ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

Design of Energy Efficient Multiplier for DSP Applications

P Narayana
PG Scholar,
AITS, Kadapa,
Andhra Pradesh, India.

O. Homa Kesav
Assistant Professor,
AITS, Kadapa,
Andhra Pradesh, India.

Dr. G. K. Rajini
Associate Professor,
VIT University,
Vellore.

Abstract:
The need to support various digital signal processing (DSP) and classification applications on energy-constrained devices has steadily grown. Such applications often extensively perform matrix multiplications using fixed-point arithmetic while exhibiting tolerance for some computational errors. Hence, improving the energy efficiency of multiplications is critical. In this brief, we propose multiplier architectures that can tradeoff computational accuracy with energy consumption at design time. Compared with a precise multiplier, the proposed multiplier can consume 58% less energy/op with average computational error of ~1%. Finally, we demonstrate that such a small computational error does not notably impact the quality of DSP and the accuracy of classification applications.

Index Terms:
Approximation, energy efficiency, multiplication.

I.INTRODUCTION:

Achieving high energy efficiency has become a key design objective for embedded and mobile computing devices due to their limited battery capacity and power budget. To improve energy efficiency of such computing devices, significant efforts have already been devoted at various levels, from software to architecture, and all the way down to circuit and technology levels. Embedded and mobile computing devices are frequently required to execute some key digital signal processing (DSP) and classification applications. To further improve energy efficiency of executing such applications, first, dedicated specialized processors are often integrated in computing devices.

It has been reported that the use of such specialized processors can improve energy efficiency by 10–100× compared with general-purpose processors at the same voltage and technology generation. Second, many DSP and classification applications heavily rely on complex probabilistic mathematical models and are designed to process information that typically contains noise. Thus, for Some computational error, they exhibit graceful degradation in overall DSP quality and classification accuracy instead of a catastrophic failure. Such computational error tolerance has been exploited by trading accuracy with energy consumption. Finally, these algorithms are initially designed and trained with floating-point (FP) arithmetic, but they are often converted to fixed point arithmetic due to the area and power cost of supporting FP units in embedded computing devices Although this conversion process leads to some loss of computational accuracy, it does not notably affect the quality of DSP and the accuracy of classification applications due to computational error tolerance.

Most of such algorithms extensively perform matrix multiplications as their fundamental operation, while a multiplier is typically an inherently energy-hungry component. To improve energy efficiency of multipliers, previous studies have explored various techniques exploiting computational error tolerance. They can be classified into three categories: 1) aggressive voltage scaling [4], [5]; 2) truncation of bit-width and 3) use of inaccurate building blocks. Chippa proposed scalable effort hardware design and explored algorithm-, architecture, and circuit-level scaling to minimize energy consumption while offering acceptable classification quality through aggressively scaling voltage and truncating least significant bits.

Kulkarni proposed an under-designed 16×16 multiplier using inaccurate 2 × 2 partial product generators (PPG) while guaranteeing the minimum and maximum accuracy fixed at design time. Each PPG has fewer transistors compared with the accurate 2×2 one, reducing both dynamic and leakage energy at the cost of some accuracy loss. Babi´ca proposed a novel iterative log approximate multiplier using leading one detectors (LODs) to support variable accuracy. In this brief, we propose an approximate multiplication technique that takes m consecutive bits (i.e., m-bit segment) from each n-bitoper and, where m is equal to or greater than n/2. An m-bit segment can start only from one of two or three fixed bit positions depending on where the leading one bit is located for a positive number. This approach can provide much higher accuracy than one simply truncating the LSBs, because it can more effectively capture more noteworthy bits.

Although we can capture m-bit segments starting from the exact leading one bit position, such an approach requires expensive LODs and shifters to take m-bit segments starting from the leading one position, steer them to an m × m multiplier, and expand 2m bits to 2n bits. In contrast, our approach is more scalable than one that captures m-bit segments starting from the leading one bits since it limits the possible starting bit positions of an m-bit segment to two or three regardless of m and n chosen at design time, eliminating LODs, and replacing shifters with multiplexers. Finally, we also observe that one of two operands in each multiplication for DSP and classification algorithms is often stored in memory (e.g., coefficients in filter algorithms and trained weight values in artificial neural networks) and repeatedly used. We exploit it to further improve the energy efficiency of our approximate multiplier. The rest of this brief is organized as follows. Section II details the proposed multiplier architecture. Section III analyzes energy consumption and computational accuracy of various approximate multipliers and impact of such multipliers on quality of DSP and accuracy of classification algorithms. Finally, Section IV conclude this brief.

II.APPROXIMATEMULTIPLIERARCHITECTURE:

In order to motivate and describe our proposed multiplier, we define an m-bit segment as m contiguous bits starting with the leading one in an n-bit positive operand. We dub this method dynamic segment method (DSM) in contrast to static segment method (SSM) that will be discussed later in this section. With two m-bit segments from two n-bit operands, we can perform a multiplication using an m× m multiplier. Fig. 1 shows an example of a multiplication after taking 8-b segments from 16-b operands. In this example, we can achieve 99.4% accuracy for a 16 × 16 multiplication even with an 8 × 8 multiplier. Such a multiplication approach has little negative impact on computational accuracy because it can eliminates redundant bits (i.e., sign-extension bits) while feeding the most useful m significant bits to the multiplier; we will provide detailed evaluations of computational accuracy for various m in Section III.

Furthermore, an m×mmultiplier consumes much less energy than an n×nmultiplier, because the complexity (and thus energy consumption) of multipliers quadratically increases with n. For example, the 4 × 4 and 8 × 8 multipliers consume almost 20 × and 5 × less energy than a 16×16 multiplier per operation on average. However, a DSM requires: 1) two LODs; 2) two n-bit shifters to align the leading one position of each n-bit operand to the MSB position of each m-bit segment to apply their m-bit segments to the m×mmultiplier; and 3) one 2n-bit shifter to expand a 2m-bit result to 2n bits. 1)–3) incur considerable area and energy penalties completely negating the energy benefit of using the m× m multiplier; we provide detailed evaluations for two m values. The area and energy penalties associated with 1)–3) in DSM are to capture an m-bit segment

ISSN No: 2348-4845
International Journal & Magazine of Engineering, Technology, Management and Research
A Peer Reviewed Open Access International Journal

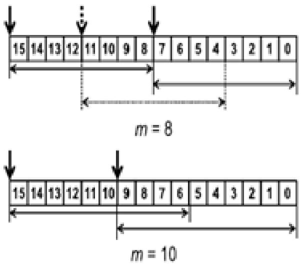
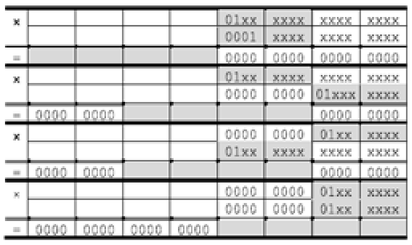
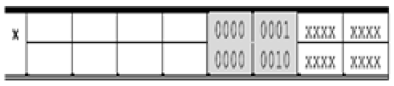


Fig.2 Possible starting bit positions of 8-b and 10-b segments indicated by arrows; the dotted arrow is the case for supporting three possible starting bit positions.



×					01xx	xxxx	xxxx	xxxx
					0001	xxxx	xxxx	xxxx
=	▨	▨	▨	▨	0000	0000	0000	0000
×					01xx	xxxx	xxxx	xxxx
					0000	0000	01xxx	xxxx
=	0000	0000	▨	▨	▨	▨	0000	0000
×					0000	0000	01xx	xxxx
					01xx	xxxx	xxxx	xxxx
=	0000	0000	▨	▨	▨	▨	0000	0000
×					0000	0000	01xx	xxxx
					0000	0000	01xx	xxxx
=	0000	0000	0000	0000	▨	▨	▨	▨

Fig. 3. Examples of 16×16 multiplications based on 8-b segments with two possible starting bit positions for 8-b segments. The shaded cells represent 8-b segments and the aligned position of 8×8 multiplication results.



×					0000	0001	xxxx	xxxx
					0000	0010	xxxx	xxxx

Fig. 4. Example of low accuracy for SSM16×16.

Starting from an arbitrary bit position in an n-bit operand because the leading one bit can be anywhere. Thus, we proposed to limit possible starting bit positions to extract an m-bit segment from an n-bit operand to two or three at most in SSM, where Fig. 2 shows examples of extracting 8-b and 10-b segments from a 16-b operand. Regardless of m and n, we have four possible combinations of taking two m-bit segments from two n-bit operands for a multiplication using the m-bit SSM. For a multiplication, we choose the m-bit segment that contains the leading one bit of each operand and apply the chosen segments from both operands to the m×mmultiplier. The SSM greatly simplifies the circuit that chooses m-bit segments and steers them to the m×mmultiplier by replacing two n-bit LODs and shifters for the DSM with two (n–m)-input OR gates and m-bit 2-to-1 multiplexers; if the first (n–m) bits starting from the MSB are all zeros, the lower m-bit segment must contain the leading one. Furthermore, the SSM also allows us to replace the 2n-bit shifter used for the DSM with a 2n-bit 3-to-1 multiplexer. Since the segment for each operand is taken from one of two possible segments in an n-bit operand, a 2m-bit result can be expanded to a 2nbit result by left-shifting the 2m-bit result by one of three possible shift amounts: 1) no shift when both segments are from the lower m-bit segments; 2) (n–m) shift when two segments are from the upper and lower ones, respectively; and 3) 2× (n–m) shift when both segments are from the upper ones, as shown in Fig. 3.

Note that the accuracy of an SSM with m = n/2 can be significantly low for operands shown in Fig. 4, where many MSBs ofm-bit segments containing the leading one bit are filled with zeros. On the other hand, such a problem becomes less severe as m is larger than n/2; there is an overlap in a range of bits covered by both possible m-bit segments as shown for m = 10 in Fig. 2. Thus, for an SSM with m = n/2, we propose to support one more bit position that allows us to extract an m-bit segment indicated by the dotted arrow in Fig. 2. This will be able to effectively capture operand pairs similar to the one shown in Fig. 4. Fig. 5 shows an SSM allowing to take an m-bit segment from two possible bit positions of an n-bit operand. The key advantage is its scalability for various m and n, because the complexity (i.e., area and energy consumption) of auxiliary circuits for choosing/steering m-bit segments and expanding a 2m-bit result to a 2nbit results scales linearly with m.

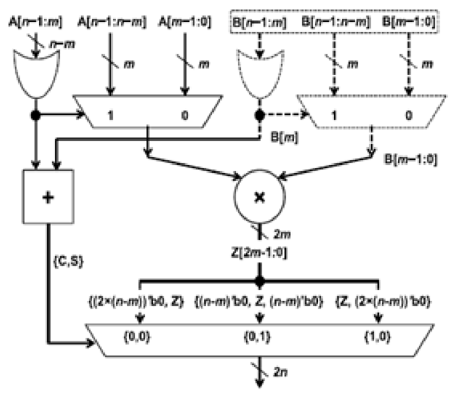


Fig. 5. Proposed approximate multiplier architecture; the logic and wires denoted by the dotted lines are not needed if B is preprocessed as proposed.

For applications where one of operands of each multiplication is often a fixed coefficient, we propose to pre compute the bit-wise OR value of B[n–1:m] and preselect between two possible m-bit segments(i.e., B[n–1:n–m] and B[m–1:0]) in Fig. 5, and store them instead of the native B value in memory. This allows us to remove the n–minput OR gate and the m-bit 2-to-1 multiplexer denoted by the dotted lines in Fig. 5. Finally, to support three possible starting bit positions for pickingan m-bit segment where m = n/2, the two 2-to-1 multiplexers at the input stage and one 3-to-1 multiplier at the output stage are replaced with 3-to-1 and 5-to-1 multiplexers, respectively, along with some minor changes in logic functions generating multiplexer control signals; we will show this enhanced SSM design for m = 8 and n = 16 (denoted by ESSM8×8) can provide as good accuracy as SSM10×10 at notably lower energy consumption.

III.IMPLEMENTATION RESULTS

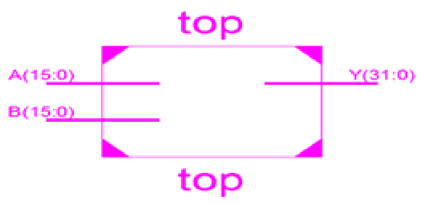


Fig. RTL Synthesis

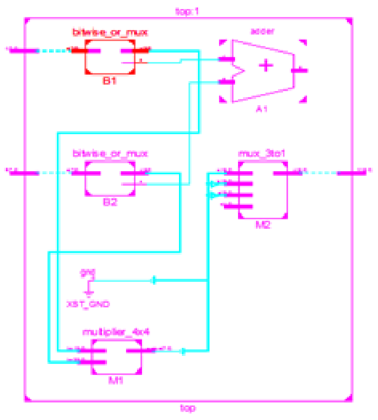


Fig. RTL Schematic view

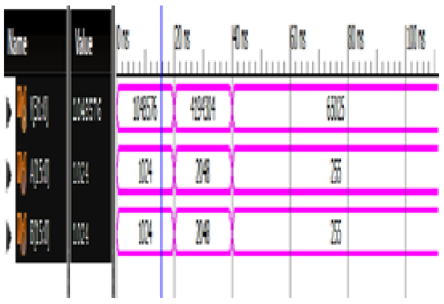


Fig. Simulation results

Design	Area (Slice LUTs)	Delay (in ns)
Multiplier using Array	57	16.515
Multiplier using Vedic	47	13.874

Fig. Comparison table

IV CONCLUSION:

The multiplier will be used in DSP and many applications as it is energy efficient. The segmentation method in this architecture is flexible and compact to design and implement and it may be used in any other signal processing applications which aim at energy efficient design.

V REFERENCES:

[1] Srinivasan Narayanamoorthy, Hadi Asghari Moghaddam, Zhenhong Liu, Taejoon Park, and Nam Sung Kim, July 2014, "Energy-Efficient Approximate Multiplication for Digital Signal Processing and Classification Applications", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Early Access Articles, pp. 1 – 5.

[2] Zdenek Vasicek and Lukas Sekanina, April 2014, "Evolutionary Design of Approximate Multipliers under Different Error Metrics", Design and Diagnostics of Electronic Circuits & Systems, 17th International IEEE Symposium, pp. 135 – 140.

[3] K. S. Ganesh Kumar, J. Deva Prasannam, M. Anitha Christy, March 2014, "Analysis of Low Power, Area and High Performance Multipliers for DSP applications", International Journal of Emerging Technology and Advanced Engineering, Volume 4, Issue 3,pp.278-382.

[4] Vinay K. Chippa, Srimat T. Chakradhar, Kaushik Roy and Anand Raghunathan, May- June 2013, "Analysis and Characterization of Inherent Application Resilience for Approximate Computing", IEEE Design Automation Conference (DAC), 2013 50th ACM / EDAC / IEEE, pp. 1- 9.

[5] Jie Han, Michael Orshansky, May 2013, "Approximate Computing: An emerging Paradigm for Energy-Efficient Design", Test Symposium (ETS), 2013 18th IEEE European, pp. 1-6.

[6]C. H. Chang and R. K. Satzoda, "A low error and high performance multiplexer-based truncated multiplier," IEEE Trans. Very Large ScaleIntegr. (VLSI) Syst., vol. 18, no. 12, pp. 1767–1771, Dec. 2010.

[7] P. Kulkarni, P. Gupta, and M. Ercegovac, "Trading accuracy for power with an under designed multiplier architecture," in Proc. 24th IEEE Int.Conf. VLSI Design (VLSID), Jan. 2011, pp. 346–351.