Using carry-save adders in low-power multiplier blocks.

Viv Bartlett
Andrew Dempster

Cavendish School of Computer Science

USING CARRY-SAVE ADDERS IN LOW-POWER MULTIPLIER BLOCKS

V.A. Bartlett, A.G. Dempster

University of Westminster, 115 New Cavendish St, London W1W 6UW, UK
Tel: (44) 20 7911 5146, Fax: (44) 20 7580 4319, email: v.bartlett@westminster.ac.uk

ABSTRACT

For a simple multiplier block FIR filter design, we compare the effects on power consumption of using direct versus transposed direct forms, tree versus linear structures and carry-save (CS) versus carry-ripple (CR) adders (for which multiplier block algorithms have been designed). We find that tree structures offer power savings, as expected, as does transposition in general but not always. Selective use of CS adders is shown to offer power savings provided that care is taken with their deployment. Our best result is with a direct form CR/CS hybrid.

The need for new multiplier-block design algorithms is identified.

1. INTRODUCTION

Multiplier blocks are structures made up of interconnected adders, configured to produce products of an input multiplicand with one or more coefficients. They are hardware-efficient replacements for dedicated multipliers, where fixed-point constant coefficients are required. Multiplier block design-algorithms [1]-[4] have, to date, mainly focussed on minimising the number of adders required to perform one [2][3] or more [1]-[4] multiplications of a single multiplicand. The latter is useful in FIR filters, IIR filters [5] and filter banks [6]. All these algorithms exploit redundancy in the shift-and-add multiplication process. The implementation cost has conventionally been measured only in number of adders (which also includes subtractors) as shifts can be performed by wiring and are essentially "free".

Recently, the low-power credentials of multiplier blocks have been investigated. Although in general, fewer components (in this case adders) implies lower power, specific examples particularly when high *logic-depth* is encountered have proved that this is not always the case due to the increased likelihood of glitch propagation [7][8]. The correlation between logic depth and power consumption in static CMOS circuits has been known for some time [9] and multiplier-block algorithms designed to minimize depth have recently been proposed in [10]. For low-power summation of partial products, tree structures [11][12] have often been advocated due to their reduced logic depth and hence glitch-count [8][13].

All of the multiplier block work described above is based on some simple assumptions:

i) each "adder" in the network has a single output
ii) all adders are effectively the same as each other
iii) adders have the same cost as subtractors.

The first assumption implies the use of adders, such as carry-ripple (CR) adders, that fully-resolve their carries. Adders that do not resolve their carries, such as carry-save (CS) adders and 4-2 counters [14] whilst producing more than one output, have been shown to be more power efficient in some applications than CR adders [13].

The contribution of this paper is to show that the universal application of CS adders does not, in general, yield the lowest power solution. However, their application in selected positions in the multiplier block can considerably reduce power consumption.

In this paper we use transition-count as a measure of the relative power consumption of the circuits.

2. SINGLE MULTIPLICATION: ARCHITECTURAL CONSIDERATIONS

Figure 1 shows several possible implementations of shift-and-add multiplication by 25. For example, in Figure 1(a), the first adder sums two products (x2 and x1 both of which can be obtained for "free" by hard-wiring) of Md, the multiplicand. The result, 3.Md, is then shifted by 3 bits to produce 8.(3.Md) before being added to 1.Md, yielding 25.Md at a cost of two adders.

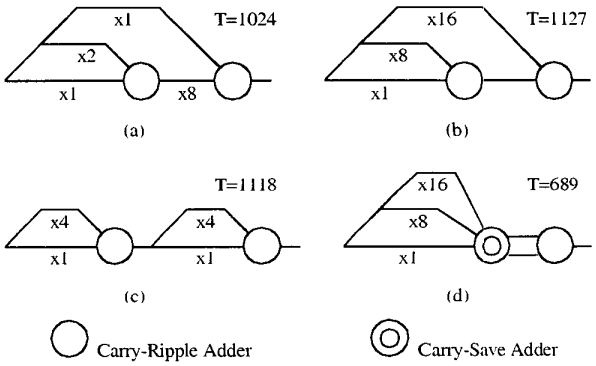

Figure 1. Different shift and add architectures for implementing multiplication by 25.

Figure 1(a-c) show multiplier block structures using two CR adders whereas Figure 1(d) uses a CS adder whose output is resolved by a CR adder. Figure 1(a) and (b) are both fundamentally the same structure with the shifted multiplicands summed in a different order.

0-7803-6685-9/01/$10.00©2001 IEEE

Also shown in Figure 1 are the transition-counts, T, at the sum and carry outputs of the full-adders for each of the structures when computing the products of 20 uniformly distributed 12-bit random numbers.

These figures have been derived from VHDL simulations using full-adders modelled as two independent sum and carry producing circuits with propagation delays set in the ratio 6:5 respectively. Glitches of width less than ~1/3 of these delays are not propagated, thereby approximating realistic CMOS behaviour.

Important inferences that can be drawn from this small study are:

- In terms of power consumption, order of summation can have a more significant impact than choice of structure. (Although an interesting topic for further investigation, this idea is not pursued further here).
- Use of CS adders offers the potential for significant power savings in multiplier block structures. This is consistent with recent array-multiplier studies which show CS to be more power efficient than CR [13].

The use of CS adders to reduce transitions and in multiplier-block based digital filters is investigated below.

3. MULTIPLIER BLOCKS: FIR FILTERS

3.1 Various Possible Structures

A multiplier-block based FIR filter with the simple set of coefficients {3, 11, 25} is shown in Figure 2.

The basis for the multiplier-block algorithms published thus far, whether optimised for adder-count [1][6] or logic depth [10] has been the all-carry-ripple, Transposed-Direct-Form (TDF) as shown in Figure 2(a). In this example, the maximum logic depth is 3, compared to 5 in the Direct-Form (DF) structure of Figure 2(b), suggesting a power advantage for the TDF. However, when we examine the use of CS adders in these structures we find that the TDF (Figure 2(c)) has certain problems. In particular, delay registers become double-width. By contrast, the DF suffers no such disadvantage with CS adders and is relatively well suited to their use. In both forms, the requirement to fan-out a CS output implies an increase in adder-count. This increase can be eliminated by using a CR adder where the output requires fanning-out, as shown in Figure 2(e) and (f), yielding the same adder-count as the all-CR case.

For the DF cases (Figure 2(b),(d),(f)) we see that there is an implicit high transition-count resulting from the long adder-chains. These result from the fact that in multiplier block design algorithms, the (shifted) multiplicand is used as often as possible, leading to a low-depth graph. There are therefore several nodes at the input of the TDF filter, which transpose to a cascade of adders at the output of the DF filter. A simple method of reducing the power is to rearrange these adders into a "tree". For instance, Figure 2(g) is a "treed" version of Figure 2(b). This reduces the logic depth with a beneficial impact on power consumption, particularly if the tree can be balanced [9]. The ability to reduce a network of n adders to a balanced tree with depth $\lceil \log_2(n) \rceil$ [11] is hampered by the asymmetries introduced by the multiplier design process. There may be scope to make the tree better balanced at the expense of more adders. With the TDF, we also found that by including the 'structural' adders (i.e. those not part of the actual multiplier block) into the trees, a reduction in logic depth can be obtained at several places in our example.

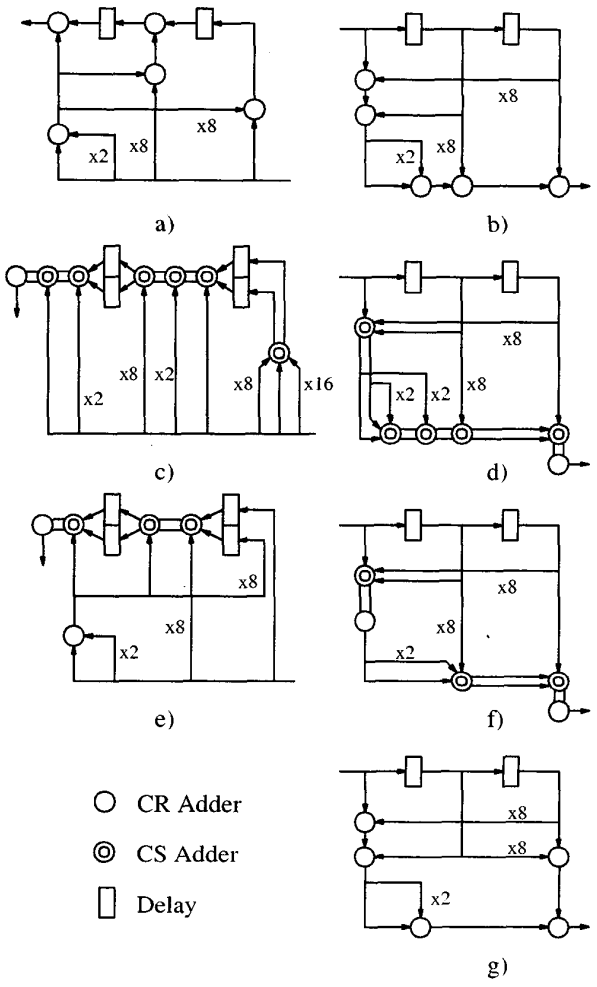

Figure 2. Transposed form and direct form respectively: a) and b) all-carry-ripple, c) and d) all-carry-save, e) and f) carry-save with resolution of fanned-out outputs.

An analysis of Figure 1 indicates that in these structures CS adders are of particular benefit when all inputs arrive in synchrony (see Figure 4). This knowledge was used in some of the filter designs.

3.2 Filter Design Example

We used the Matlab "remez" routine to design an order 9 FIR lowpass filter with identical ripple in passband and stopband with normalised cutoff frequencies 0.13 and 0.22. The coefficients: {-0.0086, 0.1598, 0.1303, 0.1563, 0.1725, 0.1725, 0.1563, 0.1303, 0.1598, -0.0086} give integer values {-4, 82, 67, 80, 88,

88, 80, 67, 82 -4} after scaling by 512 and rounding. These coefficients can be synthesised using the 5-adder multiplier block shown in Figure 3.

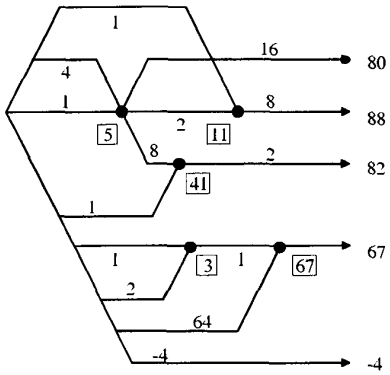

Figure 3. The multiplier block for our example.

A FIR filter based on this multiplier block was modelled using the following structures:

Filter 1: Direct form, all-CR (see Figure 2(b))

Filter 2: Transposed form, all-CR (see Figure 2(a))

Filter 3: Direct form, all-CS (see Figure 2(d))

Filter 4: Direct form, CS, but CR where outputs are distributed (see Figure 2(f))

Filter 5: Direct form, "treed", all-CR (see Figure 2(g))

Filter 6: Transposed form, "treed", all-CR

Filter 7: Direct form, "treed", CS, but CR where outputs are distributed

Filter 8: Direct form, "treed", CR except where a CS adder can be used, with all inputs arriving synchronously.

4. RESULTS AND DISCUSSION

VHDL simulations of these filters based on the full-adder model described above were carried out.

The filters were fed with the same set of 20 random 12-bit values and adder sum and carry output transitions were counted as shown in Table 1.

Table 1. Filter type and transition-count

Filter	Transitions	Comments
1	8263	DF, all-CR
2	5998	TDF, all-CR
3	8488	DF, all-CS
4	6993	DF, CS some CR
5	5583	DF, tree, all-CR
6	5299	TDF, tree, all-CR
7	5350	DF, tree, CS some CR
8	4239	DF, tree, CR some CS

General comments that can be made about these results are:

i) Transposed Direct Form (2 and 6) is, in general, better than the Direct Form (1 and 5). It should be noted, however, that the power consumption of the delay registers (not included in this study) is substantially greater with the TDF.

ii) Comparing pairs of designs before and after "treeing" (1+5, 2+6, 4+7) shows (consistent with [9]) that trees use less power.

iii) Comparisons of similar structures using CR and CS adders (1 and 3) indicate that CS does not necessarily imply lower power. Modifying the CS design so that CR adders replace those producing outputs that are distributed to several places (4 and 7) significantly reduces power consumption.

iv) The best result of all is a mainly-CR design but with CS adders applied where all its inputs arrive in synchrony as shown in Figure 4. Interestingly, this is a Direct Form filter - the greater depth providing more opportunities for application of this energy-efficient CS adder configuration.

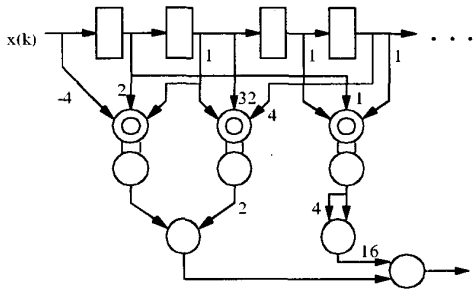

Figure 4. The filter with lowest transition-count.

5. CONCLUSIONS

Several known heuristics for low-power structures have been studied with regard to their application to multiplier-block based FIR filters.

Use of tree structures has been confirmed as being beneficial in all the considered cases especially when the tree can be well balanced.

Use of Transposed Direct Form over Direct Form is in most cases beneficial, but not always, particularly when a well-balanced Direct Form tree can be produced.

Use of Carry-Save adders offers significant benefits but only when their input transitions can be aligned closely in time. Use of CS adders when the output is reused requires extra adders and is to be avoided, due to the area and power penalties.

Of the considered designs, lowest energy consumption was found with a 'treed' Direct Form, hybrid design comprising both CR and CS adders – outperforming structures made exclusively with either.

For single multipliers, the order in which the shift-and-add process is performed seems to be as important as the multiplier architecture in terms of power minimisation.

6. RECOMMENDATIONS FOR FUTURE WORK

New approaches to algorithm design need to be considered such as:

- Algorithms that reduce the logic depth of the direct-form structure, (current algorithms produce circuits with long adder chains). Furthermore, a well-balanced tree may prove preferable to minimizing adder count and/or depth.
- Algorithms that favour structures where CS adders can be used, i.e. where three adder inputs arrive either synchronously or closely aligned in time.
- Algorithms that can take the Transposed Form's structural adders into account for tree-balancing.

As with CS adders, the use of 4:2 counters is also worthy of investigation. The findings of such a study should also be incorporated into the algorithm.

Further work is also required arising from the single-multiplier findings. Even within a given multiplier block graph, the variation in power consumption due to different vertex and edge labelling needs investigation and the implications absorbed into design algorithms.

7. REFERENCES

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

[2] A G Dempster and M D Macleod, "Constant integer multiplication using minimum adders", IEE Proceedings - Circuits, Devices and Systems, vol. 141, no 5, pp. 407-413, Oct 1994.

[3] A G Dempster and M D Macleod, "General algorithms for reduced-adder integer multiplier design", Electronics Letters, vol. 31, no 21, pp. 1800-1802, Oct 1995.

[4] A G Dempster and M D Macleod, "Use of minimum-adder multiplier blocks in FIR digital filters" IEEE Trans Circuits and Systems II, vol. 42, no. 9, pp569-577, September 1995.

[5] A G Dempster and M D Macleod, "IIR Digital Filter Design Using Minimum-adder Multiplier Blocks", IEEE Trans Circuits & Systems II - Digital & Analog Signal Processing, vol. 45, no. 6, pp. 761-763, June 1998.

[6] A G Dempster and N P Murphy, "Efficient Interpolators and Filter Banks using Multiplier Blocks", IEEE Trans. Sig. Proc. Vol. 48 no. 1, pp. 257-261, Jan. 2000.

[7] David H Horrocks and Yodchai Wongsuwan, "Reduced Complexity Primitive Operator FIR Filters for Low Power Dissipation", Proc ECCTD '99, Stresa, Italy, pp. 273-276, 1999.

[8] D Perello and J.Figueras, "RTL Energy Consumption Metric for Carry-Save Adder Trees: Application to the Design of Low Power Digital FIR Filters, PATMOS 99, pp. 301-311.

[9] A P Chandrakasan, and R W Brodersen, "Minimizing Power Consumption in Digital CMOS Circuits" Proc. of the IEEE, Vol. 83, No.4, pp. 498-523, 1995.

[10] A G Dempster, "Algorithms for Reducing Logic Depth in Multiplier Blocks", submitted to IEEE Trans C&S II.

[11] N G Kingsbury, "High-speed binary multiplier", Electronics Letters, vol. 7 no. 10, pp. 277-278, 1971.

[12] C S Wallace, "A suggestion for a fast multiplier", IEEE Trans Electronic Computers, vol. 13 pp. 14-17, Feb. 1964.

[13] F Moller, N Bisgaard and J Melanson, "Algorithm and Architecture of a 1V Low Power Hearing Instrument", Int. Symp. Low Power Elect. & Design (ISLPED99), pp. 7-11 1999.

[14] M Santoro, "SPIM: A Pipelined 64x64-bit Iterative Multiplier," IEEE J. Solid-State Circuits, vol. SC-24, pp. 487-493, 1989.

[15] J Rabaey and M Pedram "Low Power Design Methodologies", Kluwer Academic Publishers 1997.