared with a scaling scheme at the output.

7. REFERENCES

[1] Altera Corporation, "FLEX 10K Data Sheet". June 1996.

[2] M. Bayoumi, G. Jullien, and W. Miller, "A VLSI Implementation of Residue Adders". *IEEE Transactions Circuits and Systems*, 34(3):284-288, March 1987.

[3] M. Griffin, M. Sousa, and F. Taylor, "Efficient Scaling in the Residue Number System". *IEEE International Conference on Acoustics, Speech, and Signal Processing*, pages 1075 -1078, 1989.

[4] Harris Semiconductor, "Data Sheet HSP43220 Decimating Digital Filter". Aug. 1992.

[5] E. B. Hogenauer, "An Economical Class of Digital Filters for Decimation and Interpolation". *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 29(2):155-162, April 1981.

[6] G.A. Jullien, "Residue Number Scaling and Other Operations Using ROM Arrays". *IEEE Transactions on Computers*, 27:325-336, April 1978.

[7] J. Keller, "Xilinxs past bounds of adaptive chips". *Military & Aerospace Electronics*, April 1997.

[8] I. Koren. *Computer Arithmetic Algorithms*. Prentice Hall, 1993.

[9] U. Meyer-Baese, J. Mellott, and F. Taylor, "Design of RNS Frequency Sampling Filter Banks". *IEEE International Conference on Acoustics, Speech, and Signal Processing*, 3:2061-4, April 1997.

[10] J. Mellott. *Long Instruction Word Computer*. Ph.D. thesis, University of Florida, Gainesville, 1997.

[11] N. Szabo and R. Tanaka, *Residue Arithmetic and its Applications to Computer Technology*. McGraw-Hill, 1967.

[12] M. Soderstrand, W. Jenkins, G. Jullien, and F. Taylor, *Residue Number System Arithmetic: Modern Applications in Digital Signal Processing*. IEEE Press Reprint Series. IEEE Press, 1986.

[13] V. Hamann and M. Sprachmann "Fast Residual Arithmetics with FPGAs". *Proceedings of the Workshop on Design Methodologies for Microelectronics*, Smolenice Castle, Slovakia, pages 253 - 255, Sept. 1995.

[14] R. Hartenstein, J. Becker and R. Kress "Costum Computing Machines vs. Hardware/Software Co-Design: From a Globalized Point of View". *Lecture Notes in Computer Science*, 1142:65-76, Springer Berlin, Sept. 1996.