UNIVERSITY OF WESTMINSTER

WestminsterResearch
http://www.wmin.ac.uk/westminsterresearch

Design guidelines for reconfigurable multiplier blocks.

Suleyman Demirsoy
Andrew Dempster
Izzet Kale

Cavendish School of Computer Science

Copyright © [2003] IEEE. Reprinted from 2003 IEEE International Symposium on Circuits and Systems, pp. 293-296.

This material is posted here with permission of the IEEE. Such permission of the IEEE does not in any way imply IEEE endorsement of any of the University of Westminster's products or services. Internal or personal use of this material is permitted. However, permission to reprint/republish this material for advertising or promotional purposes or for creating new collective works for resale or redistribution must be obtained from the IEEE by writing to pubs-permissions@ieee.org. By choosing to view this document, you agree to all provisions of the copyright laws protecting it.

The WestminsterResearch online digital archive at the University of Westminster aims to make the research output of the University available to a wider audience. Copyright and Moral Rights remain with the authors and/or copyright owners. Users are permitted to download and/or print one copy for non-commercial private study or research. Further distribution and any use of material from within this archive for profit-making enterprises or for commercial gain is strictly forbidden.

Whilst further distribution of specific materials from within this archive is forbidden, you may freely distribute the URL of WestminsterResearch. (http://www.wmin.ac.uk/westminsterresearch).

In case of abuse or copyright appearing without permission e-mail wattsn@wmin.ac.uk.

DESINGN GUIDELINES FOR RECONFIGURABLE MULTIPLIER BLOCKS

Süleyman Sırrı Demirsoy, Andrew G. Dempster and Izzet Kale

Applied DSP and VLSI Research Group, Department of Electronic Systems
University of Westminster, 115 New Cavendish St, London, W1W 6UW, UK
Tel: +44 20 7911 5000 - 3611(ext) e-mail: demirss@cmsa.wmin.ac.uk

ABSTRACT

The newly proposed reconfigurable multiplier blocks offer significant savings in area over the traditional multiplier blocks for time-multiplexed digital filters or any other system where only a subset of the coefficients that can be produced by the multiplier block is needed in a given time. The basic structure comprises a multiplexer connected to at least one input of an adder/subtractor that can generate several partial products, leading to better area utilization. The multiplier block algorithm complexity of a design increases logarithmically as the number of the multiplexers is increased. Design guidelines for the maximum utilization of the reconfigurable multiplier block structures are also presented.

1. INTRODUCTION

Multiplier blocks are very efficient ways of implementing fixed multipliers for digital filters. The multiplication can be performed by means of shifts and additions of the multiplicand defined by the multiplier, or coefficients in the case of filters. By exploring the common sub-expressions in terms of numbers forming the coefficients, the redundancy in the filter bank implementation can be significantly reduced [1],[2].

Another common sub-expression elimination method employs Canonical Signed Digit (CSD) representation to reduce the number of '1's in the coefficient representation, and seeks common bit patterns among the coefficients [3].

These techniques mainly target the problem of parallel implementation where several multiplications are performed in parallel. For time-multiplexed filtering, although they still can produce better solutions than a general purpose multiplier, there will be redundant parts in the multiplier block that are not active during a certain time slot. This problem is illustrated in Figure 1. Figure 1a depicts a fully parallel FIR filter implementation. Its corresponding time-multiplexed implementation is given in Figure 1b. The multiplier block implementation for all coefficients shown in Figure 1c is time multiplexed to choose a specific partial product. Therefore all partial products except the one that is selected remain unused.

The Reduced Coefficient Multiplier (RCM) technique proposed in [4] offers a very efficient solution to this problem for the XILINX Virtex FPGA implementations where a Look-up Table (LUT) is programmed to have some extra combinational logic including a multiplexer at one input of the adder/subtractor (called a cell) without any extra area cost.

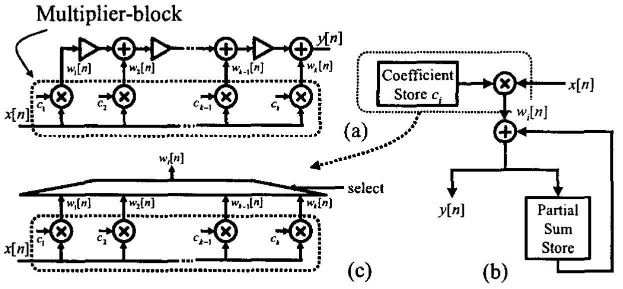


Figure 1 (a) Fully Parallel FIR filter implementation, (b) Time-multiplexed implementation (c) implementation of multiplier in 1b as a multiplier block

Although there are certain limitations in connection with multiplexer, its cost is effectively reduced by distributing the multiplexer stage at the output of the multiplier block in Figure 1c to the individual adder stages inside the multiplications. This idea is demonstrated in Figure 2. The edge values represent multiples of input after shifting is performed. Three LUTs have to be used in Figure 2a where the partial products 5 and 9 are generated separately and multiplexed afterwards. When the multiplexer is placed at one input of the adder, two partial products are generated by the same adder and the resulting circuit costs only one LUT.

The method reported in [4] to group the partial products together to form all the multiplications needed at the output of the last adder stage is based on Hartley's method [3]. Different Signed Digit (SD) representations for the coefficients are investigated and common sub-expressions that can be grouped in a single stage are identified by considering different cell definitions. Fifty eight (58) possibly useful cell implementations using a single LUT for the purpose of RCM design have also been reported in [4].

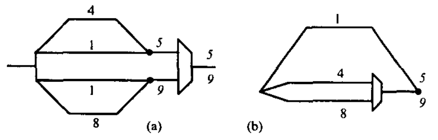


Figure 2 Two alternative implementations of a multiplier block that generates partial products for a time-multiplexed implementation

This paper describes the Reconfigurable Multiplier Blocks (ReMB) as an extension of RCM that target custom VLSI implementation as well as FPGAs. The design guidelines for the ReMB technique are also discussed. Section 2 gives the details of the structure of ReMB. Section 3 investigates several approaches for an algorithm and discusses design guidelines for an optimal result. Section 4 gives an example implementation. Section 5 concludes the paper.

2. RECONFIGURABLE MULTIPLIER BLOCKS

The basic structure of the ReMB is given Figure 3. It consists of a multiplexer connected to at least one input of the adder (in the rest of the text, an adder will refer to an adder or subtractor or an adder/subtractor). The generalized form consists of an adder with n inputs each of which can have a multiplexer with a different number of inputs. The dashed lines mean optional elements. Two particularly useful forms deduced from the generalized form are given in Figure 3b and 3c. The area of a 2x1 multiplexer is smaller than a full-adder circuit for custom VLSI implementation. As the number of inputs of the multiplexer or the number of the multiplexers increases, the area overhead increases as well as the functionality. For example the basic structure given in Figure 3b can produce double the partial results produced by the one given in Figure 3c.

A subset of the basic structures is suitable for Virtex FPGA implementation where the extra area cost of the multiplexer is zero. These forms are three cell definitions proposed in [4] for the RCM technique having the form of Figure 3c with inputs B, C (the inputs to the multiplexer) and A. Their functionality is shown in Table 1 for the different values of the select line.

The inputs to the basic structure are either the input ($x[n]$) to the multiplier block or another partial product generated by a similar basic structure or the shifted forms of them. The construction of the coefficients is similar to the normal multiplier blocks where partial products generated at each node (output of an adder) build up to coefficients. The difference of the ReMB is that the ReMB can produce more than one partial product at the output of the basic structure depending on the select signal for the multiplexer and the inputs to the multiplexer. Figure 4 shows an example design of a single basic structure, capable of producing a combination of four different partial results from signal X.

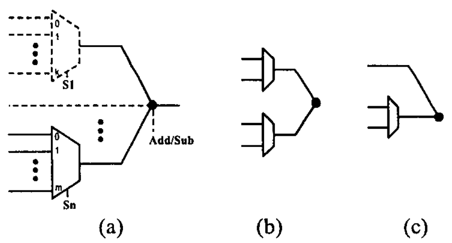
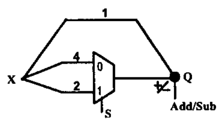


Figure 3 The basic structure of the ReMB (a) generalized form, S1-Sn being the select lines of the multiplexers, add/sub being the operation select signal of the adder/subtractor., (b) and (c) are mostly employed forms.

Select	Cell 1	Cell 2	Cell 3
0	A+B	A-B	A+B
1	A+C	A-C	A-C

TABLE 1 The operation of cells for different select values



S	Add/sub	Q
0	0	5X
0	1	-3X
1	0	3X
1	1	-X

(b)

Figure 4 (a) A basic structure with all of its inputs connected to X with the specified shift values, (b) Different output (Q) values as the select signals change.

The cascade of two basic structure, one of them taking all of its inputs from the input signal to the ReMB ($x[n]$), can be in several different forms as depicted in Figure 5. These structures are all topologically different and can generate distinct combinations of partial products that are not covered by another cascaded form. If we assume that the adder can perform only addition, the first stages in the cascaded forms would have two different partial products at their outputs. Having two different partial products as their inputs, the second stages of the cascaded forms would generate four different partial products at their outputs for Figure 5a, b, c, e and three different partial products for Figure 5d. Adding one more basic structure to the cascade line would result in producing around twice as many partial products. However, the number of topologically different connections of three basic structures as used in Figure 5 would be a hundred. This figure would continue to exponentially increase if the number of inputs to the multiplexers was increased or a different basic structure was used (i.e. Figure 3b).

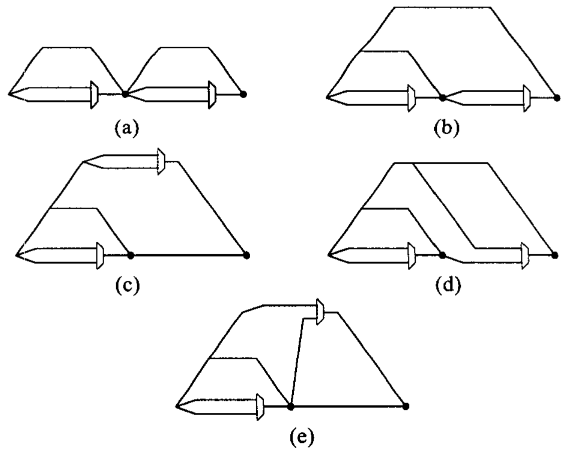


Figure 5 Five different cascaded forms of the two basic structures given in Figure 3c.

The cascaded forms are classified into two groups as regular forms and hybrid forms. Regular forms are the ones with the multiplexers having their input from the same node (Figure 5a, 5b and 5c). Hybrid forms have multiplexers taking their inputs from different nodes (Figure 5d and 5e). For bigger cascaded structures, the number of hybrid forms is much more than the regular forms. Regular forms are easy to handle during an automated design process. However hybrid forms offer a greater degree of flexibility in the design solution space for a given set of coefficients as will be shown in the next section.

3. DESIGN GUIDELINES

The optimal ReMB design for a given coefficient set requires a well defined procedure that can explore the capabilities of a particular basic structure in full. The partial product space gets extremely large as the size of the coefficient and/or coefficient set increases. The main problem is to identify the common partial products or sub-expressions that can be mapped to a single basic structure and use the minimum of such basic structures to build all the coefficient values.

The application of the ReMB to the time-multiplexed filter structures requires the implementation of multiple coefficient values with a single output as shown in Figure 1c. However, it is also possible to apply it to filter architectures with multiple outputs as well. The procedures to build the ReMB designs for these cases have to be different to get an optimal outcome.

The case of Multiple Input Single Output (MISO) applications has been investigated for RCM designs in [4]. The algorithm proposed starts combining sub-expressions into groups that suit one of the 58 cell definition until no sub-expression is left out. It tries to minimize the number of groups of sub-expressions hence the number of cells used. The flexibility of having 58 possible cell combinations (most of them operate on two inputs A,B and generate an arithmetical combination of them like A+B, 2A, 0, A-B) simplifies the grouping of sub-expressions but makes the decision of the minimisation of the number of groups more difficult. For the cells that accept three inputs as shown in Table 1, B and C inputs come from the same node [4]. In other words, the regular structures are taken into account but the hybrid forms are not utilized due to computational complexity.

One of the multiplier block algorithms Reduced Adder Graph (RAG-n), performs an exhaustive search for all possible partial results that can be formed with one adder and then continues to do so until the targets are formed. The number of adders being the cost function, the algorithm gives optimal (fewest adders) results for the sets that are covered by exhaustive search. The second part of the algorithm tries to add the coefficients to the multiplier block using the minimum number of adders individually. This part is not optimal, however gives good results up to 12-bits word-length [2]. RAG-n is still the best available algorithm for the parallel implementation of fixed multipliers.

A similar approach of starting with an exhaustive search of all partial products that can be combined on a structure and then adding the groups of partial products using the minimum number of basic structures would lead to a better design than trying to group the partial products and coefficients for the start. The limitation here is that the size of the set to undertake an exhaustive search increases exponentially as the number of basic structures in the design increases.

Table 2 displays the set sizes of exhaustive search spaces of one (Figure 4a) and two (Figure 5) basic structures for the integer numbers up to a word-length of 10-bits. The exhaustive search for all possible outputs is performed for both positive and negative numbers and recorded with the type of structure and the inputs to the structure. The sets are then filtered for repetitive and unwanted outputs. The basic structures are restricted to the subset that fits into a single half slice in a Virtex FPGA. As observed, the set size dramatically increases for the cascaded structures. For the cascade of three basic structures, the number of different forms becomes 100 as mentioned earlier. It is not feasible to do an exhaustive search for this large number of structure, although if done, it would benefit to achieve a smaller design.

Structure	Set Size	Percent of Total
Figure 4a	471	N/A
Figure 5a	22754	3.8 %
Figure 5b	62003	10.6 %
Figure 5c	126850	21.6 %
Figure 5d	259295	44.4 %
Figure 5e	114335	19.6 %
Total of Figure 5	585237	N/A

TABLE 2 The set sizes of the exhaustive search space of the numbers up to 10 bits word-length for the given structures.

The percentage column on Table 2 indicated the contribution of a particular cascaded form to the total number of sets of two basic structures. The outputs from regular structures add up to 36 % of all possible outputs where only a single hybrid structure contributes 44.4 %. As the number of basic structures increase, the percentage of the hybrid forms will become greater. Therefore, it is essential to incorporate to hybrid forms into the design methodology effectively.

In the context of multiplier blocks, the graph that produces the coefficient was not of importance since the primary goal was to construct the coefficients by sharing intermediate results or partial products [5]. Two different graphs generating the same coefficient were not considered separately. Secondly, the even coefficient values were treated after dividing them by two until an odd number is reached. The outputs from a multiplier block are always odd numbers that are shifted afterwards to generate the even coefficient [2], [5].

Both of the above issues have to be taken into consideration in the design of the ReMB. The structures are of primary importance, since the input value and the shift value on the common input line (the input line without the multiplexer) and the operation type of the basic structure affects the decision of the other inputs to the basic structure. Shifting the output of ReMB at an extra stage is not feasible due to the nature of the ReMB where a single node produces more than one partial product. Shifting only one of the partial products and not the others requires a separate multiplexer to select between the original and shifted forms. Instead, the even numbers can be handled like the odd numbers and generated throughout the ReMB design.

An efficient ReMB design should employ the aforementioned guidelines. The search of grouping of partial products that builds up the coefficients would cover all the different possible ways of constructing a coefficient. After identifying these possible ways for all coefficients, the partial products that have common input values and shift values in their formation should be grouped in a basic structure. As the number of basic structures increase, the possible outputs from the nodes in the ReMB should be taken into account to construct the remaining coefficients. The number of output lines from the ReMB design effects the decision of combining the partial products. Two coefficients that reside at different output nodes of the ReMB design should be constructed in such a way that, the last stage of the cascaded structure should be different for them. However, separating the generation of partial products before the last stage should give better results for some cases and should be considered.

4. ReMB DESIGN EXAMPLES

An example to demonstrate the effectiveness of the proposed ReMB technique is given below. Figure 6 shows the recursive loop of an 8-point Goertzel recursive DCT structure given in [6]. This multiplier has 8 different coefficient values, characterized as:

$$2\beta_k = 2Cos(k\pi/8) \quad \text{for } k \in [0,7] \qquad (1)$$

The coefficient values for k=5, 6 and 7 are the negatives of the coefficients for k=3, 2 and 1 respectively. Therefore the multiplier can generate the multiplications for $k \in [1,3]$ and the results can be negated afterward at the next adder.

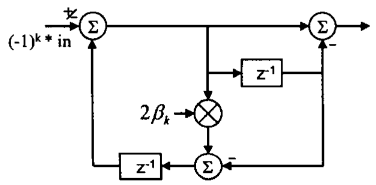


Figure 6 Recursive part of a Goertzel DCT implementation

The ReMB design of this multiplier for a coefficient word-length of 12-bit is shown in Figure 7. The outputs are shifted 9-bits to right to get the result of multiplication with the coefficients. It consists of four basic structures that occupy four half slices for a single bit input in a Virtex FPGA implementation. This design is highly efficient when compared to a classical multiplier block implementation by the RAG-n algorithm [2] that would occupy seven adders and extra multiplexer stages at the output. It also occupies one less half slice in comparison to the RCM design of the same coefficients given in [4]. This saving is due to the exploitation of the hybrid forms of cascades in the ReMB designs as depicted in Figure 7.

Another ReMB design is for the newly proposed reconfigurable recursive DCT kernel given in [6] where the 12-bit coefficients are smaller in magnitude due to the realization of different Goertzel configurations for different frequency bins. As seen from the Figure 8, this ReMB design has three basic structures, again connected in hybrid form cascades. It occupies three half-slices for a single bit input in Virtex FPGA.

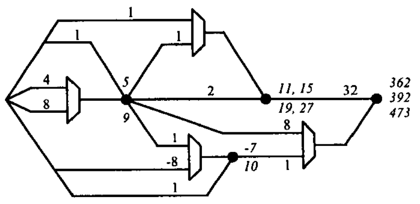


Figure 7 ReMB design of the DCT loop multiplier. The italic numbers are the partial products generated at each node.

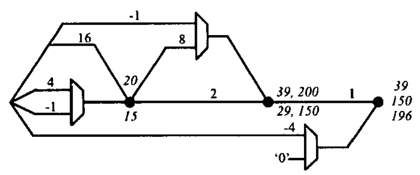


Figure 8 ReMB design for the reconfigurable DCT kernel in [6].

5. CONCLUSION

Reconfigurable Multiplier Blocks (ReMB) offer significant savings in the area of fixed multipliers for time-multiplexed systems for both custom VLSI and FPGA implementations. The cascade forms of the basic structures have to be fully utilized for achieving smallest area. The choice of the basic structure is crucial to reduce the area for FPGA environments. Several factors that are not of importance in the multiplier block design have to be taken into account when designing ReMB, such as even number generation and the structure that generates the partial products. A design methodology employing an exhaustive search as a start would give better results as shown with an example in this paper.

6. ACKNOWLEDGEMENT

The authors would like to thank Richard H. Turner for his generosity in providing us with a copy of his PhD thesis.

7. REFERENCES

[1] Bull D.R. and D.H. Horrocks, "Primitive operator digital filters", *IEE Proceedings-G*, vol. 138, no.3, pp.401-412, June 1991

[2] Dempster A. G. and Macleod M. D., "Use of minimum-adder multiplier blocks in FIR digital filters", *IEEE Trans. CAS-II*, vol. 42, no. 9, pp. 569-577, Nov. 1995

[3] Hartley R., "Optimization of canonic signed digit multipliers for filter design", *IEEE Proc. of Int. Symp. On Circuits and Sytems*, vol. 4, pp. 1992-1995, July 1991

[4] Turner R. H., "Functionally diverse programmable logic implementations of digital signal processing algorithms", PhD Thesis, Queen's University of Belfast, August 2002

[5] Gustafsson O., A. Dempster and L. Wanhammar, "Extended results for minimum-adder constant integer multipliers", *IEEE Proc. of ISCAS'2002*, vol.1, pp. 73-76, May 2002

[6] Demirsoy S. S., et. al, "Reconfigurable implementation of Recursive DCT kernels with reduced quantization noise", *submitted to IEEE ISCAS'2003*