Dual segmentation approximate multiplier

Ling Li, Issam Hammad, and Kamal El-Sankary
Department of Electrical and Computer Engineering, Dalhousie University, Halifax, Canada
Email: lingli@dal.ca

This letter proposes a new design for a dual segmentation approximate multiplier. The proposed approximate multiplier uses the static segment method (SSM) to select an initial multiplication segment. Following that, the dynamic segment method (DSM) is utilized to further reduce the initial segment size. Using only the DSM segmenter leads to reduced performance gains. Therefore, the proposed multiplier uses a mixture of the SSM and DSM approaches. Compared to the DSM approach, the proposed design can reduce the energy by 37.90% for 32-bit multipliers, and by 16.68% for 16-bit multipliers. The DSM and proposed multipliers have almost identical accuracy.

Introduction: Approximate computing can be applied in error-resilient applications to reduce power, area, and delay [1–10]. Multiplication is a fundamental high-energy operation in image processing and deep learning applications [11–14]. Prior works have explored different techniques to reduce the cost of multiplication using approximate multipliers. Examples of these techniques include rounding the multiplicands to the nearest power of two [10], partial product matrix simplification (e.g., using approximate compressors and sub-multipliers) [5–6], and segment-based approximate designs (e.g., truncating the operands to designated bit-width) [7–9]. Segment-based approximate multipliers allow for a trade-off between accuracy and performance by adjusting segment size \( m \). In [7], a static segment method (SSM)-based multiplier was presented, which statically split the input operand into \( n \) segments. Then using OR gates and a MUX, the segment which contains the most significant one is selected. Hashemi et al. [8] extended the idea of leading one segment to implement the dynamic segment method (DSM) in approximate multiplier designs. In [9], a truncation and rounding-based scalable approximate multiplier (TOSAM) was proposed where the multiplicands were truncated with two different lengths and rounded to perform smaller core multiplications. The DSM-based multiplier can provide notably high accuracy, although it has a larger area and higher energy consumption than the SSM design. The SSM-based multiplier has lower accuracy, but it is faster and it consumes less energy compared to other segment-based approximate multipliers such as [8–9]. In this paper, a dual segmentation approximate multiplier is proposed. The proposed multiplier relies on using SSM initially to reduce the bit length of the input operands and then followed by dynamic segment selectors to further reduce the segment size of each operand. The proposed multiplier achieves better performance in terms of the delay, area and energy compared to the DSM-based multiplier while having almost the same accuracy level as the DSM-based multiplier.

Proposed approximate multiplier: Segment-based approximate multipliers can be classified into two categories: dynamic [8–9] and static [7]. The dynamic segment selector requires utilizing extra complex circuitry, which leads to significant energy and area overheads [7]. As seen in Figure 1a, the \( n \)-bit leading one detector (LOD) is implemented to locate the most significant one in an \( n \)-bit operand. Then an encoder and a MUX capture the following \( m - 1 \) bits and set the least significant bit of the segment to one. The LOD is the module with the highest power consumption and the largest area in the dynamic segment selector [7]. The static segment selector is much faster, smaller and consumes less energy compared to the dynamic segment selector. However, the approximation using the static segment selector results in large mean relative errors (MREs). The implementation for the static segment selector is presented in Figure 1b where the \( n \)-bit multiplicand is split into \( i (m \)-bit\) segments. Then using OR gates and a MUX, the segment which contains the leading one bit is detected. Thus, the \( n \)-bit static segment selector can be implemented in the DSM-based multiplier to truncate the \( n \)-bit input operands to \( n/2 \)-bit segments or even \( n/4 \)-bit segments, which can efficiently reduce the hardware overhead of the LODs. Hence, the dual segmentation design can achieve small improvements in delay, area and power dissipation.

In the proposed design, dual segmentation design, the static segment selector truncates the \( n \)-bit inputs to the \( k \)-bit segments as shown in Figure 2. As previously discussed, the dynamic segment selector of the DSM-based multiplier only needs to identify the leading one bit from the \( k \)-bit segment instead of the \( n \)-bit operand. After selecting the \( m \)-bit segments that contain the leading one bit of each \( k \)-bit segment, the chosen segments are applied to an accurate \( m \)-bit multiplier. The result of this multiplication is shifted according to the positions of the leading one bit of each \( n \)-bit operand to generate the final output. The main benefit of the dual segmentation logic is that the size of the most power-consuming components, LODs, is from \( n \)-bit down to \( k \)-bit for dynamic segment selectors.

As can be seen in Figures 3a and 4a, the dynamic segment selector captures \( n/2 \)-bits segment in a number starting from the leading one bit and set the least significant bit of the segment to one when the leading one bit is not in the least significant \( m \)-bit. In the case where the leading one bit is within the least significant \( m \)-bit of a number, then the most significant bit and the last significant bit of the \( m \)-bit segment are set as ‘1’, and the bits between them are approximated by zeros (see Figures 3b and 4b). Thus, the \( m \)-bit segment of the proposed selector and the dynamic segment selector always have the same most and least significant bits. Also, the dynamic segment selector and the proposed selector can produce the same \( m \)-bit segment when the leading one bit is not within the least \( m \)-bit of the \( k \)-bit segment. As expected, the proposed multiplier has similar accuracy as the DSM-based multiplier.

Figure 5 shows an example of a 16-bit dual approximate multiplication using \( k = 8 \) and \( m = 4 \). As can be seen in the figure,
Fig. 3 A general example of the m-bit dynamic segment approximation: (a) the leading one bit is not within the least significant m-bit, (b) the leading one bit is within the least significant m-bit

Fig. 4 A general example of the m-bit dual segment approximation: (a) The leading one bit is not within the least significant m-bit, (b) the leading one bit is within the least significant m-bit

Fig. 5 Example of 16-bit dual segmentation multiplier (k = 9, m = 4)

the result of the proposed multiplication is the same as that of the DSM-based approximate multiplication. This is due to the same m-bit segments that are captured in the proposed and DSM-based multipliers.

Results and discussion: In this section, the proposed design is compared with an exact multiplier and three previously proposed segment-based approximate multipliers. All designs were implemented using Verilog HDL and synthesized using Synopsys DC compiler the Synopsys in a 65 nm library at the typical process corner. All designs are unsigned multipliers. To evaluate the impact of changing the multiplier size, each multiplier is exploited to design 16-bit and 32-bit multipliers, and we fix k = 8 and m = 4. The accuracy is evaluated by using the MRE for 16-bit and 32-bit multiplier designs with ten million uniformly distributed random input pairs.

The results show that the SSM-based multiplier has the lowest delay, energy, area, energy-delay product (EDP) and power-delay-area (PDA) product, but it has the worst MRE. The DSM-based multiplier provides the highest accuracy. The proposed multiplier outperforms the DSM-based multiplier in terms of the delay, area and energy consumption while having almost the same accuracy level as the DSM-based multiplier.

The delay, energy, and EDP of the 16-bit proposed multiplier are approximately 12%, 17% and 27% lower than those of the DSM-based multiplier. Also, the 16-bit proposed multiplier has 21% smaller area compared to the DSM-based multiplier. As can be seen in Table 1, the MRE value of the 16-bit proposed multiplier is about 11% lower than the 16-bit SSM. The 32-bit proposed multiplier improves the speed, area, and energy up to 58%, 92%, and 98% compared to the exact multiplier as can be seen in Table 2. Additionally, the MRE value of the 32-bit proposed design is 18% lower than that of the 32-bit SSM-based multiplier.
This paper presented a new design for a dual segment-based approximate multiplier. This proposed design can approximate multipliers of unsigned 32-bit exact and approximate multipliers. The proposed design achieves a better performance-accuracy trade-off compared to previously proposed segment-based approximate multipliers. The efficiencies of the proposed multipliers as 16-bit and 32-bit designs were evaluated and compared against an exact multiplier and previously proposed segment-based approximate multipliers. The proposed design achieves a better performance-accuracy trade-off compared to previously proposed segment-based approximate multipliers.

Conclusion: This paper presented a new design for a dual segmentation approximate multiplier. This proposed design can approximate an n-bit multiplication using an n/4-bit multiplication or even an n/8-bit multiplication. The multiplier relies on using static segmentation initially followed by a dynamic segmentation. The static segment selector implemented in the proposed multiplier significantly reduces the hardware cost of the LOD of the DSM-based approximate multiplier. The efficiencies of the proposed multipliers as 16-bit and 32-bit designs were evaluated and compared against an exact multiplier and previously proposed segment-based approximate multipliers. The proposed design achieves a better performance-accuracy trade-off compared to previously proposed segment-based approximate multipliers.

Acknowledgments: The authors would like to thank the reviewers for their insightful suggestions and careful reading of the manuscript. The authors declare that there is no conflict of interest to disclose. This is an open access article under the terms of the Creative Commons Attribution License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited.

Received: 14 April 2021  Accepted: 19 May 2021
doi: 10.1049/iet-el.2021.12243

References

1 Han, J., Orshansky, M.: Approximate computing: An emerging paradigm for energy-efficient design. In: 2013 18th IEEE European Test Symposium (ETS), Avignon, France, 27–30 May 2013 https://doi.org/10.1109/ETS.2013.6509370.
2 Xu, Q., et al.: Approximate computing: a survey. *IEEE Des. Test. 33*(1), 8–22 (2016) https://doi.org/10.1109/MDAT.2015.2505723.
3 Venkatachalam, S., Ko, S.: Design of power and area efficient approximate multipliers. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* 25*(5), 1782–1786, (2017) https://doi.org/10.1109/TVLSI.2016.2643639.
4 Zervakis, G., et al.: Design-efficient approximate multiplication circuits through partial product perforation. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* 24*(10), 3105–3117 (2016) https://doi.org/10.1109/TVLSI.2016.2535398.
5 Strollo, A.G.M., et al.: Comparison and extension of approximate 4-2 compressors for low-power approximate multipliers. *Circuits Syst. I Regul. Pap. IEEE Trans.* 67*(9), 3021–3034 (2020)
6 Esposito, D., et al.: Approximate multipliers based on new approximate compressors. *IEEE Trans. Circuits Syst. I, Reg. Pap.* 65*(12), 4169–4182 (2018)
7 Narayananmooorthy, S., et al.: Energy-efficient approximate multiplication for digital signal processing and classification applications. *IEEE Trans. Very Large Scale Integr. Syst.* 23*(6), 1180–1184 (2015)
8 Hashemi, S., et al.: DRUM: a dynamic range unbiased multiplier for approximate applications. In: Proceeding IEEE/ACM International Conference Computer-Aided Design (ICCAD). Austin, TX, USA, 2–6 Nov. 2015
9 Vahdat, S., et al.: TOSAM: an energy-efficient truncation- and rounding-based scalable approximate multiplier. *IEEE Trans. Very Large Scale Integra (VLSI) Syst.* 27*(5), 1161–1173 (2019)
10 Zendegani, R., et al.: RoBa multiplier: a rounding-based approximate multiplier for high-speed yet energy-efficient digital signal processing. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* 25*(2), 393–401, (2017)
11 Kim, M.S., et al.: The effects of approximate multiplication on convolutional neural networks. *IEEE Trans. Emerging Top. Comp.* 1–1, (2021) https://doi.org/10.1109/TETC.2021.3050989.
12 Hammad, I., et al.: CNN inference using a preprocessing precision controller and approximate multipliers with various precisions. *IEEE Access.* 9, 7220–7232 (2021) https://doi.org/10.1109/ACCESS.2021.3049299.
13 Hammad, I., El-Sankary, K.: Impact of approximate multipliers on VGG deep learning network. *IEEE Access.* 6, 60438–60444 (2018) https://doi.org/10.1109/ACCESS.2018.2875376.
14 Hammad, I., et al.: Deep learning training with simulated approximate multipliers. In: 2019 IEEE International Conference on Robotics and Biomimetics (ROBIO), Dali, China, 6–8 Dec. 2019 https://doi.org/10.1109/ROBIO49542.2019.8961780.

Table 1. Circuit’s parameter of unsigned 16-bit exact and approximate multipliers of unsigned 16-bit exact and approximate multipliers

| Design | Delay (ns) | Power (uW) | Area (um^2) | Energy (fJ) | EDI (nsxfJ) | PDA (fsxum^2) | MRE (%) |
|--------|------------|------------|-------------|-------------|--------------|---------------|---------|
| Exact  | 2.0        | 256.7      | 2626        | 518.6       | 1047.5       | 1361776       | -       |
| Proposed | 1.3        | 34.6       | 676         | 48.1        | 66.9         | 32558         | 5.8     |
| DSM    | 1.6        | 36.6       | 854         | 57.8        | 91.3         | 49358         | 5.7     |
| SSM    | 0.9        | 15.9       | 395         | 14.3        | 12.9         | 5652          | 16.9    |
| TOSAM (2,4) | 1.6 | 37.9 | 854 | 60.7 | 97.1 | 51800 | 11.4 |

Table 2. Circuit’s parameter of unsigned 32-bit exact and approximate multipliers of unsigned 32-bit exact and approximate multipliers

| Design | Delay (ns) | Power (uW) | Area (um^2) | Energy (fJ) | EDI (nsxfJ) | PDA (fsxum^2) | MRE (%) |
|--------|------------|------------|-------------|-------------|--------------|---------------|---------|
| Exact  | 3.8        | 1343.1     | 10652       | 5036.8      | 18887.9      | 53651727      | -       |
| Proposed | 1.5        | 38.3       | 894         | 60.1        | 94.5         | 53813         | 6.8     |
| DSM    | 2.2        | 44.9       | 1519        | 96.9        | 209.4        | 147220        | 6.7     |
| SSM    | 1.0        | 17.9       | 629         | 17.6        | 17.2         | 11040         | 24.5    |
| TOSAM (2,4) | 2.3 | 45.7 | 1487 | 103.8 | 235.7 | 154394 | 12.3 |