Asian Journal of Information Technology 15 (19): 3852-3857, 2016
ISSN: 1682-3915

Design of a Multi Precision Floating Point Multiplier Using a Hybrid Technique

[1]D. Gokila and [2]H. Mangalam
[1]United Institute of Technology, 641020 Coimbatore, India
[2]Sriramakrishna Institute of Technology, 641008 Coimbatore, India

Abstract: Floating-Point (FP) multipliers are the key energy consumers in most of the present embedded processors based on digital signals and multimedia based appliances. A most common approach for reducing the energy utilization of FP multipliers is by cutting down the accuracy of FP multiplication operations under acceptable precision loss. This study proposes a multiple-precision FP multiplier to ably trade the energy utilization with the output quality. The proposed FP multiplier can perform low precision multiplication that generates 8, 14, 20 or 26 bit mantissa product through a hybrid technique that effectively fuses the row suppression and column suppression methodologies. Radix-8 Booth algorithm is used as the row suppression technique and truncation methodology is used for column suppression technique to tailor the output. Energy saving for this low precision multiplication is achieved by partly suppressing the computation of mantissa multiplier. In addition, the proposed multiplier allows the mantissa's bit width of the output product to change dynamically when it performs different FP multiplication operations to further decrease the energy consumption.

Key words: FP multiplier, modified booth, truncated, partial product, bit width

INTRODUCTION

Multiplication is the nucleus of many algorithms used in scientific computations such as Digital Signal Processing (DSP). Over the years the computational complexities of algorithms used in Digital Signal Processors (DSPs) have progressively increased. Therefore, it requires fast and efficient parallel multipliers for general purpose as well as application specific architectures. Recently, Field Programmable Gate Arrays (FPGAs) have emerged as a platform of choice for efficient hardware implementation of computation intensive algorithms. FPGA have the benefit of hardware speed and the flexibility of software. In general, FP arithmetic units can support a wide dynamic range and high computation precision for real numbers. However, FP arithmetic units occupy the major portion of a processor's area and energy consumption. In addition, FP operations are usually the performance bottleneck in these applications. Consequently, it is essential to develop FP arithmetic units with flexible processing ability, high performance, small area and low energy consumption to compose the portable device with the required features.

Related researches: Several low-power integer multiplier designs have been proposed. An approach to reach low power design involves Minimizing Floating-Point Power Dissipation via Bit-Width Reduction (Cho *et al*., 2004). Analysis of several floating point programs that utilize low-resolution sensory data shows that the programs suffer almost no loss of accuracy even with a significant reduction in bit-width (Tong *et al*., 2000). This floating point bit-width reduction can deliver a significant power saving through the use of a variable bit-width floating point unit. However, this design is not compliant with IEEE 754 standard; results are not precise and have significant error. Similar to previous approach, strudy (Schulte and Swartzlander, 1993) explores ways of reducing floating-point power consumption by minimizing the bit-width representation of floating-point data. Analysis of several floating-point programs that manipulate low-resolution human sensory data shows that these programs suffer no loss of accuracy even with a significant reduction in bit width but produces only one truncated output. The work in (Wires *et al*., 2001) presented a FP multiplier that allows IEEE compliant or truncated multiplication (Lim, 1992; Schulte and Swartzlander, 1993; Cho *et al*., 2004) to be performed according to an input control signal. Study (Visalli and Pappalardo, 2003) is using similar variable-width floating-point unit with specific application in wireless communication and image manipulation. As stated previously such approach is not compliant with IEEE standards and requires modification of existing applications to be suitable for use in such format. One of the multiplier circuit is based on the modified Booth algorithm and the pipeline technique which are most widely used to accelerate the multiplication speed with

Corresponding Author: D. Gokila, United Institute of Technology, 641020 Coimbatore, India

area penalty. Another series of study (Kuang *et al*., 2009, 2003; Wang *et al*., 2008; Yeh and Jen, 2000) proposed lightweight floating-point system design flow of which the bit-width optimization engine is a core component. It takes both the hardware cost and the numerical performance into account and finds the optimal bit-width configuration. Variable grouping enables the designer to give some hardware topology information to the optimizer so that the configuration result is easier to be mapped onto hardware design. In power minimization of functional units by partially guarded computation (Choi *et al*., 2000), the researchers presented a partially guarded computation technique to suppress the signal transitions in an array multiplier based on the dynamic range of the input operands. A two-dimensional signal gating technique (Huang and Ercegovac, 2005) was employed in to design a low-power array multiplier. Other works (Chowdhury *et al*., 2008) adopted power gating technique to achieve power saving when lower precision multiplication is performed. Recently, many reconfigurable integer multipliers were constructed by the sub-word partitioning technique (Kim and Cho, 2010) to offer one n×n, two n/2×n/2, or four n/4×n/4 multiplication operations and the bit-widths of their output products are 2n, n and n/2, respectively. To achieve high flexibility and regularity, these reconfigurable integer multipliers (Wen *et al*., 2005) are frequently designed based on array multipliers. Another approach (Tan *et al*., 2009) for designing reconfigurable integer multipliers was modified to construct an area and power efficient multi-precision iterative FP multiplier. An iterative FP multiplieris compliant with IEEE-754 standard and cannot reduce energy consumption for applications that don't require IEEE compliant results. In (Kuang *et al*., 2009) an energy efficient FP multiplier is designed with an MBE multiplier with a truncation technique to produce multi-precision output.

This proposed work 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 with only slight area and delay overheads to generate a family of low-precision multiplications.

MATERIALS AND METHODS

Proposed work

Multiple precision: A lot of DSP and multimedia applications which requires the wide dynamic range of FP arithmetic, don't need the accurate rounding modes offered by the IEEE-754 FP multiplication. Many applications can manipulate a slight loss of accuracy even when the bit width of mantissa is largely reduced. For such applications, the energy consumption of FP multiplier can be traded with the output accuracy. Many FP applications allow a slight output distortion, thereby trade output quality with energy consumption via reducing the precision of FP multiplication operations to be less accurate than IEEE single-precision FP multiplication.

Table 1: Precision modes

Configuration mode	PM[2:0]	Function description	Latency (cycles)
0	000	Produce 26-bit mantissa product	3
1	001	Produce 26-bit mantissa product	2
2	010	Produce 20-bit mantissa product	2
3	011	Produce 14-bit mantissa product	2
4	100	Produce 8-bit mantissa product	2

A sort of multi-mode FP recursive booth multiplier which can provide Multiple Precision Modes (PMs) is proposed. Here, the Booth Multiplier is used to generate the partial products and the truncation technique is followed to produce multimode outputs. However, the maximum error of each PM with respect to IEEE single-precision FP multiplication is very difficult to compute by using exhaustive simulation. To efficiently assign each multiplication operation in an application to a proper PM for satisfying output error constraint and achieving more energy saving, an exact analysis method was proposed to estimate the maximum error of each PM (Thomas, 2014). Different applications frequently require different FP precisions. Even different FP multiplication operations in the same application can be performed with different precisions to achieve more energy saving while maintaining the accuracy requirement. Hence, designing one FP multiplier (Wu *et al*., 2013) that can perform different low-precision multiplications (less precision than the single-precision IEEE-754 compliant multiplication) is imperative to fit different applications and trade the energy consumption with the output accuracy.

The FP multiplier is designed with five precision modes. The input control signal PM[2:0] is utilized to indicate the precision modes (Fig. 1 and Table 1).

Radix-8 booth encoder multiplier: The inputs of the multiplier are multiplicand X and multiplier Y. Before designing Radix-8 Booth Encoder, the multiplier has to be converted into a Radix-8 number by dividing them into four digits respectively according to Booth Encoder Table 2. Prior to convert the multiplier, a zero is appended into the Least Significant Bit (LSB) of the multiplier.

Partial product generator: A product formed by multiplying the multiplicand by one digit of the multiplier

Table 2: Radix-8 booth recoding

Quartet value	Signed digit value
0000	0
0001	+1
0010	+1
0011	+2
0100	+2
0101	+3
0110	+3
0111	+4
1000	-4
1001	-3
1010	-3
1011	-2
1100	-2
1101	-1
1110	-1
1111	0

when the multiplier has more than one digit. Partial products are used as intermediate steps in calculating larger products. Partial product generator is designed to produce the product by multiplying the multiplicand A by 0, 1, -1, 2, -2, 3, -3, 4,-4. For product generator:

- Multiply by zero means the multiplicand is multiplied by "0"
- Multiply by "1" means the product still remains the same as the multiplicand value
- Multiply by "-1" means that the product is the two's complement form of the number
- Multiply by "-2" is to shift left one bit the two's complement of the multiplicand value
- Multiply by "2" means just shift left the multiplicand by one place
- Multiply by "-4" is to shift left two bit the two's complement of the multiplicand value and multiply by "4" means just shift left the multiplicand by two places

Here an odd multiple of the multiplicand, 3Y which is not immediately available, is performed. To generate it is required to perform this previous add: 2Y+Y=3Y. In this manner, overall multiplication time can be improved compared to Radix-4 architecture.

The objective is to design multiple precision floating point multiplier using Radix-8 Booth Algorithm. Booth's algorithm involves repeatedly adding one of two predetermined values to a product P and then performing a rightward arithmetic shift on P.Radix-8 Booth encoding generates n/3 partial products in parallel.

The conventional Modified Booth Encoding (MBE) generates an irregular partial product array because of the extra partial product bit at the least significant bit position of each partial product row. A simple approach to generate a regular partial product array with fewer partial product rows and negligible overhead is necessary,

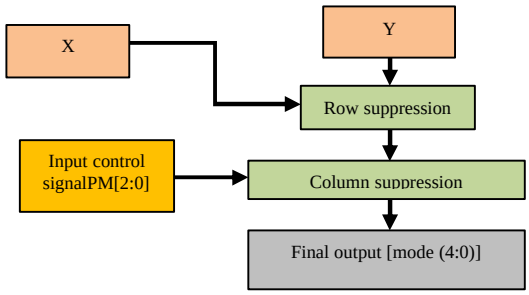

Fig 1: Flowchart of FP multiplier

thereby lowering the complexity of partial product reduction and reducing the area, delay and power of MBE multipliers. Radix-8 Booth encoding is most often used to avoid variable size partial product arrays.

Architecture: The architecture of the multiple precision FP multiplier is given below. The multiple-precision recursive multiplier needs extra iterations to achieve higher precision, leading to longer latency and higher energy consumption. To avoid the long latency, divide the partial product rows are divided into only two parts: the Lower Part (LP) and the Upper Part (UP) (Fig. 2).

Truncation: Truncated multiplier suppresses the computation of some less significant columns of partial products. Therefore, the truncated multiplier will introduce less error than the recursive multiplier when both of them omit the same amount of partial product bits. In this subsection, the truncation multiplication is employed to increase the number of precision modes to five at the expense of extra area overhead. To achieve the goal, the partial products are further partitioned into six parts with five cut lines labelled MODE0 to MODE4.

For Mode 0-4, the proposed multiplier performs low-precision multiplication and produces 26-, 26-, 20-, 14 or 8-bit mantissa product, respectively. Mode 0 performs the FP multiplication with a latency of three cycles and the other modes perform the FP multiplication with a latency of two cycles.

Figure 3 shows the arrangement of partial products in the FP multiplier and various modes are selected which provides different results with selected accuracy.

Mode 0: When Mode 0 is performed, the recursive multiplier executes two iterations, but the partial product bits on the right side of direction line in Fig. 3 will not be generated by PPGs and are omitted from the partial product addition. The rest partial product bits on the left side are summed to produce a 26-bit mantissa product

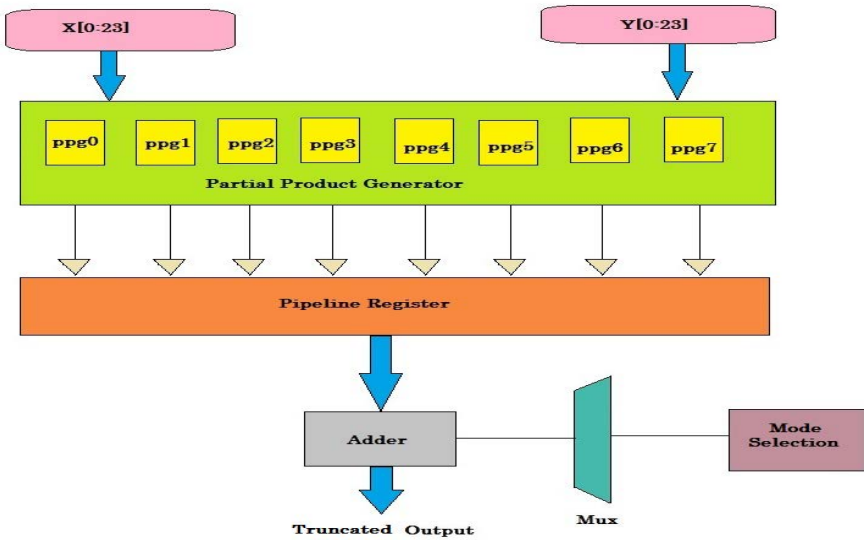

Fig. 2: Architecture of FP multiplier

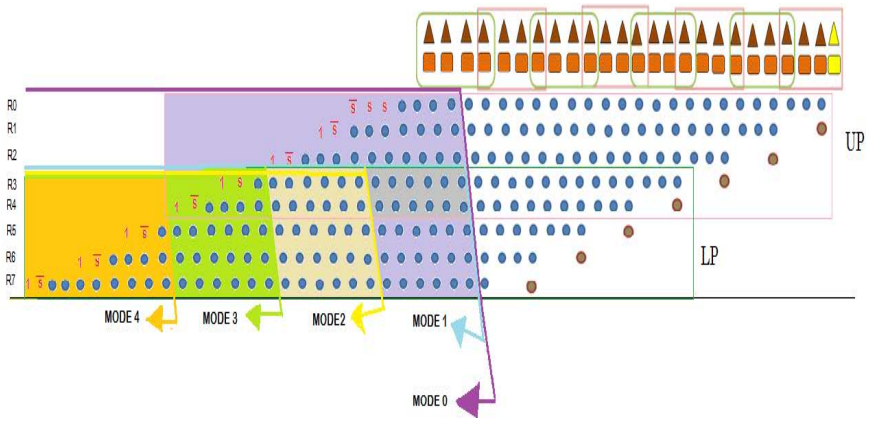

Fig. 3: Modes and results

S0[47:22] and C0[47:22] which are then stored into RS[47:22] and RC[47:22].Subsequently, these intermediate values are summed and normalized in the second pipeline stage to obtain the final FP product.

Mode 1: If Mode 1 is selected, the multiplier only executes once and all the partial product bits on the right and upper sides of cut line in Fig. 3 are omitted from partial product addition. The rest partial product bits are summed to produce a 26-bit mantissa product S1[47:22] and C1[47:22] and the final FP product. Since the multiplier only executes once and more partial product bits are omitted, more energy saving and shorter latency are achieved at the expense of larger error.

Mode 2: The multiplier also executes once and omits more partial product bits than Mode 1, leading to shorter mantissa products S2[47:28], C2[47:28].

Mode 3: The multiplier executes once and omits more partial product bits than Mode 1, leading to shorter mantissa products S3[47:34], C3[47:34].

Mode 4: The multiplier also executes once and omits more partial product bits thanMode 1, leading to shorter mantissa products S4[47:40], C4[47:40].

RESULTS AND DISCUSSION

The multiplier unit design for the Radix-8 Booth Encoder Multiplier is designed in VHDL, Simulated in Xilinx 8.1. VHDL code is written to generate the required hardware and to produce the partial product. The design and simulation of a 24x24 bit, radix-8 multiplier unit has been performed. In all multiplication operation product is obtained by adding the partial products. Thus the final speed of the multiplier circuit depends on the speed of the

Table 3:Numerical data

Type	Power (mW)	Delay (ns)	Gate count	Energy (pJ)
Radix 4				
Mode 0	68	29.41	10,029	1999.88
Mode 1	59	28.06	7,329	1655.54
Mode 2	56	26.93	5,402	1508.08
Mode 3	49	25.55	3,250	1249.5
Mode 4	43	22.69	1,340	975.67
Radix 8				
Mode 0	41	25.33	6,395	1038.53
Mode 1	38	24.57	3,941	933.66
Mode 2	35	22.72	2,921	795.2
Mode 3	32	21.07	2,291	54.499
Mode 4	28	19.57	13	547.96

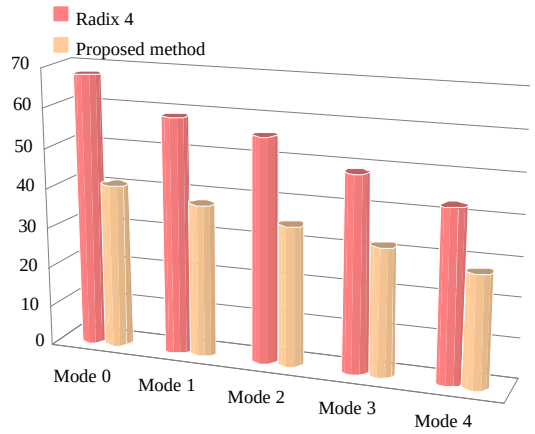

Fig. 4: Power results

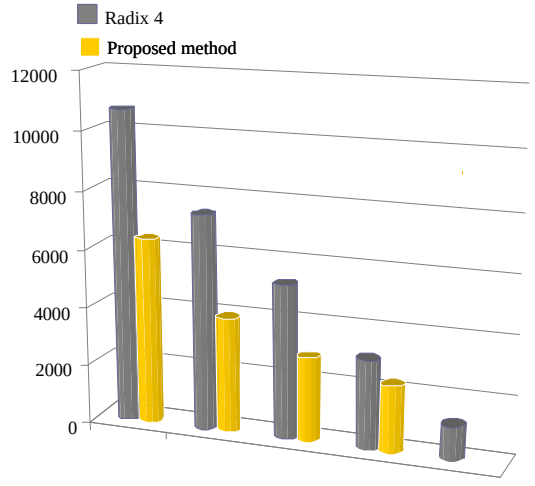

Fig. 5: Area comparison

adder circuit and the number of partial products generated. The results for the existing radix-4 multiplier are compared with the proposed radix-8 multiplier with the same specifications.

The results of the radix 4 and the proposed radix 8 are presented which shows that the proposed radix 8 is better than that of the radix 4. The numeric data is displayed in the Table 3.

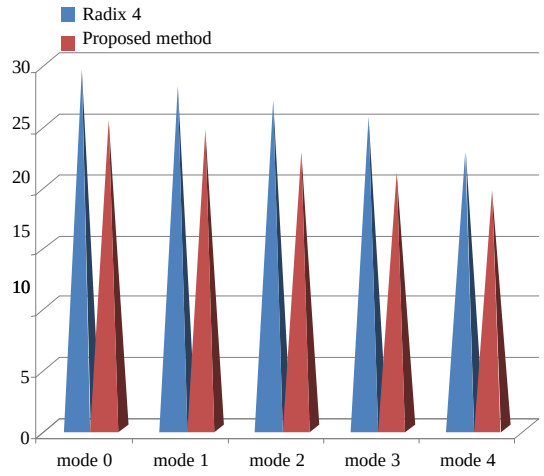

Fig. 6: Results for delay

From the Fig. 4, it is observed that the power consumption for the lower precision modes has reduced comparatively and the proposed method results in an power improvement of 40% in mode 0 and 36, 37.5, 35 and 35% for modes 1, 2, 3 and 4 respectively which results in an average of 36.7% improvement in power metric (Fig. 5).

From the results for all the modes, it was observed that the gate count has reduces drastically as precision decreases with a maximum of 99% in mode 4 which is the lowest precision. The average gate count reduction is upto 51.2% (Fig. 6).

It is observed that an average delay of 14.5% decrease is seen with respect to the radix 4 multiplier. From the Area, Delay and Power results, it is observed that the proposed radix-8 Booth multiplier has achieved a drastic reduction compared to the Radix-4 multiplier with the energy consumption reduced under an acceptable accuracy loss.

CONCLUSION

In this study, an energy-efficient multiple-precision FP multiplier has been proposed using the Radix-8 Booth encoding and truncation technique. The proposed FP multiplier can perform low-precision multiplication that generates 8-, 14-, 20-, or 26-bit mantissa product through recursive and truncation multiplications.

It has been proved that it can be useful to apply a radix-8 architecture in high-speed multipliers for specific purpose because of the gain in time and number of transistors compared to the conventional radix-4 recoding architecture.

REFERENCES

Cho, K.J., K.C. Lee, J.G. Chung and K.K. Parhi, 2004. Design of low-error fixed-width modified booth multiplier. IEEE. Trans. Very Large Scale Integr. VLSI. Syst., 12: 522-531.

Choi, J., J. Jeon and K. Choi, 2000. Power minimization of functional units partially guarded computation. Proceedings of the 2000 International Symposium on Low Power Electronics and Design, July 25-27, 2000, ACM, Rapallo, Italy, ISBN:1-58113-190-9, pp: 131-136.

Chowdhury, M.H., J. Gjanci and P. Khaled, 2008. Innovative power gating for leakage reduction. Proceedingd of the 2008 IEEE International Symposium on Circuits and Systems, May 18-21, 2008, IEEE, Chicago, USA, ISBN:978-1-4244-1683-7, pp: 1568-1571.

Huang, Z. and M.D. Ercegovac, 2005. High-performance low-power left-to-right array multiplier design. IEEE Trans. Comput., 54: 272-283.

Kim, S. and K. Cho, 2010. Design of high-speed modified booth multipliers operating at GHz ranges. World Acad. Sci. Eng. Technol., 37: 1-4.

Kuang, S.R., J.P. Wang and C.Y. Guo, 2009. Modified booth multipliers with a regular partial product array. IEEE. Trans. Circuits Syst. Express Briefs, 56: 404-408.

Kuang, S.R., K.Y. Wu and K.K. Yu, 2013. Energy-efficient multiple-precision floating-point multiplier for embedded applications. J. Signal Process. Syst., 72: 43-55.

Lim, Y.C., 1992. Single-precision multiplier with reduced circuit complexity for signal processing applications. IEEE Trans. Comput., 41: 1333-1336.

Schulte, M.J. and E.E. Swartzlander, 1993. Truncated multiplication with correction constant [for DSP]. Proceedings of the Workshop on VLSI Signal Processing, October 20-22, 1993, IEEE, Austin, Texas, USA, ISBN:0-7803-0996-0, pp: 388-396.

Stevenson, D., 1981. A proposed standard for binary floating-point arithmetic. Comput., 14: 51-62.

Tan, D., C.E. Lemonds and M.J. Schulte, 2009. Low-power multiple-precision iterative floating-point multiplier with SIMD support. IEEE. Trans. Comput., 58: 175-187.

Thomas, M., 2014. Design and simulation of radix-8 booth encoder multiplier for signed and unsigned numbers. Int. J. Innov. Res. Sci. Technol., 1: 1-10.

Tong, J.Y.F., D. Nagle and R.A. Rutenbar, 2000. Reducing power by optimizing the necessary precision/range of floating-point arithmetic. IEEE. Trans. Very Large Scale Integr. VLSI. Syst., 8: 273-286.

Visalli, G. and F. Pappalardo, 2003. Low-power floating-point encoding for signal processing applications. Proceedings of the IEEE Workshop on Signal Processing Systems SIPS-2003, August 27-29, 2003, IEEE, Catania, Italy, ISBN:0-7803-7795-8, pp: 292-297.

Wang, L.R., S.J. Jou and C.L. Lee, 2008. A well-structured modified Booth multiplier design. Proceedings of the IEEE International Symposium on VLSI Design, Automation and Test VLSI-DAT-2008, April 23-25, 2008, IEEE, Taiwan, China, ISBN:978-1-4244-1616-5, pp: 85-88.

Wen, M.C., S.J. Wang and Y.N. Lin, 2005. Low power parallel multiplier with column bypassing. Proceedings of the IEEE International Symposium on Circuits and Systems, Volume 2, May 23-26, 2005, China, pp: 1638-1641.

Wires, K.E., M.J. Schulte and J.E. Stine, 2001. Combined IEEE compliant and truncated floating point multipliers for reduced power dissipation. Proceedings of the 2001 International Conference on Computer Design ICCD-2001, September 23-26, 2001, IEEE, Allentown, Pennsylvania, USA, ISBN:0-7695-1200-3, pp: 497-500.

Wu, K.Y., S.R. Kuang and K.K. Yu, 2013. An exact method for estimating maximum errors of multi-mode floating-point iterative booth multiplier. Int. J. Comput. Sci. Eng., 8: 306-315.

Yeh, W.C. and C.W. Jen, 2000. High-speed booth encoded parallel multiplier design. IEEE Trans. Comput., 49: 692-701.