J Sign Process Syst (2012) 67:229–237
DOI 10.1007/s11265-010-0549-7

A Reconfigurable Channel Filter for Software Defined Radio Using RNS

K. G. Smitha · A. P. Vinod

Received: 11 March 2010 / Revised: 15 October 2010 / Accepted: 18 October 2010 / Published online: 18 November 2010

Abstract This paper presents a high-speed FIR channel filter using residue number system (RNS) whose frequency response can be reconfigured to adapt to a multitude of channel filtering specifications of a multi-standard software defined radio (SDR) receiver. The channel filters in the channelizer of an SDR extract multiple narrowband channels corresponding to different communication standards from the wideband input signal. The proposed architecture has been synthesized on TSMC 0.18 μm CMOS standard cell technology. Synthesis result shows that the proposed reconfigurable FIR channel filter, for a Digital Advanced Mobile Phone Systems (D-AMPS) example, offers speed improvement of 42% and AT complexity reduction of 26% over existing reconfigurable FIR method.

Keywords Multi-standard wireless communications · Residue number system · Channel filter · Software defined radio · Reconfigurability

1 Introduction

Software defined radio (SDR) is emerging as a powerful platform for future generation cellular systems due to its capability of operating according to multiple mobile radio standards. The adaptability of SDR to different communication standards is guaranteed by the reconfigurability of the channel-of-interest selection unit called the channelizer. The channelizer extracts multiple narrowband channels from a wideband signal using a bank of finite impulse response (FIR) filters, called channel filters. Apart from reconfigurability, high speed and low power consumption are two key requirements in channel filters as they need to operate at the highest sampling frequency in the digital front-end.

Several implementation approaches for reconfigurable FIR filters have been proposed in literature [1]. The performance of programmable multiply-accumulate (MAC) based filter [1] is mainly restricted by the increased delay and power consumption of the coefficient multiplier. In [2], a digit reconfigurable FIR filter architecture was proposed where the objective was to reduce the precision of coefficients and thus the filter complexity, without affecting the filter performance. But the architecture demanded huge hardware resources. It performs multiplication using direct shift and add method and hence will result in low speed operation, which makes the method infeasible for SDRs. A high-speed, canonic signed digit based (CSD) based reconfigurable FIR filter is presented in [3], which resulted in high speed operation, but at the cost of large area and high power consumption. The aim in [3] was solely to design a high-speed reconfigurable filter and no consideration was given for area and power reduction. In [4], the concept of reconfigurable multiplier block (REMB) was introduced, which utilized graph dependence (GD) algorithms. As GD algorithms are sequential in nature, the architecture resulted in low speed operation. A multiplexed multiple constant multiplication (MMCM) approach is proposed in [5], which takes coefficients as constants and used GD algorithms to reduce redundancy. But this

K. G. Smitha (✉) · A. P. Vinod
School of Computer Engineering,
Nanyang Technological University,
Nanyang Avenue,
Singapore 639798, Singapore
e-mail: smitha@ntu.edu.sg

A. P. Vinod
e-mail: asvinod@ntu.edu.sg

architecture will also result in longer critical path lengths and consequently will reduce the speed of operation. A reconfigurable FIR filter based on binary representation of common subexpression elimination (CSE) technique is described in [6]. The method in [6] incorporated binary based CSE optimization in the multiplier and hence the method effectively reduces the area of the filter.

Filter realizations using alternative number systems such as Residue number system (RNS) have been investigated in literature. RNS is suitable for implementation of high-speed digital signal processing due to their inherent parallelism, modularity, fault toleration and local carry propagation properties [7, 8]. Arithmetic operations like multiplication and addition can be carried out more efficiently as RNS ensures localized carry propagation properties. RNS is particularly suitable for implementing FIR filters as the core operations in FIR filtering are multiplications and additions.

A fast RNS based communication receiver design and implementation using Field programmable gate arrays (FPGAs) is presented in [9]. RNS arithmetic operations are implemented in [9] as a set of concurrent primitive operations with non-communicating small wordlength channels. A hybrid RNS adaptive filter was introduced in [10] where the adaptive filter implementation is done partly in RNS and partly in two's complement. This enables the architecture to have the advantage of area and speed of the hybrid RNS adaptive filter with respect to the two's complement one. Both [9, 10] discussed dedicated RNS filter architectures while [10] deals with the hybrid architecture using RNS. Reconfigurable computing in the RNS domain is a rather new field. Most research in this field has been focused on variable word length (VWL) RNS processors [11], which provide reconfigurability in terms of dynamic range by providing the ability to turn off the channels that are not required. This kind of reconfigurability can offer variable precision realization of FIR filters. A multi-modulus modulo multiplier based on sharing of common resources for the three moduli set $\{2^n-1,\ 2^n,\ 2^n+1\}$ has been investigated in [12]. Although reconfigurability has been addressed in RNS literature from the perspective of realizing filters with variable coefficient wordlengths and multi-modulus modulo multipliers, hardly any RNS-based work addressed the problem of designing filter architectures for different frequency response specifications as required in the case of multi-standard wireless communication receivers. In such receivers, very-high-order FIR channel filters are required to meet the stringent adjacent channel attenuation specifications of wireless communication standards. It has been shown in [13] that such higher-order FIR filters can be implemented with low hardware complexity in RNS domain. However, reconfigurability of filters for multi-standard operation was not addressed in [13].

In this paper, we propose an FIR filter implementation using RNS, that can be reconfigured to satisfy the filter frequency response specification of a new communication standard when the receiver switches its operation from its current standard to a new standard, in a multi-standard wireless communications scenario. Existing RNS based FIR filter implementations focus on achieving different dynamic range, and there is hardly any work addressing reconfiguration for a different frequency response specification. The preliminary idea was presented in a conference paper [14]. In our current work, we have optimized the encoder hardware proposed in [14] by exploiting the commutative multiplication property between quantized input signal and filter coefficients. Also, the synthesis results of a 350-tap Digital Advanced Mobile Phone Systems (D-AMPS) channel filter as well as comparison of proposed product encoder with several other methods are obtained. Section 2 provides a review on RNS arithmetic for FIR filter implementation. Section 3 presents the proposed reconfigurable RNS FIR filter. Design example and implementation are shown in Section 4 followed by conclusions in Section 5.

2 Background on RNS FIR Filters

RNS is defined by a moduli set, which consists of p pair wise relatively prime integers $\{m_0,\ m_1,\ .\ .\ .\ ,m_{p-1}\}$. The useful computational range M of such a number system, which is called the legitimate range, is defined by the product of all moduli in the moduli set, i.e. $M = \prod_{i=0}^{p-1} m_i$. A residue number system with valid range M is able to uniquely represent unsigned numbers in the range of $[0,\ M-1]$, or signed numbers in the range of $[-(M-1)/2,\ ((M+1)/2)-1]$ for odd M, and $[-M/2,\ (M/2)-1]$ if M is even. These ranges are known as the dynamic ranges of RNS [7, 8].

Any integer X within the dynamic range has a unique RNS representation that can be characterized by the list of its residues with respect to the moduli defined in the moduli set.

$X \xrightarrow{RNS} \left(\langle X\rangle m_1, \langle X\rangle m_2, ..., \langle X\rangle m_{p-1}\right)$ where $\langle X\rangle m_i$, denotes the operation [7, 8]:

$$\langle X\rangle\, m_i = \begin{cases} X \bmod m_i \text{ for X} \geq 0 \\ (M - X) \bmod m_i \text{ for X} < 0 \end{cases}$$

Addition, subtraction and multiplication can be performed in RNS domain on the residue representation of the operands. All the above operations, on different m_i (moduli) are done in parallel. Consequently, operations on large wordlengths can be split into several modular operations executed in parallel with shorter wordlengths. Since two's complement is the most common number representation, conversions between residue and binary representations are required. Mapping from the RNS back to the binary representation is defined by the Chinese Remainder Theorem (CRT) [8].

A Finite Impulse Response (FIR) filter of order N is described by the expression

$$y(n) = \sum_{k=0}^{N-1} H_k x(n-k) \tag{1}$$

The implementation of RNS based FIR filter can be shown by (2) [8].

$$\begin{cases} y_{m_1}(n) = \left\langle \sum_{k=1}^{N} \langle H_{m_1}(k) \cdot X_{m_1}(n-k) \rangle_{m_1} \right\rangle_{m_1} \\ y_{m_2}(n) = \left\langle \sum_{k=1}^{N} \langle H_{m_2}(k) \cdot X_{m_2}(n-k) \rangle_{m_2} \right\rangle_{m_2} \\ \ldots\ldots\ldots \\ y_{m_{p-1}}(n) = \left\langle \sum_{k=1}^{N} \langle H_{m_{p-1}}(k) \cdot X_{m_{p-1}}(n-k) \rangle_{m_{p-1}} \right\rangle_{m_{p-1}} \end{cases} \tag{2}$$

FIR filtering in RNS domain is done using multiple modulo m_i FIR filter blocks working in parallel, as shown in Fig. 1. Each of them is implemented with additions and multiplications *mod* m_i as shown in Fig. 2 where '*D*' represents inter-tap delay. A drawback of RNS is the need for converters to/from the conventional number system (two's complement or sign-and magnitude) termed forward/reverse converters respectively. However, this conversion overhead has only a small impact for high-order filters such as the ones needed for channel filtering in SDR receivers [13].

3 Proposed Reconfigurable RNS-FIR Filter

In conventional multi-standard transceivers, a separate filter is needed for each standard, and reconfigurability is achieved by switching among distinct filters according to the current mode of operation. This is not an efficient approach due to its increased hardware complexity and poor resource utilization. In this section, we present a reconfigurable filter architecture using RNS technique.

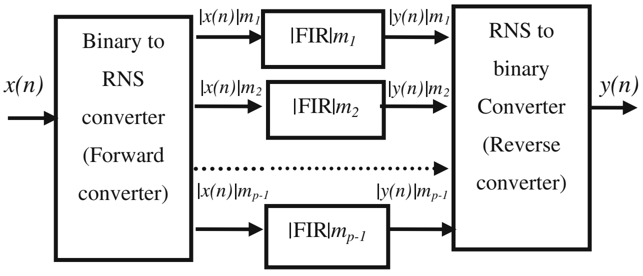
x(n)
Binary to RNS converter (Forward converter)
|x(n)|m1
|x(n)|m2
|x(n)|mp-1
|FIR|m1
|FIR|m2
|FIR|mp-1
|y(n)|m1
|y(n)|m2
|y(n)|mp-1
RNS to binary Converter (Reverse converter)
y(n)

Figure 1 FIR filter implementation using RNS.

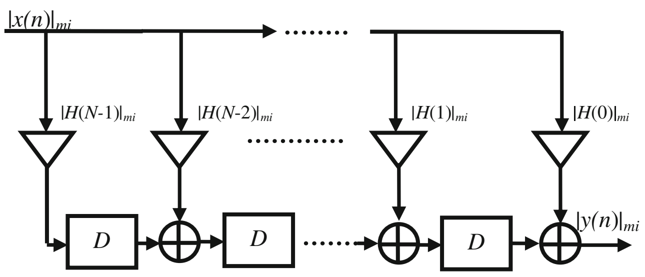

Figure 2 $|\mathrm{FIR}|_{mi}$ stucture.

3.1 Proposed Channel Filter Architecture

The coefficient values of FIR filters, catering for multiple standards are generally stored in look up tables (LUTs). In conventional LUT based filter implementation [4], the filter coefficients are stored as such with its full precision. The multiplication is done as a one-to-one mapping of ($k_1 \times k_2$) bits, where k_1 and k_2 are the possible combinations of input and coefficients respectively. This approach will consume huge search space in LUT and thus the area also increases. In the proposed method the coefficient values of filters for multiple standards are stored in a LUT with respect to their corresponding residue value. Hence the coefficient value to be stored in the LUT will enjoy less area as the wordlength is limited by the modulo value. Our proposed reconfigurable filter realization is explained using the following example. The moduli values are fixed for a particular format, $\{2^n-1, 2^n, 2^n+1\}$ as {31, 32, 33}, for illustration purposes. The precision or wordlength of the filter is defined as the number of bits used to represent the filter coefficients. The wordlength is fixed for fixed moduli. Our proposed method aims to reconfigure the filter coefficients on the fly, corresponding to multiple standards, for a given word length. It can be noted that for moduli based number systems such as RNS, the maximum possible value of the residue of any number in the modulus $\{m_1, m_2, m_3\}$= {31,32,33} is ($m_{3-}1$)=32. Thus the residue result of any computation in the modulus m_1 can only have values from 0 to m_1−1. We propose to exploit this reduced range of residue values for hardware optimization when compared to the original range as taken by other LUT based algorithms [4]. Let the maximum modulus in a given moduli set is $m_{max.}$ The quantized input and the filter coefficient are converted to the corresponding moduli and let $m_{max-}1$ be the maximum value for both, for the given moduli. Then, if we compute all the bit combinations of (m_{max}−1) x (m_{max}−1), the net result can only have a maximum value of ($m_{max-}1$). Hence the output value of the modulo multiplier can vary only between 0 to ($m_{max-}1$). The interesting property is that for a given moduli, the input as well as any set of filter coefficient can vary only between 0 to ($m_{max-}1$), and hence the modulo multiplier needs to be implemented only once and it can be reused for any set of input and coefficients. Using this property we can redesign the multiplier as a product encoder which selects the pre-computed multiplied result as output corresponding to the filter input and filter coefficient. The search space of the encoder is limited as the output value is again limited by the modulo value. There are redundant cases of the multiplications between filter input and coefficient when they share a commutative property (filter input x filter coefficient =filter coefficient x filter input). The cases of a commutative multiplication property between input and coefficients are identified and eliminated in the proposed approach when compared to [14], to improve the area efficiency of the product encoder. The unique product encoder proposed in this work has low area-time complexity compared to conventional filter coefficient multipliers, which are designed for a generalized format for variable wordlengths. In addition to reducing the area of the filter circuit, we propose a technique to integrate reconfigurability in FIR filter architecture.

The proposed scheme shown in Fig. 3 can be explained as follows. The precision or wordlength of input and filter coefficient for the moduli set of {31, 32, 33} is 8. If the wordlength of the quantized input signal is less than 8 bits then a forward converter to convert the binary input number to RNS is not required. If quantized input has more than 8 bits, it is converted to respective residue value using a forward converter. The residue values for each coefficient set corresponding to a distinct communication standard can be calculated offline and stored in LUT and hence we don't need a forward converter for filter coefficients. Thus, for a *k*-standard wireless communication receiver, the LUT will have modulo values of all the coefficients of *k* filters. The product encoder for the particular moduli selects the pre-computed product of the input signal with the corresponding filter coefficient stored in LUT. The results are added using modulo add operation for the corresponding moduli. The output of each FIR filters corresponding to each moduli, $|y(n)|_{m_i}$, is reverse converted

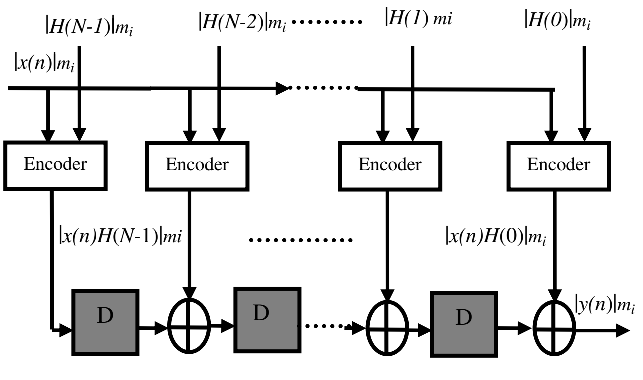

Figure 3 Reconfigurable RNS-FIR filter for *mod* m_i.

to binary to obtain the filter output, *y (n)*. Note that forward and reverse converters are not shown in Fig. 3. The proposed RNS-FIR reconfigurable filter implementation is shown in Fig. 4.

The moduli set is made fixed for a multi-standard receiver taking into account of the maximum dynamic range corresponding to the wireless standard that has the most stringent specification among the set of wireless standards. There is hardly any need of changing the dynamic range for the wireless standards for which the receiver is designed as the product encoder is designed for the most stringent dynamic range. When the mode of communication switches to a new standard, the coefficient set corresponding to the filter for new standard will be loaded in the LUT. In this case also, the same pre-designed encoder performs product selection. The modulo adder and the reverse converter are also designed corresponding to the pre-fixed moduli set. Thus the proposed high-speed architecture can work as a reconfigurable FIR filter, for a fixed moduli value.

The procedure to implement the proposed RNS reconfigurable filter is given below in a step by step format.

1. Fix the moduli set which fixes the dynamic range of the multiplier.
2. For a given set of coefficients we can calculate their value in RNS domain for the corresponding moduli. When there is a need to change the set of coefficients, we need to only furnish the new set with corresponding moduli (calculated offline) to the proposed product encoder.
3. The input is passed through a forward converter if the wordlength is more that the dynamic range set by the moduli. The forward converter will convert the binary input to the residue domain for the corresponding moduli.
4. The LUT based product encoder for a particular moduli is designed in such a way that the product in RNS domain is selected when the inputs (both input and coefficient in the RNS domain for a particular moduli) are provided.
5. The same product encoder can be used when the coefficient and the input changes for the given moduli thus making it reconfigurable.

An example is given below.

1. Moduli is fixed as {7, 8, 9} which fixes the dynamic range of the multiplier as 9. We can take the input wordlength as 4 bit and the coefficient wordlength as 5 bit.
2. Let first set of coefficients be {102, 34, 52, 78} in the decimal format and input to be multiplied be {98}. Converting them to the RNS domain for the moduli {7,8,9}, the coefficients set becomes

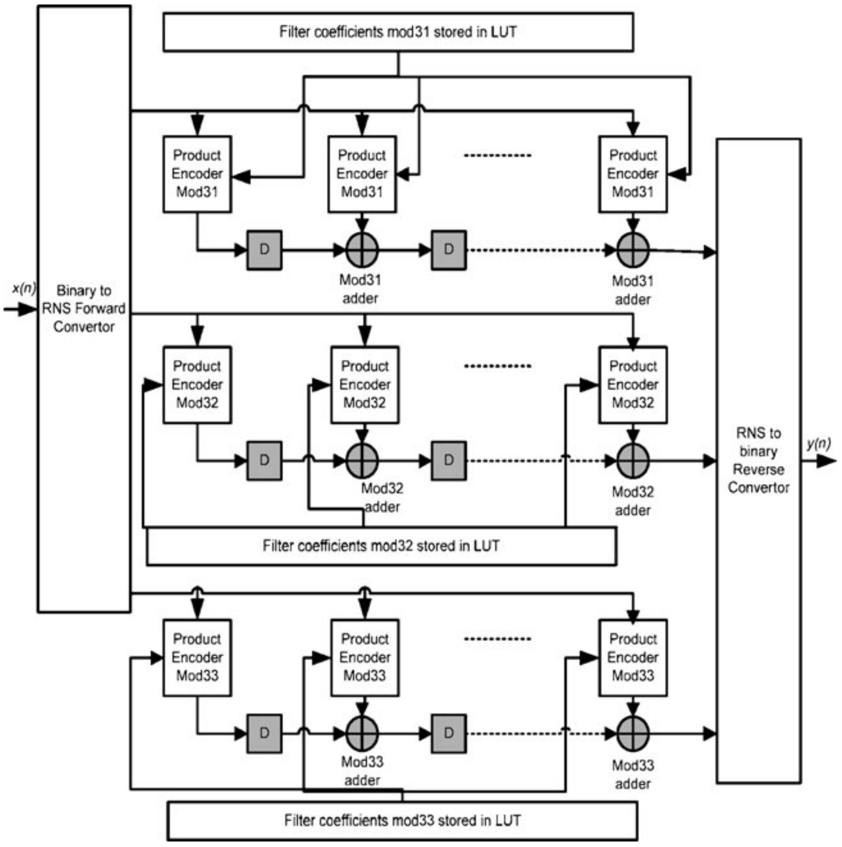

Figure 4 High–speed and reconfigurable RNS-FIR filter.

$\{[102]_{\{7,8,9\}},[34]_{\{7,8,9\}},[52]_{\{7,8,9\}},[78]_{\{7,8,9\}}\}=\{[4, 6, 3], [6, 2, 7],[3, 4, 7],[1, 6, 6]\}$, the input set in RNS domain is $\{[98]_{\{7,8,9\}}\}=\{[0,2,8]\}$.

3. The LUT implementation of product encoder for each moduli will multiply the corresponding input and coefficients in the RNS domain. Hence for this example the product encoder for

   moduli{7} = coefficient set {4,6,3,1} * input set {0} as $\{0,0,0,0\}_7=\{0,0,0,0\}$.
   moduli{8} = coefficient set {6,2,4,6} * input set {2} as $\{12,4,8,12\}_8=\{4,4,0,4\}$.
   moduli{9} = coefficient set {3,7,7,6} * input set {8} as $\{24,56,56,48\}_9=\{6,2,2,3\}$.

4. The product results are inputted to the reverse convertor. The reverse convertor maps the output corresponding to the input sets as {9996,3332,5096,7644}
   i.e output y(n)=9996(n−3)+3332(n−2)+5092(n−1)+7644.

A change in input or coefficient in the corresponding moduli will definitely lie within the same range of input of the product encoder. Hence we can use the same product encoder for various coefficient sets (corresponding to variable frequency responses) and input sets, once the dynamic range of the multiplier is fixed.

3.2 Product Encoder Implementation Results

The product encoder has been implemented in TSMC 0.18 µm CMOS standard-cell technology as a proof-of-concept. The architectures were coded in Verilog and functionally is simulated using MODELSIM SE. Synopsys Design Compiler (v2001.08) was used for the synthesis. The synthesis results and their comparison with RNS multiplier architectures available in literature [12], [15–19] are shown in Table 1. The area-time metrics of proposed 2^n+1 encoder for different bit-width (n=4, 8, 16) and that of the multipliers in [12, 15–19] are summarized in Table 1.

For 4 bit × 4 bit multiplication, both area and time metrics of our proposed product encoder outperforms the multiplier architectures in [16, 18, 19]. The proposed architecture is having a greater delay when compared to [17], but in terms of Area x time (AT) complexity our method outperforms [16] by 50%. For 8 bit × 8 bit multiplication, our method offers lower delay and area when compared to the multipliers in [12, 15, 18, 19]. Our method is having a higher delay when compared to [16, 17], but taking the AT complexity, our architecture outperforms [16]. The AT complexity is least for our method in 16 bit × 16 bit multiplication. On an average, the AT complexity results shows that, our 2^n+1 product encoder architecture has 67% improvement over [12], 62.2% over [15] 23.43% improvement over [16], 35.8% improvement

Table 1 Synthesis result of product encoder.

	Area (μm^2)	Time delay (in ns)	Area-Time
(4bit × 4 bit)			
Proposed 2^n+1 encoder	1350.50	1.27	1715.1
Efstathiou. et. al [16]	3724.00	1.34	4990.2
Vergos & Efstathiou [17]	3206.00	1.09	3494.5
Wang et. al [18]	3939.00	1.51	5947.9
Ma [19]	3134.00	1.64	5139.8
(8bit × 8 bit)			
Proposed 2^n+1 encoder	9225.00	2.07	19095.8
Zimmermann. [15]	20232.00	2.47	49973.0
Efstathiou et al. [16]	9685.00	1.98	19176.3
Vergos & Efstathiou [17]	8819.00	1.96	17285.2
Menon & Chang [12]	21231.75	3.05	64756.8
Wang et al. [18]	10427.00	2.19	22835.1
Ma [19]	8830.00	2.32	20485.6
(16bit × 16 bit)			
Proposed 2^n+1 encoder	32485.00	2.82	91607.7
Zimmermann. [15]	66580.00	3.68	245014.4
Efstathiou. et. al [16]	36541.00	2.65	96833.7
Vergos & Efstathiou [17]	34252.00	2.61	89397.7
Menon & Chang [12]	67236.00	3.69	248100.8
Wang et. al [18]	38619.00	2.94	113539.9
Ma [19]	29856.00	3.09	92255.0

over [18] and 24.4% improvement over [19]. The proposed architecture results in slightly inferior AT complexity result, when compared to [17], for the 8 bit and 16 bit multiplication. This is basically because the method in [17] is a dedicated multiplier optimized for area and delay and it is not reconfigurable whereas the proposed architecture is reconfigurable. In the proposed architecture, area and delay reductions are made possible due to the reduction in the output search space of the product encoder and the elimination of redundant multiplications by fixing the moduli to the most stringent dynamic range.

4 Design Example and Synthesis Results

In this section, we present the synthesis results of the FIR channel filter employed in the channelizer of a wireless communication receiver. The Digital Advanced Mobile Phone Systems (D-AMPS) receiver in [20] is considered to implement our RNS based reconfigurable channel filter. The sampling rate chosen is 34.02 MHz as in [20]. The channel filters extract 30 kHz D-AMPS channels from the input signal after down sampling by a factor of 350. The passband and stopband edges are 30 kHz and 30.5 kHz respectively. The peak passband ripple (PPR) and peak stopband ripple (PSR) specifications are 0.1 dB and−24 dB respectively. The length of the FIR filter, N, is determined using (3) [21]

$$N = \frac{-10\log_{10}(\delta_p.\delta_s) - 13}{14.6(f_s - f_p)} \tag{3}$$

where δ_p is the PPR, δ_s is the PSR and $(f_s - f_p)$ is the transition band width normalized by the sampling frequency. We have chosen a PSR of −24 dB and the transition to be 0.01 so that the optimum filter length is found to be 350 according to (3).

Table 2 0.18 μm Syntheis result of 350 tap D-AMPS channel filter.

	Proposed RNS-FIR Filter	Reconfigurable FIR filter BPSM [6]	Reconfigurable FIR filter BCSM [6]	FIR filter [2]	FIR filter [4]
Filter length	350	350	350	350	350
Area (mm^2)	5.86	4.531	4.82	14.08	12.37
Delay (ns)	5.53	9.87	9.08	17.45	19.65
AT complexity	32.4	44.7	43.76	245.7	243

Table 3 0.18 μm Syntheis result in comparison with reconfigurable fir filters [2, 4].

	RNS-FIR	FIR filter [2]	ReMB filter [4]
Coefficient wordlength (bits)	8	8	8
Filter length	32	32	32
Area (mm^2)	1.41	1.47	1.394
Delay (ns)	2.55	5.97	7.17
AT Complexity (mm^2x ns)	3.6	8.77	9.99
Power (mW)	2.9	8.5	6.5

It must be noted that the proposed architectures do not alter the coefficient values and therefore the frequency response of the filter implemented using our architectures is unaffected.

We have implemented the D-AMPS channel filter for a coefficient wordlength of 16 bits. The product encoders corresponding to each moduli is designed only once and it is reused for the realization of reconfigurable channel filter architecture. The channel filter is synthesized on TSMC 0.18 μm CMOS standard cell technology. The synthesis results are shown in Table 2 along with a comparison of our method with the most recent reconfigurable FIR filter proposed in [6]. Two methods of binary constant shifts method (BCSM) and binary programmable shifts method (BPSM) are proposed in [6]. The reconfigurable FIR method in [6] uses binary based CSE technique to optimize the multipliers by using addition and shifts. Thus it results in reduced area implementation when compared to the proposed reconfigurable RNS-FIR filter.

As in any RNS filter implementations, a reverse converter is needed to convert the RNS representation back to binary representation [8]. It can be noted that the forward and reverse converters can have fixed moduli, as the moduli is selected to satisfy the most stringent communication standard. Here we have implemented an LUT based reverse converter for the moduli selected according to the most stringent standard among the set of standards considered. The LUT based reverse convertor maps the input sequences within the fixed moduli to the corresponding output. The LUT mapping is a one-to-one mapping and hence this will increase the speed of reverse converter. The area overhead of the reverse converter is 1.14 mm^2 and it is included in the results given in Table 2. The forward converter can be omitted for cases when the input wordlength is less than the designed wordlength. The area overheads of forward and reverse converters are also included in the results. It can be noted that the proposed method offers a speed improvement of 42% over [6], 68% over [2] and 72% over [4]. The proposed method also offers an AT complexity improvement of 26% over [6], 86.8% over [2] and 86% over [4]. The higher-order examples of [2] and [4] are taken from [6].

RNS based FIR filters found in literature are not reconfigurable to achieve different frequency response specifications as required in the case of multi-standard communication receivers. Hence we have compared our technique with general reconfigurable filters available in literature [2–5]. We have given a comparison of the proposed method with [2–5] in Tables 3 and 4. The synthesis results of the digit reconfigurable method [2] and the high speed reconfigurable CSD method [3] are taken from [6]. From Table 3, the proposed method has the least AT complexity and power when compared with the 32 tap implementations in [2] and [4]. Our method offers a speed improvement of 57.28% and 64.43% over the architecture in [2] and [4] respectively at the cost of an area increase by 1.13%. The AT complexity of our architecture is 58.95 % less than that of [2] and 63.96 % less than [4]. The proposed method offers a power savings of 65.8% over [2] and 55.4% over [4]. From Table 4, the MMCM method in [5] is having an area reduction of 64.5% when compared to the proposed method, as the multiplication is done serially. The delay for MMCM method is

Table 4 0.18 μm Syntheis result in comparison with reconfigurable fir filters [3, 5].

	RNS-FIR	CSD-FIR [3]	RNS-FIR	MMCM—filter [5]
Coefficient wordlength (bits)	10	10	16	16
Filter length	18	18	20	20
Area (mm^2)	1.32	13.872	1.54	0.5467
Delay (ns)	2.12	7	2.39	15.6
AT Complexity (mm^2x ns)	2.8	97.1	3.68	8.53
Power (mW)	2.4	–	3.02	16

84.65% higher and hence the AT complexity of MMCM method is 56.6% higher than the proposed method. The power dissipation of the CSD based reconfigurable FIR in [3] is not reported in [3]. The proposed reconfigurable RNS-FIR method outperforms the high speed CSD method in [3] in area and delay by 90.5% and, 69.7% respectively. Our method has a huge power savings of 81.1% over [5].

5 Conclusions

A high-speed reconfigurable channel filter based on RNS is presented in this paper. The results show that our proposed product encoder offers AT improvements of 67% improvement over [12], 62.2% over [15], 23.43% over [16], 35.8% over [18] and 24.4% over [19]. The synthesis results using TSMC 0.18 μm CMOS technology for D-AMPS channel filter shows a speed improvement of 42% and AT complexity improvement of 26% over [6]. The AT complexity of our architecture is 58.95% less than that of [2] and 63.96% less than [4]. The proposed method offers a power savings of 65.8 % over [2] and 55.4% over [4].

References

1. Solla, T., & Vainio, O. (2002). Comparison of programmable FIR filter architectures for low power. In *Proc. of 28th European Solid-State Circuits Conference* (pp. 759–762), Sep. Firenze, Italy.
2. Chen, K. H., & Chiueh, T. D. (2006). A low-power digit-based reconfigurable FIR filter. *IEEE Transactions on Circuits and Systems II, 53*(8), 617–621.
3. Zhangwen, T., Zhang, J., & Min, H. (2002). A high-speed, programmable, CSD coefficient FIR filter. *IEEE Transactions on Consumer Electronics, 48*, 834–837.
4. Demirsoy, S. S., Kale, I., Dempster, A. G. (2004). Efficient implementation of digital filters using novel reconfigurable multiplier blocks. In *Proceedings of Thirty-Eighth Asilomar Conference on Signals, Systems and Computers* (vol. 1, pp. 461–464).
5. Tummeltshammer, P., Hoe, J. C., & Puschel, M. (2007). Time-multiplexed multiple-constant multiplication. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 26* (9), 1551–1563.
6. Mahesh, R., & Vinod, A. P. (2010). New reconfigurable architectures for implementing FIR filters with low complexity, *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 29*(2), 275–288.
7. Szabo, N. S., & Tanaka, R. I. (1967). *Residue arithmetic and its applications in computer technology.* New York: McGraw-Hill.
8. Amondi, A., & Premkumar, B. (2007). Residue number systems-Theory and implementation, vol. 2. Imperial College Press.
9. Base, U. M., Garcia, A., & Taylor, F. (2001). Implementation of a communications channelizer using FPGAs and RNS arithmetic. *Journal of VLSI Signal Processing Systems, 28*(1–2), 115–128.
10. Bernocchi, G. L., Cardarilli, G. C., Del Re, A., & Nannarelli, A. (2006) A hybrid RNS adaptive filter for channel equalization. *Fortieth Asilomar Conference on Signals, Systems and Computers,* pp. 1706–1710, Oct.–Nov.
11. Cadarilli, G. C., et al. (2002). Residue number system for reconfigurable datapath. In *Proc. IEEE int. Symp. On Circuits and Syst*., *2*, 756–759.
12. Menon, S., & Chang, C. H. (2006). A reconfigurable multi-modulus modulo multiplier. In *Proc. IEEE Asia pacific conference on Circuits and Systems* (pp. 1168–1171). Singapore, Dec.
13. Soderstrand, M. A., & Al-Marayati, K. (1995). VLSI implementation of very-high-order FIR filters. *IEEE International Symposium on Circuits and Systems, 2*(30), 1436–1439.
14. Smitha, K. G., & Vinod, A. P. (2008). A reconfigurable high-speed RNS-FIR channel Filter for multi-standard software radio receivers. *Proceedings of IEEE International Conference on Communication Systems,* ICCS 2008, pp. 1354–1358, Nov.
15. Zimmerman, R. (1999). Efficient VLSI implementation of modulo (2^n+1) addition and multiplication. In *Proc. 14th IEEE Symp. Computer Arithmetic* (pp. 158–167). Apr.
16. Efstathiou, C., Vergos, H. T., Dimitrakopoulos, G., & Nikolos, D. (2005). Efficient diminished-1 modulo 2^n + 1 multipliers. *IEEE Transactions on Computers, 54*(4), 491–496.
17. Vergos, H. T., & Efstathiou, C. (2007). Design of efficient modulo 2/sup n/ + 1 multipliers. *IET Computers and Digital Techniques, 1* (1), 49–57.
18. Wang, Z., Jullien, G. A., & Miller, W. C. (1996). An efficient tree architecture for modulo 2^n+ 1 multiplication. *Journal of VLSI Signal Processing, 14*, 241–248.
19. Ma, Y. (1998). A simplified architecture for modulo (2^n+1) multiplication. *IEEE Transactions on Computers, 47*(3), 333–337.
20. Vinod, A. P., & M-K Lai, E. (2005). An efficient coefficient-partitioning algorithm for realizing low complexity digital filters. *IEEE Transactions on Circuits and Systems II, 24*, 1936–1946.
21. Proakis, J. G., & Manolakis, D. G. (1998). Design of digital filters. In *Digital Signal Processing Principles, algorithms, and applications* (pp. 614–738). Upper Saddle River: Prentice-Hall.

K. G. Smitha received her B Tech degree in electrical and electronics engineering from Calicut University, India in 2002, the M.E degree from Anna university, India in 2004 and PhD degree from Nanyang Technological University in 2010. She was a lecturer in Amrita viswavidaypeetham, Coimbatore, India from May 2004 to November 2004. Currently she is pursuing post doctoral research in School of Computer Engineering, Nanyang Technological University, Singapore. Her main research interests are in the areas of low complexity and high speed digital signal processing circuits, computer arithmetic and cognitive radio.

A. P. Vinod (M'01-SM'07) received his B Tech degree in instrumentation and control engineering from University of Calicut, India in 1994 and the M. Engg and PhD degrees in computer engineering from Nanyang Technological University, Singapore in 2000 and 2004 respectively. Since October 1993, he worked as a project engineer for 5 years in Kirloskar, India, Tata Honeywell, India, and Shell, Singapore. From September 2000 to September 2002, he was a lecturer in the School of Electrical and Electronic Engineering at Singapore Polytechnic, Singapore. He joined the School of Computer Engineering at Nanyang Technological University, Singapore, as a faculty in September 2002 where he is currently as Associate Professor. His research interests include digital signal processing (DSP), low complexity DSP circuits for software radio and cognitive radio, number theoretic transforms and brain computer interface.