JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Efficient Design of Approximate Multiplier using High-Speed Adder Compressor

[1]T. Chandra Mouli Swamy, [2]K. Suribabu

[1]PG Scholar, [2]Assisant Professor
[1, 2]Department of Electronics and Communication Engineering
[1, 2]Avanthi Institute of Engineering and Technology, Tagarapuvalasa, Vizianagaram, AP, India

Abstract: In this modern era, many digital systems are error-resilient, which allows us to take advantage of approximate computations. It makes the use of replacement of identical computing units by their counterparts. Approximate computing can also decrease the complexity at the designing levels by increasing performance and power efficiency. Adders and multipliers are the basic buildings blocks of many digital applications. These blocks are approximated in several ways. Research works are on the rise at many levels on approximate computing. Approximation at the designing level is more advantageous as the modifications are much easier than the preceding levels. A method of designing an Approximate Multiplier (AM) with a novel structure introduced in a 16-bit adder compressor is proposed. The 16-bit Adder Compressor (AC) is designed with 8-2 adder compressors in general. The 8-2 adder compressor is designed with 7-2 and 3-2 adder compressors and half adders. The existing and proposed multiplier is designed using Xilinx 14.7 in the frontend. The speed of the proposed multiplier is a 55.44% increase compared to Existing Multiplier.

Index Terms **- 7-2 and 3-2 AC, AM, approximate computing.**

I. INTRODUCTION

The need for approximation arises from the fact that exact computation requires more energy in different applications. That means wherever the accuracy is not a significant concern, and the design has to be energy efficient, we may take advantage of approximation that requires less energy than the exact one. For most digital circuits, adders and multipliers are the basic building blocks. Replacing the same building blocks with approximate ones results in energy-efficient designs. A multiplier is a device that multiplies any two operands and gives the corresponding result. Multiplication is nothing but the repeated addition of partial products. This involves adding partial products by using half adders and full adders based on the bit size of input operands. Logic gates are used to implement these adder circuits under different technologies. In the design of high-speed multipliers, compressors are used in the reduction tree to speed up the process. These compressors are implemented using full adders.

Moreover, the Integrated circuit(IC) era of emerging digital trends prefers compact size. This ensures the necessity of area-efficient designs for most of the digital circuits. At the same time, it allows the approximate values but not the exact ones to implement energy-efficient designs. To make the most of error tolerance, various techniques are available. These are of three types: (1) insistent voltage scaling; (2) truncation of bit-width; (3) use of imprecise building blocks.

The concept in [1] is imperfect, full adder cells to implement the multi-bit adder cells with minimized complexity at the transistor level. Reflection of errors due to approximation typical digital processing at higher levels may not impact the output quality much. Two new methods for approximate 4-2 compressors are proposed in [2] for implementing a multiplier and are analyzed for a dadda multiplier. Simulation of these methods at 1GHz frequency revealed significantly reduced power, delay, and transistor count. It uses XOR-XNOR combinations for the implementation of the compressor. [3] Proposed AMs for DSP applications. This technique takes „m‟ concomitant bits (i.e., m-bit segment) from each n-bit operand where m is greater than or equal to „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 can provide much higher accuracy. AM circuits proposed in [4] use the technique of partial product perforation. In this technique, the errors are bound and predictable. This approach can be used for any multiplier regardless of its architecture. Perforation skips the generation of partial products instead of cutting them. Thu decreasing the number of operands to be accumulated, reducing the delay. An AM with configurable partial error recovery is proposed in [5]. This mainly focuses on mitigating critical paths by using only simple but fast adders in the reduction tree. An inaccurate 4-2 counter is used for error correction in implementing the multiplier proposed in [6]. The existing multiplier [7] proposed two multipliers, one is for approximating all columns, and the second one approximates the least significant columns. It is based on the probability statistics of the logic block results. This paper presents a new approach for AM design using 16-bit AC.

Contribution of Paper, Briefly introduction AM applications in Section 1 and corresponding AM Literature Survey is seen in section 2. Architecture Explanation of Existing and Proposed AM see in section 3 and section 4. Finally, the Results Explanation and conclusion see in sections 5 and section 6.

II. LITERATURE SURVEY

A detailed presentation of the existing design work that is required for the proposed multiplier implementation. It covers the necessity of minimization of power since the power-efficient implementation is a very challenging task. At the same time, to achieve the high computational speed, existing work covers the minimization of power consumption by diminishing the switched capacitance that confers one by one as follows:

In the case of the sequential multiplier and the combinational multiplier performance, the multiplication mainly contains the generation of partial product and diminishment in the partial product. Subsequently, it follows the carry propagation addition. Considering the combinational case due to realization involves less area.

The Author Karthikeya Bharadwaj et al. [1] reported in his work multiplier design by utilizing the Wallace tree to minimize the power in addition to the area. The author employs the idea of 'carry-in prediction' to diminish the critical path. He also stated that by utilizing the Wallace tree, we easily optimize the design for the area and power. His simulation result achieves an accuracy of 99.86% to 99.967%. Later on, he was executed on the applications in real-time on benchmark images. Finally, he observed and concluded that there is a vast diminishment in power. At the same time, there is a massive reduction in the area needs need to implement a multiplier. The author also noticed that there is no loss of image quality.

The Author Weiqiang Liu et al. [2] delivered in his work about the realization of approximate computing by utilizing the Radix-4 booth multiplier that has more prominence in the application of image processing. The main goal of his approximate computation is to achieve low power in addition to high performance. He designed the booth multiplier by using radix-4 modified booth encoding, commonly known as MBE the algorithm. The author designed an approximate partial product by utilizing the approximate Wallace tree. He has finally concluded that his 16-bit radix-4 booth multiplier has a better approximation and more precision than the existing booth multiplier. The author achieved all these key merits with less propagation delay in addition to minimization of power consumption.

The Author Shiksha Bathla et al. [3] represented in their work about the high-speed 4-bit multiplier with low power consumption. He has proposed the four-bit multiplier by utilizing FinFET technology. Shiksha Bathla has done four-bit multiplier work using the HSPICE tool. Since when compared to metal oxide semiconductor technology (MOSFET), FinFET utilizes less power. Furthermore, he successfully delivered optimization delay term in FinFET based multiplier by mean of sleep mode and the tri-mode technique.

K. Banerji Srinivas et al. [4] delivered in their work about the multiplier that consumes less power in addition to the high-speed performance. One of the critical characteristics of his work is diminishing the switching activities to diminish the power dissipation since the row, and column multiplier diminishes the switching actions. The author utilized the tri-state buffer to be in command of the gating element. Furthermore, with the utilization of tri-state buffer, there is a vast diminishment in signal propagation delay. The author also stated that the multiplier that he has contributed consumes less power of about 20%. The row & column bypass multiplier, which is well suited for some significant applications like filtering.

Suryasnata Tripathy et al. [5] proposed low power multiplier architecture by utilizing the critical concept of Vedic mathematics, and this architecture is implemented by a mean of 45 Nanometer technology to achieve faster computing performance. The author implemented the four-bit as well as the eight-bit multiplier by using the sutra of urdhvatiryakbhyam.

Callaway et al. [6] represented the reduction in transitioactivitieson for a digital multiplier in their work. The author further demonstrated the switching activity that has been involved in a partial product with a diminishment in the hardware structure. In common practice, a high-speed multiplier can be called the multiplier that uses a superfluous binary adder tree. There are mainly two steps involved in the fixed point multiplication. The first step involves the generation of a partial product. Subsequently follows the accumulation of partial product. To achieve faster multiplication, it is necessary to follow the standard ways. First, one produces the less number of partial products by picking up the pace in accumulation in subsequent steps.

In their paper, D. Jakuline Moni et.al. [7] have made some significant points about the design of a low power booth multiplier that can perform sixteen-bit multiplication operations or two eight-bit multiplication operations. Furthermore, the product obtained at the output was truncated to diminish the power consumption. Subsequently, it will raise the speed of processing in compensation needed to surrender the precision of the output. This is multiplier performing at high speed because it omits the switching operation of the ineffective range of values.

Koyel Dey et al. [8] represent the unique procedure for designing the high-performance eight-bit multiplier by utilizing Vedic mathematics. The author implemented the essential procedure by adopting the sixteen-nanometer technology. Later on, the author compared the existing multiplier performance with the multiple channel complementary metal-oxide semiconductors (McCMOS) technology in addition to 65-nanometer technology. Furthermore, the author carried the fundamental analysis using T-spice simulation. Finally, it was concluded that multiplier by mean of 16-nanometer technology consumes less power when judged against other technologies.

Jinpeng Xing et al. [9] in his work presented high-performance multiplier-based FPGA. The author adopted the new coding algorithm that helped design the reliable 24x24 bit multiplier with low power dissipation. Moreover, it reduces the computational steps required to generate a partial product and diminishes the number of adders required for the multiplier. The author successfully delivered the 24x24 bit multiplier based on FPGA with the diminishment of power consumption of about 8.4 % compared to the existing algorithm.

III. EXISTING APPROXIMATE MULTIPLIER

The partial products $A_{m,n,}$ and $A_{n,m}$ in the columns containing more than three partial products are combined to propagate and generate signals as shown. These form the altered partial products $P_{m,n}$, and $G_{m,n}$. These are obtained as follows:

$P_{m,n} = A_{m,n} + A_{n,m}$

$G_{m,n} = A_{m,n}. A_{n,m}$

The reduction tree for obtaining partial products is shown in below Fig.3:

Fig.3. altered partial products formed by propagating and generating signals. [7]

In approximating the altered partial products, "generate" signals are accumulated column-wise using OR gates. OR gates used for a column having m generate signals are m/4. Partial products other than the generate signals are approximated using half adder, full add, and 4-2 compressor. In the Existing method, approximate half adder and full adder blocks are designed using the Adaptive Voltage Level technique, which is used to reduce the power consumption. In general, the 4-2 compressor is designed using two full adders. The proposed method includes a half adder with an XOR gate at the output to improve the accuracy. The Existing Method partial product reduction stage diagram sees in Fig. 4.

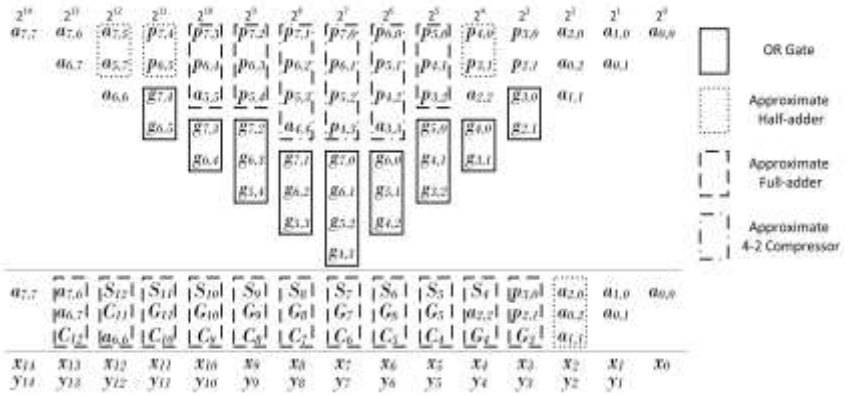

Fig. 4: Logic blocks used for approximation of altered partial products [7]

IV. PROPOSED APPROXIMATE MULTIPLIER

This section first explains the proposed AC and then discusses the proposed AM.

Compressors by far have been considered as the most efficient building blocks of a high-speed multiplier. It provides an advantage of accumulation of partial products at the expense of the least possible power dissipation. Rather than entirely summoning partial products with the help of the CSA/Ripple adder tree, a structure of compressors would complete the same task in much less time and simultaneously eradicate the problems of large power consumption and optimization of the area. When done using the conventional method of implementing Full Adders and Half Adders, this addition of partial products cannot account as much to lessening of delay associated with the critical path as when counter or compressors are used. The apparent preference of compressors over counters is the advantages it provides in terms of power, the number of transistors used, and the delay associated with the critical path (comprising of XOR's mainly). The compressor design implemented in this paper prefers both MUX's and XOR's.

The internal structure of the 3-2 adder compressor is presented in Fig. 5-a. Two XOR gates give the maximum delay. The final sum S of the 3-2 adder compressor is given in (1). The 3-2 adder compressor can also be used as a full-adder (i.e., mux-based full-adder) when the input C is used as a carry input.

$$S=Sum+2Carry \quad (1)$$

The internal structure of the 7-2 adder compressor is presented in Fig. 5-b. Ten XOR gates give the maximum delay. The final sum S of the 7-2 adder compressor is given in (2).

$$S=Sum+2(Cout1+Cout2+Carry) \quad (2)$$

In this paper 8-2 adder design using 3-2 and 7-2. The internal structure of the 8-2 adder compressor is presented in Fig. 6. The final sum S of the 8-2 adder compressor is given in (3)

.

$$S=Sum+2(Cout0+Cout1+Cout2+Cout3+Cout4+Carry) \quad (3)$$

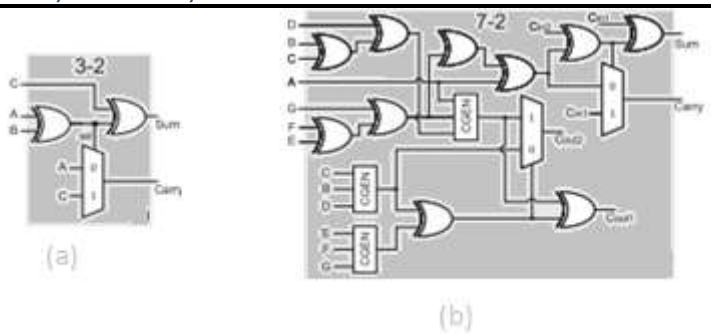

Fig. 5 Adder compressors internal structures: (a) 3-2; (b) 7-2. [8]

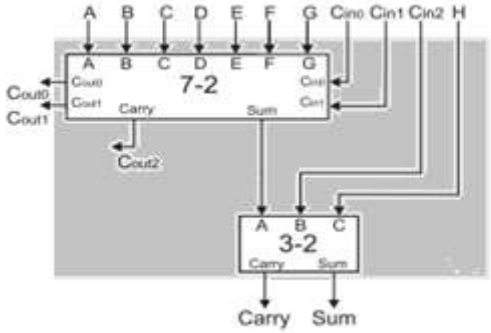

Fig. 6 The structure of 8-2 adder compressor with a Combination of 7-2 and 3-2 adder compressors [8]

Any multiplication, algorithm contains three steps but this summation of partial products is a crucial step to generate the final result. The performance of the multiplier depends on how fast partial products get added to obtain the final result. Many researchers can work in this area to achieve fast adders. The fundamental adder architecture is a Ripple Carry Adder. Further, it develops the number of adders such as Carry look-ahead adder, Carry select adder, Carry save adder and Carry skip adder etc. This ripple carry adder is well known for its stable structure and maximum delay because each step waits for the carry from the previous step. Carry look-ahead adder have a minimum delay but area associated with these adders are maximum. Carry skip adder gives more performance than ripple carry adder, but it consists of extra hardware circuitry to skip the carry generated. Carry save adder gives the further addition by reducing addition there is the number of three into two. The major drawback carry save adder consumes larger area.

Further, carry select adder uses the two ripples carry adders, and it doesn't wait for the previous stage to execute. The carry select adder with higher bits exhibits excellent area and speed trade-off compared with other adder architectures. Many modifications can be dined in carry-save adder for sacrificing its speed for the area.

To implement a *W*-bit 3-2, 4-2, 5-2, and 7-2 adder compressors,, a recombination of partial Carry and Sum terms is needed. To make the recombination of *Carry* and *Sum, a cascade of half-adder and full-adders circuits is used* in a Ripple Carry form, as presented in the example of figure 7, for an 8-bit 4-2 compressor. In this work, we use a more efficient Carry Look Ahead adder (CLA) to recombine the partial results. We have used a pipeline stage between the line of the compressors and the adder line used to recombine the partial results (highlighted in figure 7). The design of a 16-Bit high adder using different 8-2 adder compressors is shown in Fig. 8. The main objective in this section we can design high-speed adders using four 8-2 adder compressors and already explained in section 4

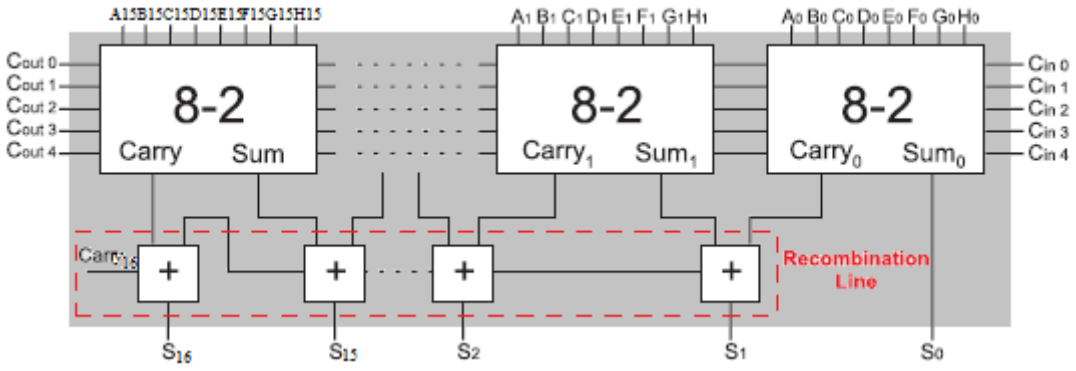

Fig. 8: 16- bit adder using 8-2 ACs [9]

The Proposed Multiplier is shown in Fig. 9. In this architecture, we can use the above section 16-bit adder using 8-2 ACs at partial production and final sum stage after completing the process of OR gate operation. The implementation above multiplier develops the Verilog code, and then simulation we can use Xilinx 14.7.

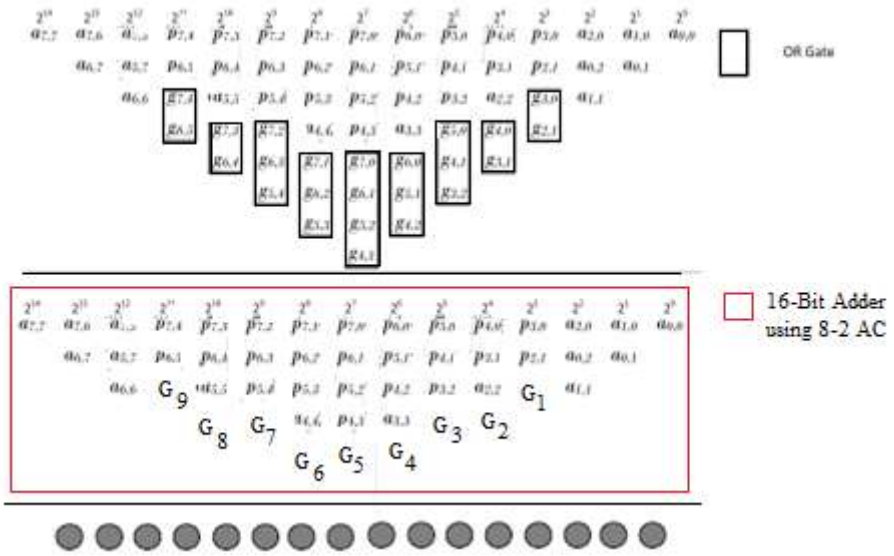

Fig. 9 The Proposed AM with 16-Bit adder

V. RESULTS AND DISCUSSION

The design was synthesized on Xilinx ISE, and the functional verification of approximate existing and the proposed multiplier was done on Xilinx ISIM. The targeted device is Spartan-3e of the Spartan family. The grade speed of the design is set to -5. The following section contains the results obtained by synthesizing the design in Xilinx ISE 14.7. The comparison result is shown in Table 1, and the resultant comparison plot is shown in Fig. 10.

Table 1 Comparison of Existing and Proposed AM in terms of area and delay

	AREA			DELAY (ns)
	SLICES	LUTS	FFS	
Existing Multiplier	43	85	63	16.52 ns
Proposed Multiplier	41	60	64	7.36 ns

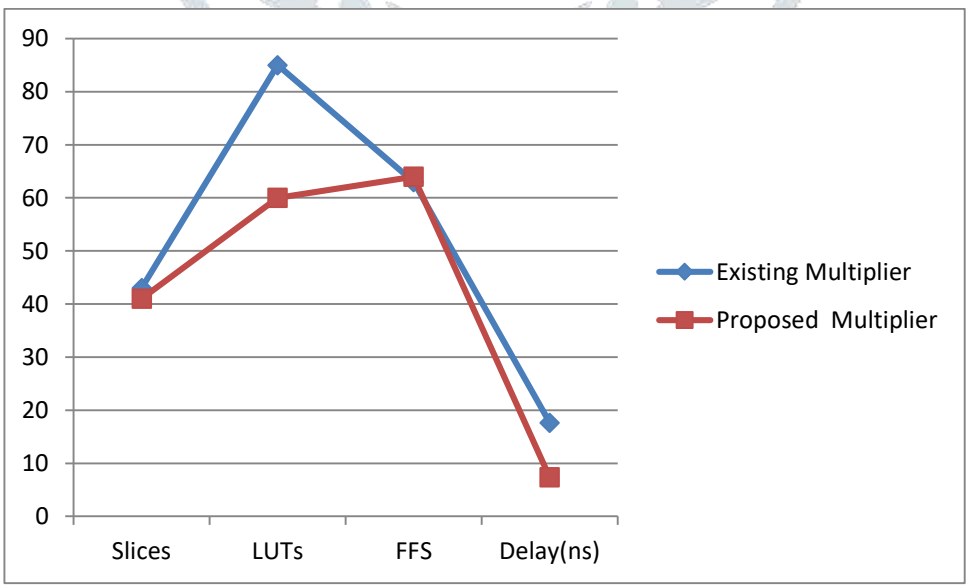

Fig. 10 Performance analysis of Existing and Proposed Multiplier

VI. CONCLUSION

The proposed multiplier increases the speed so that it can multiply and be used in image processing applications. It is highly efficient, 55.44 % delay reduced compared to that of previous works. The adaptive voltage level at the ground reduces the power consumption by lifting up the ground potential whenever required and decreasing the voltage through transistors. So this multiplier architecture can be used for high-speed applications like data mining and cases where the error occurrence is not a significant concern.

REFERENCES

[1] V. Gupta, D. Mohapatra, A. Raghunathan, and Roy, "Low-power digital signal processing using approximate adders," IEEE Trans. Comput. Aided Design Integr. Circuits Syst., vol. 32, no.1, pp.124 137, Jan.2013.

[2] A.Momeni, J. Han, P.Montuschi, and F.Lombardi, "Design and analysis of approximate compressors for multiplication,"*IEEE Trans.Comput*., vol. 64, no. 4, pp. 984–994, Apr. 2015.

[3] S.Narayanamoorthy, H.A.Moghaddam, Z.Liu, T.Park, andN.S.Kim,"Energy-efficient approximatemultiplication for digital signal processing and classification applications," *IEEE Trans. Very Large ScaleIntegr. (VLSI) Syst*., vol. 23, no. 6, pp. 1180–1184, Jun. 2015.

[4] G.Zervakis, K. Tsoumanis, S. Xydis, D. Soudris, and K. Pekmestzi, "Design-efficient approximate multiplication circuits through partial product perforation," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst*., vol. 24, no. 10, pp. 3105–3117, Oct. 2016.

[5] C. Liu, J. Han, and F. Lombardi, "A low-power, high-performance approximate multiplier with configurable partial error recovery," in Proc.Conf. Exhibit. (DATE), 2014, pp. 1–4.

[6] C.H.Lin and C.Lin, "High accuracy approximate multiplier with error correction," in Proc. IEEE 31st Int. Conf. Comput. Design, Sep. 2013, pp. 33– 38.

[7] Suganthivenkatachalam and seek-bum KO, "Design of power and area efficient approximate multipliers" *IEEE transactions on VLSI systems*, pp. 1-8, Aug. 2017.

[8] J. S. Altermann, E. A. C. da Costa, and S. Bampi, "Fast Forward and Inverse Transforms for the H.264/AVC Standard using Hierarchical Adder Compressors," *in Proc. IEEE/IFIP Int. Conf. VLSI Syst. Chip (VLSI-SoC),* Madrid, Spain, pp. 310–315, Sep. 2010.

[9] Bianca Silveira, Guilherme Paim, Cláudio Machado Dinizand and Sergio Bampi, "Power-Efficient Sum of Absolute Differences Hardware Architecture Using Adder Compressors for Integer Motion Estimation Design", *IEEE Trans. on Circuits and Systems–I*, pp. 1-12, 2017.