Reducing complexity of fixed-coefficient FIR filters

R. Conway

An approach for reducing the complexity of the multiplier block for fixed coefficient FIR filters using the transpose structure is presented. Through the use of modular arithmetic with moduli of the type $2^n \pm 1$ and 2^n, modulo multiplier blocks with a small full adder count are obtained using a minimum spanning tree algorithm. The results obtained show a very significant reduction in the filter complexity relative to existing techniques. A unique feature of the results is that the complexity is independent of filter length, but a function of the filter's dynamic range.

Introduction: Fixed coefficient FIR filters are commonly encountered in DSP and communication applications. Using the well-known canonic signed digit (CSD) representation yields the minimum number of nonzero digits for each coefficient. This in turn facilitates a reduction in the number of adders needed to implement the FIR filter in dedicated hardware. The transpose form structure for an FIR filter lends itself to another technique known as common subexpression elimination, which is used to exploit redundancy in multiplying the same filter input value by different coefficients. By sharing adders across the different constant multipliers, a significant reduction in adder count is achieved [1–3]. The residue number system (RNS) is a non-weighted numbering system that uses remainders to represent numbers. An RNS FIR filter operates by performing filtering in parallel channels with each channel operating modulo m_i for $i = 1 \ldots L$. The moduli set $\{m_1, m_2, \ldots, m_L\}$ defines the RNS and consists of relatively prime moduli, m_i. The dynamic range is given as $M = \prod_{i=1}^{L} m_i$. Of particular interest are the moduli of the type 2^n and $2^n \pm 1$, where n is an integer greater than 1, which enable the modular operations of addition and multiplication to be simplified compared to the general moduli case. This Letter presents a technique that exploits the simple modulo reduction operation of these moduli to find a low complexity multiplier block.

Proposed technique: In the case of a modulus m, a directed graph can be obtained with $m-1$ nodes labelled v_1 to v_{m-1}. An edge going from node v_i to v_j is present when $|j/i|_m$ exists where $0 < i, j < m$ and $i \neq j$. The cost $c(i, j)$ associated with the edge going from node i to node j defines the cost in terms of modulo additions to compute a modulo multiplication by $|j/i|_m$. This cost represents the cost of obtaining $|x \times j|_m$ given $|x \times i|_m$ where $0 \leq x < m$. For example, with $m = 31$, the cost of going from v_2 to v_3 is the number of modulo additions needed to compute $|3/2|_{31} = |3 \times 16|_{31} = |17|_{31}$. Multiplication by 17 (CSD representation is 10001) requires one modulo addition, so $c(2, 3)$ is 1. The edge costs used are based on a CSD representation. There are no costs associated with multiplication by a power of 2. Multiplying by a power of 2 with moduli of type $2^n - 1$ only requires a rotation of bits, and with moduli of type $2^n + 1$ requires rotating inverted bits and including a constant in the output [4]. Taking another example, the cost of going from v_1 to v_{15} is the cost of multiplying by $|15/1|_{31} = |15|_{31}$. The CSD representation is $1000\bar{1}$ which requires one modulo addition. However, $|15|_{31} = |-16|_{31}$, which has a CSD representation of $\bar{1}0000$ requiring a negation and a shift which has zero cost, i.e. $c(1, 15) = 0$. Negation of moduli of the types used involves bit inversion and the addition of a constant in the case of 2^n and $2^n - 1$. The resulting constants can be accumulated and added at the filter output. The cost of using inverters is deemed negligible relative to adder costs.

Having obtained a directed graph with all possible edges, a minimum spanning tree (MST) is then generated. An MST is a rooted directed tree, such that the sum of $c(i, j)$ for all (i, j) in the graph is minimised. The MST is defined as a graph which connects, without any cycle, all nodes with $m-2$ edges, i.e. each node, except the root (v_1), has one incoming arc. A well-known solution for finding an MST is the Chu-Liu/Edmonds algorithm, which is used in this work. Fig. 1 shows the resulting MST for the modulus 31. The significance of the MST for a modulus m is that it provides the minimum cost to obtain the modulo product of x by all values from 1 to $m-1$. For example, using Fig. 1, the resulting hardware to obtain all products is shown in Fig. 2. Table 1 details the number of modulo additions needed for moduli up to 257. The Delay entry gives the worst-case delay in terms of number of modulo additions.

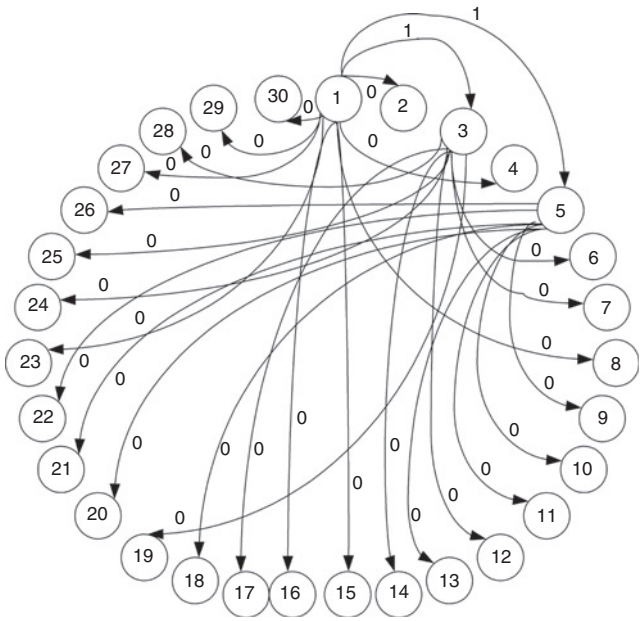

Fig. 1 *Minimum spanning tree for modulus 31*

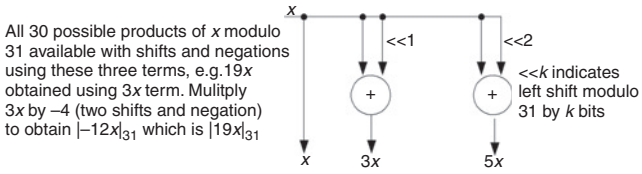

Fig. 2 *Hardware to compute all constant multiplications for $m = 31$*

Table 1: Number of modulo additions and delay costs for moduli

m	5	7	8	9	15	16	17	31	32	33
Adds	0	0	1	1	2	3	1	2	7	4
Delay	0	0	1	1	1	1	1	1	2	2
m	63	64	65	127	128	129	255	256	257	
Adds	7	15	6	8	33	11	25	66	15	
Delay	2	2	2	2	3	3	3	3	3	

Implementation: Performing modulo addition requires the extra step of modulo reduction that is not needed with binary addition. Efficient modulo adders exist based on carry propagate addition [5]. Alternatively carry-save modulo adders could be used. The modulo reduction operation is simply the feedback of the most significant carry out bit for $2^n - 1$ modulo addition and the feedback of an inverted carry bit [4] for $2^n + 1$ modulo addition. Another key issue with modular arithmetic is the conversion back to a weighted representation. For the purposes of the moduli used in this Letter, the best known architecture is that given in [6] which requires just $L+1$ binary additions for L moduli, where the wordwidth is approximately equal to the dynamic range.

Results: Using Table 1 and the CRT in [6], the moduli set with the optimum area cost is obtained by exhaustive search. For moduli of type 2^n and $2^n - 1$, n full adders (FAs) are used for each modulo addition, while $n+1$ are used for the $2^n + 1$ case. Table 2 lists the moduli set for a given dynamic range, the cost of the multiplier block in terms of FAs and the worst-case delay (for multiplier block) in terms of modulo additions. The bracketed entry is the total cost in FAs of the multiplier block and the CRT.

Though early work on this topic focused on comparing the actual number of additions in the multiplier block, recent work [2, 7] has actually quantified the results in terms of FAs. Table 3 shows the area

costs for a 16 bit coefficient example for the best method in [2] and the proposed method in this Letter. The RNS moduli set with a 44 bit dynamic range (sufficiently large) is used for comparison purposes. The results presented in [7] cite an average 32% improvement over all the examples in [2], whereas the average reduction using the proposed technique in this Letter is 73%. The MST RNS approach exhibits a unique feature in that the cost is independent of filter size, but a function of the dynamic range of the filter.

Table 2: Optimum moduli sets for given dynamic range

Moduli set	Dynamic range	Area (FAs)	Delay
{5, 7, 8, 9, 17, 31}	20	22(117)	1
{7, 8, 9, 17, 31, 65}	24	64(175)	2
{5, 7, 9, 16, 17, 31, 127}	28	87(244)	2
{7, 9, 16, 17, 31, 65, 127}	32	129(294)	2
{5, 7, 9, 16, 17, 31, 127, 257}	36	222(452)	3
{7, 9, 16, 17, 31, 65, 127, 257}	40	264(502)	3
{7, 17, 31, 32, 65, 127, 129, 257}	44	371(679)	3

'Area' entry indicates cost of multiplier block; brackets are total cost of multiplier block and CRT

Table 3: Comparison with 16 bit coefficient example in [2] using FA count

Filter length	HSSE (Example 1) [2]	Proposed RNS using MST	Percentage reduction
260	4710	679	85
610	8600	679	92
940	12 090	679	94
1180	13 042	679	94

Conclusions: Using a minimum spanning tree algorithm with relatively small moduli of the type $2^n \pm 1$ and 2^n, it is possible to obtain very low complexity multiplier blocks for FIR filters. This is possible by using small values for n and exploiting the ease of modulo reduction for such moduli. Including the cost of the CRT in addition to the multiplier block, the proposed approach offers a large complexity reduction over recent results in the literature.

16 July 2006
Electronics Letters online no: 20062230
doi: 10.1049/el:20062230

R. Conway (*ECE Department, University of Limerick, Plassey Technological Park, Limerick, Ireland*)

E-mail: richard.conway@ul.ie

References

1 Dempster, A.G., and Macleod, M.D.: 'Use of minimum-adder multiplier blocks in FIR digital filters', *IEEE Trans. Circuits Syst. II*, 1995, **42**, (9), pp. 569–577
2 Vinod, A.P., and Lai, E.M.K.: 'On the implementation of efficient channel filters for wideband receivers by optimizing common subexpression elimination methods', *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, 2005, **24**, (3), pp. 295–304
3 Muhammad, K., and Roy, K.: 'A graph theoretic approach for synthesizing very low-complexity high-speed digital filters', *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, 2002, **21**, (2), pp. 204–216
4 Wrzyszcz, A., and Milford, D.: 'A new modulo 2a +1 multiplier'. IEEE Int. Conf. on Computer Design, 1993, pp. 614–617
5 Mohan, A.: 'Residue number systems: algorithms and architectures' (Kluwer Academic, 2002)
6 Conway, R., and Nelson, J.: 'New CRT based RNS converter using restricted moduli set', *IEEE Trans. Comput.*, 2003, **52**, pp. 572–578
7 Maskell, D.L., and Liewo, J.: 'Hardware-efficient FIR filters with reduced adder step', *Electron. Lett.*, 2005, **41**, (22), pp. 1211–1213