# **Linköping University Post Print**

# Addition Aware Quantization for Low Complexity and High Precision Constant Multiplication

Oscar Gustafsson and Fahad Qureshi

N.B.: When citing this work, cite the original article.

©2009 IEEE. 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 to servers or lists, or to reuse any copyrighted component of this work in other works must be obtained from the IEEE.

Oscar Gustafsson and Fahad Qureshi, Addition Aware Quantization for Low Complexity and High Precision Constant Multiplication, 2010, IEEE Signal Processing Letters, (17), 2, 173-176. http://dx.doi.org/10.1109/LSP.2009.2036384

Postprint available at: Linköping University Electronic Press http://urn.kb.se/resolve?urn=urn:nbn:se:liu:diva-52893

# Addition Aware Quantization for Low Complexity and High Precision Constant Multiplication

Oscar Gustafsson, Member, IEEE, and Fahad Qureshi, Student Member, IEEE

Abstract—Multiplication by constants can be efficiently realized using shifts, additions, and subtractions. In this work we consider how to select a fixed-point value for a real valued, rational, or floating-point coefficient to obtain a low-complexity realization. It is shown that the process, denoted addition aware quantization, often can determine coefficients that has as low complexity as the rounded value, but with a smaller approximation error by searching among coefficients with a longer wordlength.

*Index Terms*—Addition, constant multiplication, quantization, subtraction.

# I. INTRODUCTION

**I** N many DSP algorithms multiplier coefficients are either floating-point numbers (e.g., from filter design algorithms), rational numbers (e.g., 1/3), or real numbers (e.g.,  $\sqrt{2}$  or  $\pi/8$ ). However, when implementing digital signal processing (DSP) algorithms fixed-point computations are often preferred over floating-point due to lower complexity and power consumption. The conversion from floating-point, rational, or real valued numbers to fixed-point can be seen as quantization of an infinitely long fixed-point representation. To avoid lengthy repetition we will in the following use floating-point, rational, and real numbers interchangeably to denote numbers that can not be exactly represented using fixed-point representation.

It should be noted that typically, one distinguishes between quantization of the data and quantization of the multiplier coefficients. Data quantization leads to round-off noise, which is usually modeled as an additive error signal, where the error signal is characterized as a stochastic process with properties depending on the type of quantization used. Coefficient quantization on the other hand leads to a static deviation from the ideal transfer function. It should be noted that data quantization is also often performed within the algorithm implementation to reduce the wordlength of the computations. Especially, for recursive algorithms this is required as, otherwise, the wordlength would grow indefinitely.

In this work we consider multiplication by a constant fixedpoint number approximating a number that can not be exactly represented with the same number of bits (or possibly not at all). Consider the case where we have a real valued number A that we want to approximate with a fixed-point value  $\hat{A}$ . For ease

The authors are with the Department of Electrical Engineering, Linköping University, SE-581 83 Linköping, Sweden (e-mail: oscarg@isy.liu.se; fahadq@isy.liu.se).

Digital Object Identifier 10.1109/LSP.2009.2036384

of presentation we will without loss of generality assume that 0 < A < 1. Using N fractional bits and proper rounding the approximation error,  $\epsilon$ , is

$$\epsilon = \left| A - \hat{A} \right| \le 2^{-(N+1)}. \tag{1}$$

It is throughout this work assumed that we should meet an approximation specification of N fractional bits, as in (1), although other measures can be dealt with in a similar way.

An unsigned fractional fixed-point coefficient,  $\hat{A}$ , represented using N fractional bits can be written as

$$\hat{A} = \sum_{i=1}^{N} a_i 2^{-i}$$
 (2)

where  $a_i \in \{0,1\}$ . Now, assume that a multiplication with a data, X is performed. The result is

$$Y = \hat{A}X = \sum_{i=1}^{N} a_i 2^{-i} X.$$
 (3)

The multiplication can then be performed as a sum where the input is shifted and multiplied by either 0 or 1, once for each bit of  $\hat{A}$ . In total there are N - 1 additions to compute the result. Note that for bit-parallel computation the shifts can be hardwired, and, hence, no logic cells are required for shifting. If the coefficient  $\hat{A}$  is known in advance the multiplication by 0 or 1 can be simplified to either 0 or X. Zero-valued data does not contribute to the sum. Therefore, the number of additions is directly proportional to the number of nonzero bits of  $\hat{A}$ .

Using a signed-digit (SD) representation we have  $a_i \in \{-1, 0, 1\}$ . Hence, each bit is now a ternary digit. As for the constant coefficient multiplier case we do not represent the coefficients explicitly as inputs to the multiplier, the complexity does not increase by introducing a third alternative for each position. Instead, it just leads to that some of the additions may be replaced by subtractions. As a subtraction has about the same complexity as an addition, for simplicity throughout this work we will refer to both as additions. The potential benefit of using a SD representation is that it is often possible to find a representation with fewer nonzero positions compared to using a binary representation. An SD representation with the smallest possible number of nonzero digits is referred to as a minimum signed-digit (MSD) representation. One MSD representation of special interest is the canonic signed-digit (CSD) representation. For a CSD representation we have  $a_i a_{i+1} = 0, \forall i$ . For each coefficient there are several possible SD representations. There may also be several MSD representations. However, the CSD representation is unique (hence, the name canonic), so if a CSD representation is found we know that it is also an MSD representation and the minimum number of nonzero positions is well established. The average number of nonzero positions in a

Manuscript received July 28, 2009; revised October 19, 2009. First published November 10, 2009; current version published November 25, 2009. The work of F. Qureshi was supported by the Higher Education Commission, Pakistan. The work of O. Gustafsson was supported by the Swedish Research Council and CENIIT, Linköping University. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Alfred Mertins.

CSD representation is asymptotically N/3 compared with N/2 for binary, while the maximum number of nonzero positions is N/2 for CSD compared with N for binary. Hence, the number of additions are on average reduced by using MSD/CSD representation compared to binary.

Over the years several algorithms have been proposed to design DSP algorithms with a few number of nonzero SD terms, often referred to as sum-of-powers-of-two (SOPOT) or signedpower-of-two (SPT) terms. Examples include specific digital filters [1], [2] and transforms [3], [4], as well as general DSP algorithms [5], [6]. The resulting realization is often called *multiplierless* as general multiplications are replaced by shifts and additions. In [2] the statistical properties of SD representations are investigated for multiplier coefficients. There has also been investigations on using SD representations with a low number of nonzero digits for data [7].

Despite the fact that the CSD representation is minimal it is still possible to find constant multiplication realizations using fewer additions compared to a straightforward shift-and add realization based on CSD [8]-[10]. In [8] an optimal approach was introduced and it was shown that all constant multiplications with coefficients with up to 12 bits can be realized using at most four additions. In [9] that approach was simplified and it was shown that at most five additions were required for up to 19 bits coefficients. In addition to the optimal approach, a heuristic was also introduced in [9] based around the idea that it is sometimes worthwhile to increase the number of nonzero signed-digit terms to reduce the number of additions. The generation of all signed-digit representation for a coefficient can be obtained as in [11]. Finally, in [10] an efficient heuristic was proposed, based on the heuristic in [9], to allow low complexity multiplication with arbitrary wordlength. In terms of theoretical results it has been shown that the maximum number of additions grows as  $N/\log_2 N$ , where N is the coefficient wordlength [12], [13], while at least  $\lceil \log_2 C(A) \rceil$  additions are required, where C(A) denote the number of nonzero SD terms for the coefficient A [9], [14].

The discussion in this paper is based on carry-propagation addition, i.e., addition of two numbers to yield a single result. A similar approach can be used for different types of additions, e.g., using high-speed redundant carry-save additions where the constant multiplications structures in [15] should be used instead. It should also be noted that the number of bits involved in each addition, and, hence, the number of full adder cells required, differs between the additions [9]. Furthermore, the number of cascaded additions may also be of interest to consider. It is possible to consider this as well during the search process described in the paper by simply adopting a different cost measure when selecting the best solution. The presented results focus on the number of additions only.

### II. ADDITION AWARE QUANTIZATION

If the allowed wordlength is increased with E fractional bits, the approximation error can be guaranteed to meet  $\epsilon \leq 2^{-(N+E+1)}$ . However, another way of using the additional fractional bits is to realize that there are exactly  $2^E$  different representable coefficients for which  $\epsilon \leq 2^{-(N+1)}$ , including the one obtained by rounding to N fractional bits. The basic idea in this work is to search these  $2^E$  and select the coefficient value that has the smallest approximation error for the allowed complexity. The allowed complexity is typically assumed to



Fig. 1. Possible coefficients with N correct fractional bits using (a) N fractional bits, (b) N + 1 fractional bits, and (c) N + E fractional bits.

be the same number of additions as required by the coefficient rounded to N fractional bits. We refer to this scheme as *addition aware quantization*. It should also be noted that in some cases it is possible to find valid representations that require a lower complexity compared to the rounded N fractional bits coefficient. This is further illustrated in Section III.

To further illustrate the fact that there are  $2^E$  different solutions consider Fig. 1(a) where the possible alternatives for N fractional bits are illustrated. Clearly, there is only one value, denoted  $\hat{A}_N$ , that meets the requirements. Now, increasing the resolution with one bit gives the case in Fig. 1(b), where an additional possible solution is available. The fact that it here happened to have a smaller approximation error is not crucial. Instead, we are interested in the fact that we have a second, alternative, approximation. Finally, the general case with E extra fractional bits is illustrated in Fig. 1(c).

There will be  $K_S$  extra coefficients where  $k \cdot 2^{-(N+E)}$ ,  $k = \{1, 2, 3, \ldots, K_S\}$  are subtracted from the N fractional bits approximation,  $\hat{A}_N$ ; see Fig. 1(c). Similarly, there will be  $K_A$  extra coefficients where  $k \cdot 2^{-(N+E)}$ ,  $k = \{1, 2, 3, \ldots, K_A\}$  are added to  $\hat{A}_N$ . If  $\hat{A}_N \ge A$  (as in Fig. 1(a)), then

$$K_S = 2^{E-1} + \left\lfloor \frac{\hat{A}_N - A}{2^{-(N+E)}} \right\rfloor$$
(4)

otherwise we have

$$K_A = 2^{E-1} + \left\lfloor \frac{A - \hat{A}_N}{2^{-(N+E)}} \right\rfloor$$
(5)

where the other term can be determined by  $K_S + K_A = 2^E - 1$ .

#### III. DESIGN EXAMPLES

In this section, we provide a number of examples illustrating the concept and results of addition aware quantization. The design examples also illustrate various ways of applying the addition aware quantization concept. For the addition costs we use the optimal results in [9] for up to 19-bits coefficients. For longer wordlengths the heuristic in [10] is used. The number of *correct fractional bits*, CFB, is defined as

CFB = 
$$-\log_2 |A - \hat{A}| - 1 = -\log_2 \epsilon - 1.$$
 (6)

Clearly, there is a tradeoff between the number of extra bits to search, and, hence, the offline computational complexity, and



Fig. 2. Required number of additions for some rational coefficients.

 TABLE I

 Results for Constant Multiplication Using Three Additions

Coefficient	Rounding		Addition aware quantization	
	CFB	Value	CFB	Value
$\sin \frac{\pi}{3} = \frac{\sqrt{3}}{2}$	11.396	$\frac{887}{1024}$	12.944	$\frac{7095}{8192}$
$\sin \frac{\pi}{4} = \frac{\sqrt{2}}{2}$	12.693	$\frac{181}{256}$	12.693	$\frac{181}{256}$
$\tan \frac{\pi}{16}$ [3]	13.029	$\tfrac{1629}{8192}$	13.377	$\tfrac{13039}{65536}$
$2\cos\frac{1336\pi}{4000}$ [16]	18.082	$\frac{32649}{32768}$	18.082	$\frac{32649}{32768}$
$2\cos\frac{1477\pi}{4000}$ [16]	11.219	$\frac{409}{512}$	11.385	$\tfrac{26163}{32768}$

the possible obtainable results, and, hence, the online computational complexity. It should be noted that eventually all newly introduced coefficients will have such a large number of nonzero positions that it is not possible to find realizations with the required number of additions [9], [11]. However, it is yet not known if there exists such a bound based on the number of fractional bits.

#### A. Rational Numbers

Multiplication with rational numbers (or division with integers) occurs frequently in some DSP algorithms. As many rational numbers have a repeating base-2 representation it means that when the pattern has a suitable length it is possible to use fewer additions compared to having a shorter wordlength. The results of this are illustrated in Fig. 2. This also provides a good example of that increasing the wordlength sometimes can decrease the addition complexity; the multiplication with 1/7 requires five additions when rounded to 23 fractional bits, but only three additions when rounded to 24 fractional bits.

#### B. Trigonometric Constants

Trigonometric constants occur in, e.g., FFTs, DCTs, and Goertzel filters [3], [4], [16]. Furthermore, it is notable that constants such as  $\sqrt{2}$  and  $\sqrt{3}$  are special cases of trigonometric constants. Here, we consider the best obtainable approximation using three additions. The results are shown in Table I for a number of different trigonometric constants found in the literature.

As can be seen the proposed methodology sometimes increases the precision for a given complexity. However, it is not always the case that coefficients exist with the same complexity but higher precision, as illustrated for some of the coefficients. With the proposed method this can be verified.

# C. Cordic Scale Factor Compensation

The CORDIC algorithm is a method to compute certain trigonometric and hyperbolic elementary functions based on

 TABLE II

 RESULTS FOR CORDIC GAIN COMPENSATION MULTIPLICATION

Coefficient	Additions	CFB	Value
$\frac{1}{G_{\infty}}$	2	8.130	$\frac{155}{256}$
	3	12.178	$\tfrac{9951}{16384}$
	4	18.802	$\tfrac{39797}{65536}$
	5	27.024	$\frac{326016439}{536870912}$
$rac{1}{\hat{G}_{\infty}}$	2	7.183	$\frac{155}{128}$
	3	10.506	$\frac{9889}{8192}$
	4	20.167	$\frac{158269}{131072}$
	5	24.000	$\frac{162067517}{134217728}$

rotating vectors. However, each rotation introduces a magnitude gain of the vector. This gain, after k iterations, is

$$G_k = \prod_{i=0}^k \sqrt{1 + 2^{-2i}}$$
(7)

for trigonometric operations and

$$\hat{G}_k = \prod_{i=1}^k \sqrt{1 - 2^{-2(i-h_i)}} \tag{8}$$

for hyperbolic operations<sup>1</sup>, cf. [17].

We consider compensation of the asymptotic gain factors, i.e., multiplication with  $1/G_k$  and  $1/\hat{G}_k$  when  $k \to \infty$ . The results are given in Table II and show the best possible approximations using a given number of additions. The results for five additions are given by the heuristic from [10], and, hence, those can not be guaranteed to be optimal. As a final note, it can be seen that  $1/\hat{G}_k$  is almost two times larger than  $1/G_k$ . Hence, the number of total correct bits is one more for  $1/\hat{G}_k$  for the same number of correct fractional bits.

### D. Joint Optimization of Several Factors

Sometimes a cascade of two or more constant multiplications are used. Then, the approximations of the individual multiplications are accumulated. While this may lead to cancellation of approximation errors having negative signs, it may also lead to that the total approximation error is larger than the individual approximation errors. The straightforward way of handling this is to increase the wordlengths of the individual multiplications until the total error meets the specification. Addition aware quantization provides a better way of obtaining this accuracy increase.

This is illustrated using a reconfigurable double constant multiplier for certain types of FFT algorithms as proposed in [18]. The multiplier structure is shown in Fig. 3 and it can multiply a single input with any of the coefficient pairs  $\{(1,0), (\sin \pi/8, \cos \pi/8), (\sin \pi/4, \cos \pi/4)\}$  using only constant multiplications with  $\sin \pi/8$  and  $\cos \pi/8$  by using that  $\sin \pi/4 = \cos \pi/4 = 2 \sin \pi/8 \cos \pi/8$ .

<sup>&</sup>lt;sup>1</sup>The factor  $h_i$  in (8) is defined as the largest integer such that  $3^{h_i+1} + 2h_i - 1 \le 2n$ . In practice this leads to that certain iteration angles, such that  $i = (3^{m+1} - 1)/2$ , are used twice to obtain convergence [17].

Fig. 3. Reconfigurable double constant multiplier proposed in [18].



Fig. 4. (a) Maximum approximation errors and (b) addition counts for the reconfigurable double constant multiplier is Fig. 3. Rounding (black), increasing fractional bits (gray), and addition aware quantization (white).

The approximation error for the  $\sin \pi/4$  multiplication is  $\epsilon_{\sin \pi/4} = 2\cos \pi/8\epsilon_{\sin \pi/8} + 2\sin \pi/8\epsilon_{\cos \pi/8}$ . Hence, it is possible that even though the multiplications with  $\sin \pi/8$  and  $\cos \pi/8$  are correct to N bits, the multiplication with  $\sin \pi/4$  is only correct<sup>2</sup> to N - 2 bits.

To reduce the approximation error to the required level we will use addition aware quantization. For each precision requirement we select the solution with smallest maximum approximation error among those solutions with the smallest addition count. For comparison we will also use a straightforward scheme based on increasing the number of fractional bits and rounding, as discussed above. The results in terms of approximation error is shown in Fig. 4(a), where it can be seen that in seven out of the 14 considered precisions, the rounded version actually breaks the precision requirements for the  $\sin \pi/4$  multiplication. The results in terms of required number of additions is shown in Fig. 4(b). Here, it can be seen that the proposed method in rare cases even decrease the number of additions. The reason that more additions are sometimes required is due to the fact that in these cases the rounded version do not meet the specification (compare to Fig. 4(a)). A benefit of the addition aware quantization scheme that is manifested in this example is the ability to select coefficient values such that the signs and magnitudes of the approximation errors cancel.

# IV. CONCLUSION

In this work we have proposed addition aware quantization as a way to find fixed-point coefficients suitable for shift-and-add

<sup>2</sup>  $\sin \pi/8 + \cos \pi/8 \approx 1.3065629648763 > 1.$ 

realization of the corresponding multiplication. By searching nearby coefficients it is often possible to find values that either have a smaller approximation error with the same addition count or, in some cases, a smaller addition count still meeting the error specification. Several examples illustrated the usefulness and the properties of the method.

#### ACKNOWLEDGMENT

The authors thank J. Thong and N. Nicolici (the authors of [10]) for kindly providing a copy of their algorithm implementation. They also thank the reviewers for their valuable comments on the manuscript.

#### REFERENCES

- J. Yli-Kaakinen and T. Saramäki, "A systematic algorithm for the design of lattice wave digital filters with short-coefficient wordlength," *IEEE Trans. Circuits Syst. I*, vol. 54, no. 8, pp. 1838–1851, Aug. 2007.
- [2] Y.-C. Lim, R. Yang, D. Li, and J. Song, "Signed power-of-two term allocation scheme for the design of digital filters," *IEEE Trans. Circuits Syst. II*, vol. 46, no. 5, pp. 577–584, May 1999.
- [3] J. Liang and T. D. Tran, "Fast multiplierless approximations of the DCT with the lifting scheme," *IEEE Trans. Signal Process.*, vol. 49, pp. 3032–3044, Dec. 2001.
- [4] S. C. Chan and P. M. Yiu, "An efficient multiplierless approximation of the fast fourier transform using sum-of-powers-of-two (SOPOT) coefficients," *IEEE Signal Process. Lett.*, vol. 9, pp. 322–325, Oct. 2002.
- [5] M. Püschel, A. C. Zelinski, and J. C. Hoe, "Custom-optimized multiplierless implementations of DSP algorithms," in *Proc. IEEE/ACM Int. Conf. Computer-Aided Design*, Nov. 2004, pp. 175–182.
- [6] M. Püschel et al., "SPIRAL: Code generation for DSP transforms," Proc. IEEE, vol. 93, pp. 232–275, Feb. 2005.
- [7] Y.-J. Yu and Y.-C. Lim, "Roundoff noise analysis of signals represented using signed power-of-two terms," *IEEE Trans. Signal Process.*, vol. 55, pp. 2122–2135, May 2007.
- [8] A. G. Dempster and M. D. Macleod, "Constant integer multiplication using minimum adders," *Proc. Inst. Elect. Eng., Circuits Devices Syst.*, vol. 141, no. 6, pp. 407–413, Oct. 1994.
- [9] O. Gustafsson, A. G. Dempster, K. Johansson, M. D. Macleod, and L. Wanhammar, "Simplified design of constant coefficient multipliers," *Circuits, Syst. Signal Process.*, vol. 25, no. 2, pp. 225–251, Apr. 2006.
- [10] J. Thong and N. Nicolici, "Time-efficient single constant multiplication based on overlapping digit patterns," *IEEE Trans. VLSI Syst.*, vol. 17, pp. 1353–1357, Sep. 2009.
- [11] A. G. Dempster and M. D. Macleod, "Generation of signed-digit representations for integer multiplication," *IEEE Signal Process. Lett.*, vol. 11, pp. 663–665, Aug. 2004.
- [12] R. G. E. Pinch, "Asymptotic upper bound for multiplier design," *Electron. Lett.*, vol. 32, no. 5, p. 420, Feb. 1996.
- [13] V. Dimitrov, L. Imbert, and A. Zakaluzny, "Multiplication by a constant is sublinear," in *Proc. 18th IEEE Symp. Comput. Arithmetic*, 2007, pp. 261–268.
- [14] O. Gustafsson, "Lower bounds for constant multiplication problems," *IEEE Trans. Circuits Syst. II*, vol. 54, no. 11, pp. 974–978, Nov. 2007.
- [15] O. Gustafsson and L. Wanhammar, "Low-complexity constant multiplication using carry-save arithmetic for high-speed digital filters," in *Proc. Int. Symp. Image, Signal Processing, Analysis*, Istanbul, Turkey, Sep. 27–29, 2007, pp. 212–217.
- [16] R. Beck, A. G. Dempter, and I. Kale, "Finite-precision Goertzel filters used for signal tone detection," *IEEE Trans. Circuits Syst. II*, vol. 48, no. 7, pp. 691–700, Jul. 2001.
- [17] J.-M Muller, *Elementary Functions: Algorithms and Implementation*, 2nd ed. Berlin, Germany: Birkhäuser, 2005.
- [18] J.-E. Oh and M.-S. Lim, "New radix-2 to the 4th power pipeline FFT processor," *IEICE Trans. Electron*, vol. E88-C, no. 8, pp. 1740–1764, Aug. 2005.