Combined IEEE Compliant and Truncated Floating Point Multipliers for Reduced Power Dissipation

Kent E. Wires
Agere Systems
Allentown, PA 18103

Michael J. Schulte
EECS Dept.
Lehigh University
Bethlehem, PA 18015

James E. Stine
ECE Dept.
Illinois Institute of Technology
Chicago, IL 60616

Abstract

Truncated multiplication can be used to significantly reduce power dissipation for applications that do not require correctly rounded results. This paper presents a power efficient method for designing floating point multipliers that can perform either correctly rounded IEEE compliant multiplication or truncated multiplication, based on an input control signal. Compared to conventional IEEE floating point multipliers, these multipliers require only a small amount of additional area and delay, yet provide a significant reduction in power dissipation for applications that do not require IEEE compliant results.

1 Introduction

Most modern processors perform floating point operations according to the IEEE 754 floating point standard [1]. This standard requires that the result of each floating point operation be identical to the result obtained by first computing the result to infinite precision and then rounding according to one of the four supported IEEE rounding modes. A significant portion of the total power for floating point multiplication is dissipated while computing the least significant bits of the product and rounding the result.

Many DSP and multimedia applications require a wide dynamic range, but do not require the directed rounding modes provided by the IEEE 754 floating point standard. Consequently, several multimedia instruction set extensions and digital signal processors support floating point operations that are not fully compliant with the IEEE 754 standard [2, 3, 4, 5, 6, 7].

Since many applications do not require correctly rounded multiplications, the total power dissipation for these applications can be reduced by utilizing truncated multiplication [8, 9, 10]. With truncated multiplication, the less significant columns of the multiplication matrix are eliminated and a correction constant is added to the more significant columns. Typically, however, the same processor needs to perform other applications that require IEEE compliant multiplication. By designing a single multiplier that can perform either truncated multiplication or IEEE compliant multiplication, a significant reduction in power dissipation can be achieved for applications that do not require IEEE compliant results.

This paper presents designs for floating point multipliers that perform either IEEE compliant or truncated multiplication, based on an input control signal. These multipliers are similar to designs presented in [8] and [10], however, modifications are made to allow both IEEE compliant and truncated multiplication to be performed. These multipliers require only slightly more area and delay than IEEE compliant multipliers, but dissipate significantly less power for applications that use truncated multiplication.

2 IEEE Compliant Multipliers

High performance floating point multipliers, such as those presented in [11, 12], often have a latency of two cycles and are pipelined for a throughput of one result per cycle. In the first cycle, the sign bits are exclusive-ored, the exponents are added and the bias is subtracted, the inputs are tested for special values, and the partial products for the significand multiplication are generated and reduced to sum and carry vectors. In the second cycle, the sum and carry vectors are added, the significand product is normalized and rounded, the exponent is adjusted to account for normalization, exceptional outputs are detected, and the final result is produced.

Guaranteeing that results are correctly rounded is the most difficult part of the floating point multiplier design. Using the method for correct IEEE rounding presented in [13], an injection term that depends on the rounding mode and the size of the multiplier is included as an additional partial product term in the significand multiplica-

tion matrix. A block diagram of the rounding scheme appears in Figure 1. With this method, the sum and carry vectors produced by the significand multiplier are each partitioned into two segments, one containing the most significant bits (MSBs) of the vectors, and the other containing the least significant bits (LSBs). The lower segments of the sum and carry vector are added together to produce a Carry bit (the MSB), Round bit (the second MSB), and Sticky bit (the logical OR of the remaining bits). Simultaneously, the higher segments are combined using a half-adder tree to produce Lx (the LSB), and Xs and Xc (the new sum and carry vectors). Xs and Xc are then passed to a compound adder, which computes two sums; $Xs + Xc$ and $Xs + Xc + 1$. The first sum has a carry-in of zero, which corresponds to rounding down, and the second sum has a carry-in of one, which corresponds to rounding up. While these sums are being computed, an increment decision is generated based on Carry, Round, Sticky, s_P, rm, and Lx. The increment decision determines which sum to use for the result and helps determine L, the LSB of the final result. Although this rounding method uses more hardware than other techniques, it has the advantage that only a single carry-propagate adder is on the critical delay path [13].

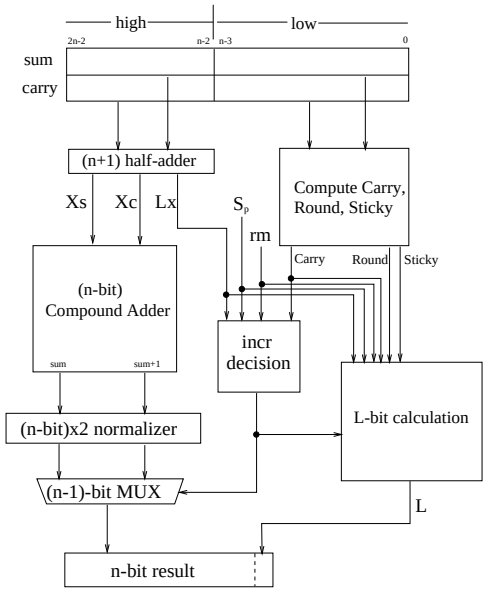

Figure 1. IEEE Rounding Logic

3 Truncated Multipliers

The method for truncated multiplication used in this paper is based on the constant correction truncated multiplication method presented in [8] for fixed-point numbers. Modifications are made to correctly handle floating point numbers and to ensure that results have a maximum error of less than one unit in the last place (ulp).

Figure 2 shows the significand multiplication matrix used by the proposed method for constant correction truncated floating point multiplication. With this method, the $n - k - 1$ least significant columns of the partial product matrix are completely eliminated, and a constant is added to columns $n - 3$ to $n - k - 1$ of the partial product matrix. This constant is determined by

$$R = \frac{round(2^{n+k+1} E_{total})}{2^{n+k+1}} \quad (1)$$

where E_{total} is given by

$$E_{total} = 0.5 \sum_{i=0}^{n-k-2} (i+1)2^{-2n+i} + 2^{-n-2}(1 - 2^{-k}) \quad (2)$$

The first term corresponds to half the maximum absolute error from eliminating columns 0 to $n - k - 2$, and the second term corresponds to half the maximum absolute error from not including p_{n-2} to p_{n-k-1} in the final result. Compared to the truncated multiplication method presented in [8], this method eliminates one less column in the partial product matrix and has a slightly modified correction constant. This is done to compensate for the fact that the result of the addition of the sum and carry vectors may or may not be normalized, and the required accuracy of the multiplier must be maintained in either case.

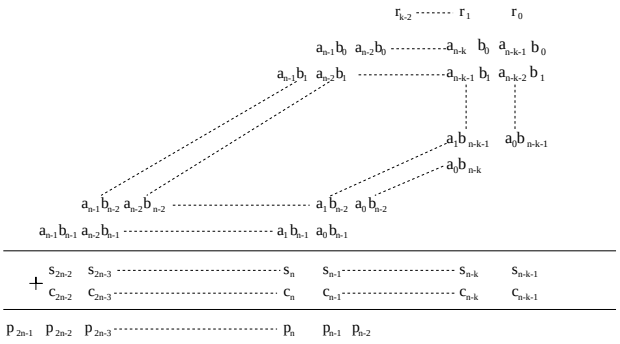

Figure 2. Truncated Multiplication Matrix

After the sum and carry vectors are added, the significand product is selected as $p_{2n-2}p_{2n-3}\ldots p_{n-1}$ when $p_{2n-1} = 0$, and $p_{2n-1}p_{2n-2}\ldots p_n$ when $p_{2n-1} = 1$. If $p_{2n-1} = 1$, a one is added to column $n - 2$ to compensate for not using p_{n-1} in the final product. This is achieved by inspecting bits p_{2n-1}, p_{n-1}, and p_{n-2}. If all three bits are one, a one is added to P at bit p_n. Otherwise, the result is truncated at bit p_n or p_{n-1}.

4 Combined IEEE Compliant & Truncated Multipliers

To combine the IEEE compliant and truncated multipliers, we introduce a control signal, t, where $t = 1$ for correctly rounded multiplication and $t = 0$ for truncated multiplication. The IEEE compliant multiplier and the truncated multiplier differ primarily in the significand multiplier and the rounding logic. Figure 3 shows the significand multiplication matrix for the combined multiplier, where $\hat{b}_i = b_i \cdot t$. Terms that include $\hat{b}_i$ correspond to the partial product bits of the eliminated columns of the truncated matrix. When $t = 0$, the hardware in those columns is effectively turned off, since all the partial product bits are zero, which reduces power dissipation. The logic that generates the injection value is modified so that R is injected when $t = 0$. With this approach, a standard significand multiplier is converted to a combined significand multiplier by adding $n - k$ AND gates to compute $\hat{b}_0$ to $\hat{b}_{n-k-1}$ and a small amount of logic to set the injection value to R when $t = 0$.

Error analysis based on Equation 2 and computer simulations verify that for single precision multipliers ($n = 24$), using $k = 5$ guarantees that the maximum error is less than 0.75 ulps. For this value of k, the least significant 18 columns of the multiplication matrix are completely eliminated, and the correction constant is $R = .11000_2 \times 2^{-25}$. For double precision multipliers ($n = 53$), using $k = 6$ completely eliminates the 46 least significant columns of the multiplication matrix and limits the maximum error to less than 0.84375 ulps. For this case, $R = .110110_2 \times 2^{-54}$.

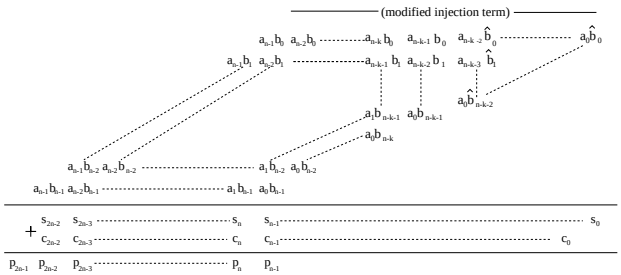

Figure 3. Combined Multiplication Matrix

The sum and carry vectors that are computed by the significand multiplier are next passed to the rounding stage. The rounding logic of the IEEE compliant multiplier is modified slightly to accommodate truncated calculations for the combined multiplier. A block diagram of the rounding logic of the combined multiplier is shown in Figure 4. The rounding logic is similar to that presented in [13], with some additional logic to handle the case of truncated multiplication.

When truncated multiplication is performed, the result is formed by adding the $n + k$ most significant bits of the sum and carry vectors, and truncating the result to n bits. With the existing IEEE compliant rounding logic the $n+1$ MSBs (columns $n - 2$ to $2n - 2$) of the sum and carry vectors are added together. To also perform truncated multiplication, an extra $(k - 1)$-bit adder is use to add columns $n - k - 1$ to $n - 3$, and determine if there is carry into column $n - 2$. This carry is denoted as C_K. An additional one is added to columnn $n - 2$ if there is carry out of the MSB of the sum output of the compound adder. Since this corresponds to the case where p_{2n-1} is one, normalization is necessary, and R is too small. If this carry out is denoted as OVF, the increment decision is computed as

$$incr = OVF \cdot C_K + OVF \cdot Lx + C_K \cdot Lx \qquad (3)$$

When truncated multiplication is performed, the least significant bit of the product, L, is computed as

$$L = Nsum_L \cdot \overline{incr} + Nsump1_L \cdot incr \qquad (4)$$

where $Nsum_L$ and $Nsump1_L$ are the least significant bits of the normalized sum and $sum + 1$ outputs of the compound adder, respectively.

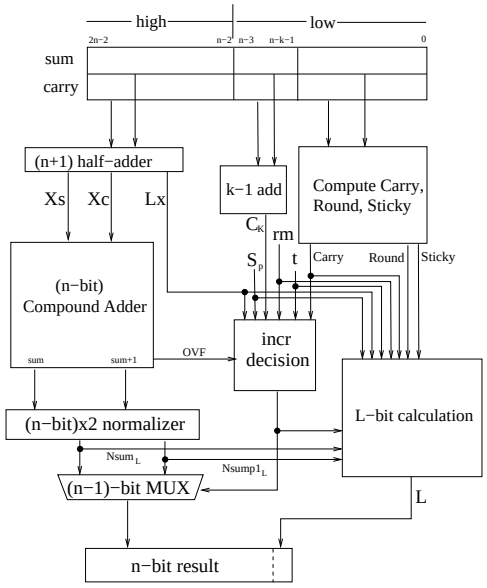

Figure 4. Combined Rounding Logic

5 Area, Delay, and Power Comparison

Single precision and double precision IEEE compliant, truncated, and combined floating point multipliers were implemented in Verilog and synthesized using the Synopsys Module Compiler and a 2.5 Volt, 0.25 micron CMOS standard cell library. Average power dissipations were estimated using PowerMill and pseudo-random input data

vectors. The partial product matrices were reduced using Dadda's reduction method [14], and carry lookahead adders were used to implement the compound adder. The area, critical path delay, and power dissipation of each of these units are found in Table 1.

Unit	Area (mm^2)	Delay (ns)	Power (mW)
Single Precision			
IEEE	0.2985	4.00	93.83
trunc	0.2217	3.50	70.31
comb	0.3005	4.10	72.07 ($t = 0$) 94.16 ($t = 1$)
Double Precision			
IEEE	1.378	5.61	390.23
trunc	0.845	4.63	259.40
comb	1.390	5.74	253.51 ($t = 0$) 394.96 ($t = 1$)

Table 1. Area, Delay, and Power Estimates

Compared to the IEEE single precision multiplier, the combined single precision multiplier has an area that is just 0.7% larger and a worst-case delay that is 2.5% longer. The combined unit dissipates 23% less power than the single precision IEEE multiplier when $t = 0$, and dissipates 1.2% more power when $t = 1$. The truncated multiplier has an area that is 26% less than the IEEE multiplier and a worst-case delay that is 13% shorter than that of the IEEE multiplier. It dissipates roughly 25% less power than the IEEE single precision multiplier for the same set of input vectors.

The combined double precision multiplier has an area that is 0.9% larger and a worst-case delay that is 2.3% longer than that of the IEEE double precision multiplier. The combined unit dissipates 35% less power than the double precision IEEE multiplier when $t = 0$, and dissipates 1.2% more power when $t = 1$. The truncated multiplier has an area that is 39% less than the IEEE double precision multiplier and a worst-case delay that is 17% shorter than that of the IEEE multiplier. It dissipates roughly 34% less power than the IEEE multiplier for the same set of input vectors.

6 Conclusions

Combined IEEE compliant and truncated multipliers are useful in computer systems that support digital signal processing and graphics applications in which strict adherence to the IEEE 754 floating point standard is often not necessary. The use of IEEE compliant or truncated multiplication can be controlled by a mode bit in the floating point status and control register. When the mode bit indicates truncated multiplication, the partial product bits in the least significant columns of the multiplication matrix are set to zero, which effectively turns off the hardware in these columns and prevents transitions. Compared to IEEE compliant multipliers, the combined multipliers have only a small increase in area and delay, yet provide a significant savings in power consumption when truncated multiplication is performed.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant No. CCR-9703421. This research is also supported by a grant from Agere Systems and the Pennsylvania Infrastructure Technology Alliance under Project No. IST-001.

References

[1] *IEEE Standard 754 for Binary Floating Point Arithmetic.* IEEE, 1985.

[2] R. Weiss, "32-Bit Floating-Point DSP Processors," *EDN*, vol. 36, no. 23, pp. 127–146, 1991.

[3] J. R. Boddie *et al.*, "A 32-Bit Floating-Point DSP with C Compiler," in *Conference Record of the Twenty-Second Asilomar Conference on Signals, Systems and Computers*, vol. 2, pp. 880–884, 1989.

[4] K. Diefendorff, P. Dubey, R. Hochsprung, and H. Scale, "AltiVec Extension to PowerPC Accelerates Media Processing," *IEEE Micro*, vol. 20, no. 2, pp. 85–95, 2000.

[5] S. Thakkur and T. Huff, "Internet Streaming SIMD Extensions," *Computer*, vol. 32, no. 12, pp. 26–34, 1999.

[6] S. Oberman, F. Weber, N. Juffa, and G. Favor, "AMD 3DNow! Technology and the K6-2 Microprocessor," in *Proceedings of Hot Chips 10*, pp. 245–254, 1998.

[7] N. Vasseghi, K. Yeager, E. Sarto, and M. Seddighnezhad, "A 200-MHz Superscalar RISC Microprocessor," *IEEE Journal of Solid-State Circuits*, vol. 31, no. 11, pp. 1675–1686, 1996.

[8] M. J. Schulte and E. E. Swartzlander, Jr., "Truncated Multiplication with Correction Constant," in *VLSI Signal Processing, VI*, pp. 388–396, 1993.

[9] M. J. Schulte, J. G. Jansen, and J. E. Stine., "Reduced Power Dissipation Through Truncated Multiplication," in *IEEE Alessandro Volta Memorial International Workshop on Low Power Design*, pp. 61–69, 1999.

[10] K. E. Wires, M. J. Schulte, and J. E. Stine, "Variable-Correction Truncated Floating Point Multipliers," in *Proceedings of the Thirty Fourth Asilomar Conference on Signals, Circuits and Systems*, pp. 1344–1348, 2000.

[11] H. Yamada *et al.*, "13.3 ns Double-Precision Floating-Point Multiplier and ALU," in *Proceedings of the International Conference on Computer Design*, pp. 466–470, 1995.

[12] W.-C. Park, T.-D. Han, S.-D. Kim, and S.-B. Yang, "A Floating Point Multiplier Performing IEEE Rounding and Addition in Parallel," *Journal of Systems Architecture*, vol. 45, no. 14, pp. 1195–1207, 1999.

[13] G. Even and P.-M. Seidel, "A Comparison of Three Rounding Algorithms for IEEE Floating-Point Multiplication," in *Proceedings of the 13th Symposium on Computer Arithmetic*, pp. 225–232, 1997.

[14] L. Dadda, "Some Schemes for Parallel Multipliers," *Alta Frequenza*, vol. 34, pp. 349–356, 1965.