IMPLEMENTATION OF 8X8 DADDA MULTIPLIER USING APPROXIMATE COMPRESSION FOR IMAGE ENHANCEMENT

Harish Rao. B , Ramesh Kumar. V

Department of Electronics and Communication Engineering,

SRM University, Kattankulathur-603 203, India.

ABSTRACT

Inexact (Approximate) computing is an attractive paradigm for digital processing. Inexact computing is particularly interesting for computer arithmetic designs. This project deals with the analysis and design of two new approximate 4-2 compressors for utilization in a multiplier. These designs rely on different features of compression, such that imprecision in computation can make up for circuit-based figures of merit of a design. Two different schemes for utilizing the proposed approximate compressors are proposed and analysed for a Dadda multiplier. Extensive simulation results are provided and hardware implementation of the Dadda multiplier using approximate compression is carried out with the help of a Field Programmable Gate Array (FPGA).

1. INTRODUCTION

Digital multipliers are among the most critical arithmetic functional units in many applications, such as the Fourier transform, discrete cosine transforms, and digital filtering. The throughput of these applications depends on multipliers, and if the multipliers are too slow, the performance of entire circuits will be reduced. To reduce the computation error, many compensation techniques were presented for array multipliers. There is an apparently trade-off between accuracy and hardware complexity. Recently, compensation works have been increasing focused on reducing the truncation error on the Booth multiplier. Multipliers have been important since the introduction of the digital computers. Multiplication occurs frequently in Digital Signal Processing (DSP) systems, communication systems and other Application Specific Integrated Circuits (ASICs). Because of the significance of multiplication in scientific and engineering computations, this area has received much attention in the past decades which has led to a number of implementation techniques for multiplication.

The vast variety of application areas for multipliers exhibits different requirements for speed, area, power consumption and other specifications. Based on these requirements, which are imposed from the system that the multiplier will be operating in, different characteristics of the multiplier will be given different priorities. It is the designers task to choose a suitable multiplication algorithm and

implementation method according to these priorities. Traditionally, the design priorities have been given to speed and area. However, the fast evolution of digital systems has caused a major paradigm shift in the past years. Now, other parameters like low-power, flexibility, testability and reliability have entered into the play. Power dissipation has become an important constraint in the design of digital systems.

This is even more important for battery-powered applications where the energy budget is extremely limited. Low-power design has become a new area in VLSI technology and power-aware design is inevitable in the new Electronic Design Automation (EDA) tools. Multiplier is generally computationally heavy circuit parts. Basically a large number of transistors with high transition activities have to be devoted to perform the multiplication. More transistors with high transition activities mean more internal capacitance, more overall switching and consequently more power dissipation. Also the total leakage current is expected to be large in a multiplier because of its large active area. Multipliers are among the main contributors of area and power consumption in a DSP system and, more importantly, they are usually placed in the critical paths of such systems. Throughout this thesis, a design methodology is proposed for reducing the dynamic and static power dissipation in parallel multipliers. It is assumed that the multiplier will be operating in real-time systems where high-speed is essential.

Therefore, among the variety of implementation methods, high-speed parallel implementation methods are addressed. The optimization method is an interconnection reordering algorithm based on the input data characteristics. It is applicable directly on all full-adder based parallel multipliers. With some changes the optimization method is applicable for majority of the reduction schemes. The optimization only modifies the inter connects between logic gates and the logic gates and the architecture remains unchanged.

2. DESIGN AND ANALYSIS

Addition and multiplication are widely used operations in computer arithmetic; for addition full-adder cells have been extensively analyzed for approximate computing. It has compared these adders and proposed several new metrics for evaluating approximate and probabilistic adders with respect to unified figures of merit for design assessment for inexact computing applications. For each input to a circuit, the error distance (ED) is defined as the arithmetic distance between an erroneous output and the correct one. The mean error distance (MED) and normalized error distance (NED) are proposed by considering the averaging effect of multiple inputs and the normalization of multiple-bit adders. The NED is nearly invariant with the size of an implementation and is therefore useful in the reliability assessment of a specific design. The trade-off between precision and power has also been quantitatively evaluated.

However, the design of approximate multipliers has received less attention. Multiplication can be thought as the repeated sum of partial products; however, the straightforward application of approximate adders when designing an approximate multiplier is not viable, because it would be very inefficient in terms of precision, hardware complexity and other performance metrics. Several

approximate multipliers have been proposed in the literature. Most of these designs use a truncated multiplication method; they estimate the least significant columns of the partial products as a constant. In an imprecise array multiplier is used for neural network applications by omitting some of the least significant bits in the partial products (and thus removing some adders in the array).

A truncated multiplier with a correction constant is proposed. For an n×n multiplier, this design calculates the sum of the n+k most significant columns of the partial products and truncates the other n-k columns. The n+k bit result is then rounded to n bits. The reduction error (i.e. the error generated by truncating then-k least significant bits) and rounding error (i.e. the error generated by rounding the result to n bits) are found in the next step. The correction constant (n+k bits) is selected to be as close as possible to the estimated value of the sum of these errors to reduce the error distance.

2.1 EXACT COMPRESSOR MODULE

The main goal of either multi-operand carry-save addition or parallel multiplication is to reduce n numbers to two numbers; therefore, n-2 compressors (or n-2 counters) have been widely used in computer arithmetic. An-2 compressor is usually a slice of a circuit that reduces n numbers to two numbers when properly replicated. In slice i of the circuit, the n-2 compressor receives n bits in position i and one or more carry bits from the positions to the right, such as i – 1 or i – 2. It produces two output bits in positions i and i + 1 and one or more carry bits into the higher positions, such as i + 1 or i + 2.

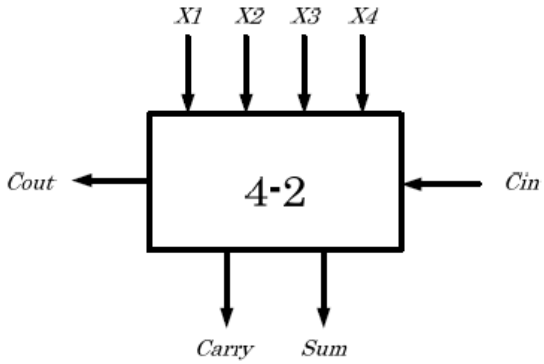

Fig 1.Exact compression module.

2.2 APPROXIMATE COMPRESSOR MODULE

The mentioned phenomenon here is a combined effect two designs of an approximate compressor are proposed. Intuitively to design an approximate 4-2 compressor, it is possible to substitute the exact full-adder cells an approximate full-adder cell. However, this is not very efficient, because it produces at least 17 incorrect results out of 32 possible outputs, i.e. the error rate of this inexact

compressor is more than 53% (where the error rate is given by the ratio of the number of erroneous outputs over the total number of outputs). Two different designs are proposed next to reduce the error rate; these designs offer significant performance improvement compared to an exact compressor with respect to delay, number of transistors and power consumption.

2.3 MULTIPLIER MODULE

The impact of using the proposed compressors for multiplication is investigated. A fast (inexact) multiplier is usually composed of three parts (or modules). The multiplier consists of Partial product generation, A Carry Save Adder (CSA) tree to reduce the partial products' matrix to an addition of only two operands and A Carry Propagation Adder (CPA) for the final computation of the binary result. In the design of a multiplier, the second module plays a pivotal role in terms of delay, power consumption and circuit complexity. The use of approximate compressors in the CSA tree of a multiplier results in an approximate multiplier. An 8×8 unsigned Dadda tree multiplier is considered to assess the impact of using the proposed compressors in approximate multipliers. The proposed multiplier uses in the first part AND gates to generate all partial products. In the second part, the approximate compressors proposed in the previous section are utilized in the CSA tree to reduce the partial products. The last part is an exact CPA to compute the final binary result.

3. APPROXIMATE COMPRESSION

3.1 APPROXIMATE COMPRESSOR DESIGN 1

The carry output in an exact compressor has the same value of the input cin in 24 out of 32 states. Therefore, an approximate design must consider this feature. In Design 1, the carry is simplified to cin by changing the value other 8 outputs. Since the Carry output has the higher weight of a binary bit, an erroneous value of this signal will produce a difference value of two in the output. For example, if the input pattern is "01001", the correct output is "010" that is equal to 2. By simplifying the carry output to cin, the approximate compressor will generate the "000" pattern at the output (i.e. a value of 0). This substantial difference may not be acceptable; however, it can be compensated or reduced by simplifying the cout and sum signals. In particular, the simplification of sum to a value of 0 reduces the difference between the approximate and the exact outputs as well as the complexity of its design. Also, the presence of some errors in the sum signal will results in a reductions of the delay of producing the approximate sum and the overall delay of the design (because it is on the critical path).

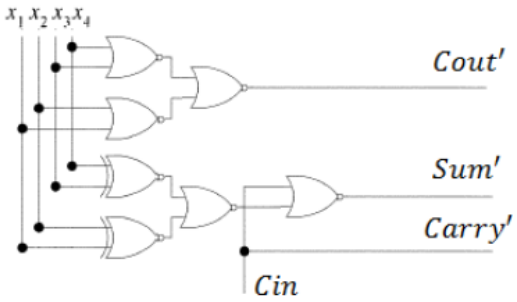

Fig 2. Approximate compressor design 1 gate level diagram.

3.2 APPROXIMATECOMPRESSOR DESIGN 2

A second design of an approximate compressor is proposed to further increase performance as well as reducing the error rate. Since the carry and cout outputs have the same weight, the proposed equations for the approximate carry and cout in the previous part can be interchanged. In this new design, carry uses the right hand side of the module and cout is always equal to cin; since cin is zero in the first stage, cout and cin will be zero in all stages. So, cin and cout can be ignored in the hardware design.

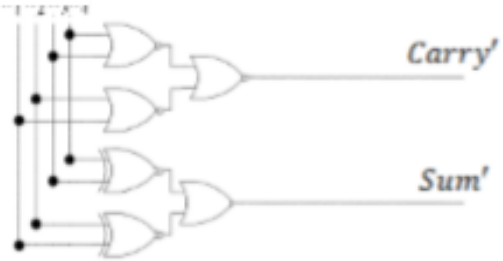

Fig 3. Approximate compressor design 2 gate level implementation.

4. 8X8 DADDA MULTIPLIER

An 8×8 unsigned Dadda tree multiplier is considered to assess the impact of using the proposed compressors in approximate multipliers. The proposed multiplier uses in the first part AND gates to generate all partial products. In the second part, the approximate compressors proposed in the previous section are utilized in the CSA tree to reduce the partial products. The last part is an exact CPA to compute the final binary result. The reduction circuitry of an exact multiplier for n=8. In this figure, the reduction part uses half-adders, full-adders and 4-2 compressors; each partial product bit is represented by a dot. In the first stage, 2 half-adders, 2 full-adders and 8 compressors are utilized to reduce the partial products into at most four rows. In the second or final stage, 1 half-adder, 1 full-adder and 10 compressors are used to compute the two final rows of partial products. Therefore, two

stages of reduction and 3 half-adders, 3 full-adders and 18 compressors are needed in the reduction circuitry of an 8×8Dadda multiplier.

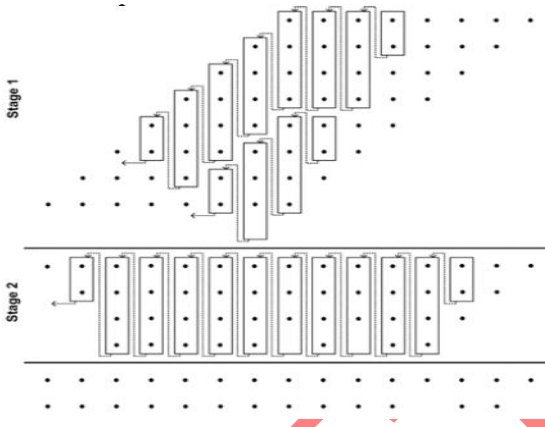

Fig 4. 8X8 Dadda multiplier architecture.

However, when the energy of the incoming electrons aligns with that of one of the internal energy levels, the energy of the electrons outside the well is said to be "in resonance" with the allowed energy inside the well. Then, maximum current flows through the device at this resonant voltage or peak voltage (V_p) called the peak current (I_p).As the voltage increases further the current through the device drops due to reduction in tunneling until the voltage reaches the valley voltage (V_v). The current at this voltage is the valley current (I_v). The negative differential resistance property which can be exploited for high speed and compact circuits.

The Conductance-Voltage curve in the fig.3 shows the maximum conductance at the peak voltage (V_p) and the diode does not conduct at the valley voltage (V_v). The effectiveness of the operation of a particular RTD often is characterized by how well defined are the peak and valley in the current versus voltage plot. This is measured by the peak-to-valley current ratio.

5. SIMULATON RESULTS

Verilog HDL is a Hardware Description Language (HDL). A Hardware Description Language is a language used to describe a digital system, for example, a computer or a component of a computer. One may describe a digital system at several levels. For example, an HDL might describe the layout of the wires, resistors and transistors on an Integrated Circuit (IC) chip, i. e., the switch level. Or, it might describe the logical gates and flip flops in a digital system, i. e., the gate level. The Verilog simulation outputs were obtained for the modules which have been analysed so far. The outputs are as follows:

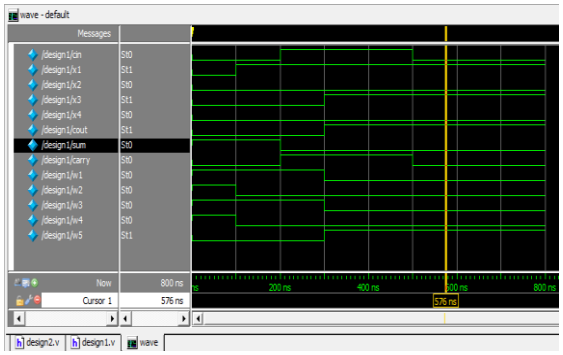

Fig 5. Exact compressor Verilog output waveform.

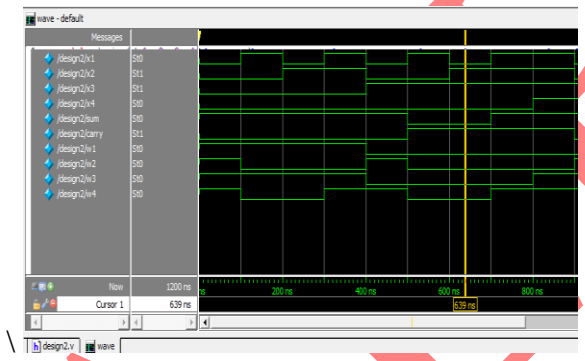

Fig 6. Approximate compressor design 1 Verilog output waveform.

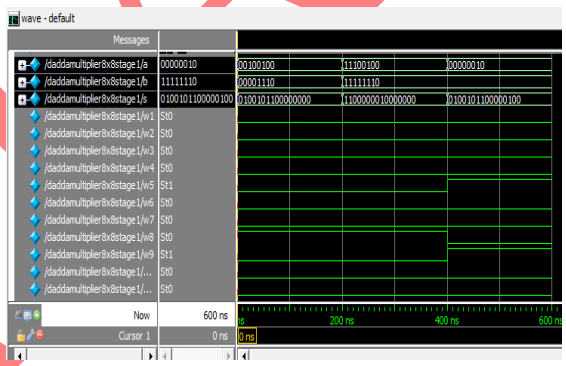

Fig 7. Approximate compressor design 2 Verilog output waveform.

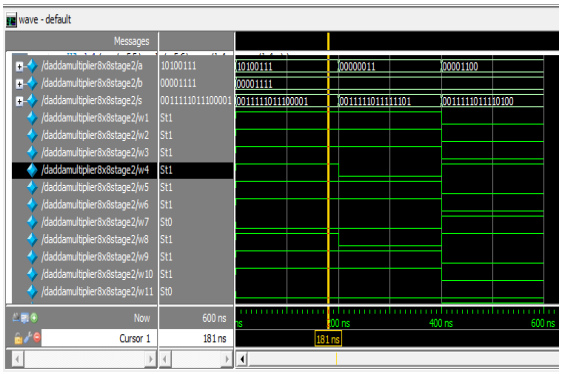

Fig 8. 8X8 Dadda multiplier Verilog output waveform.

6. IMAGE MULTIPLICATION DEPICTION

ModelSim SE is our UNIX, Linux, and Windows-based simulation and debug environment, combining high performance with the most powerful and intuitive GUI in the industry. The following output results were obtained on the ModelSim environment.

Fig 9. ModelSim Input image 1.

Fig 10. ModelSim Input image 2.

Fig 11. Multiplied Output Image.

7. HARDWARE IMPLEMENTATION

A Field-Programmable Gate Array is an FPD featuring a general structure that allows very high logic capacity. FPGAs offer high narrow logic resources. FPGAs also offer a higher ratio of flip-flops to logic resources. Xilinx XC3S400 4PQ208C was used to interface the 8X8 Dadda multiplier program code.

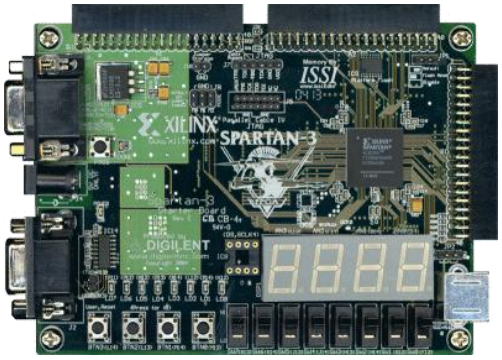

Fig 12. SPARTAN XC3S400 PQ208C Board.

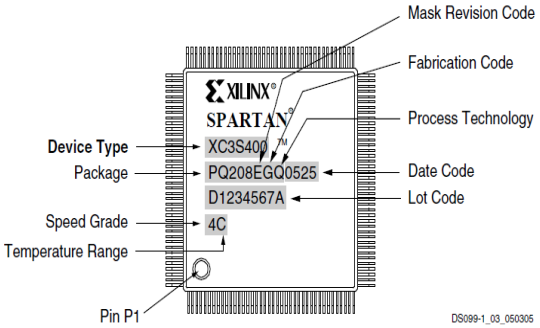

Fig 13. Spartan-3 QFP package for part number SPARTAN XC3S400 PQ208C.

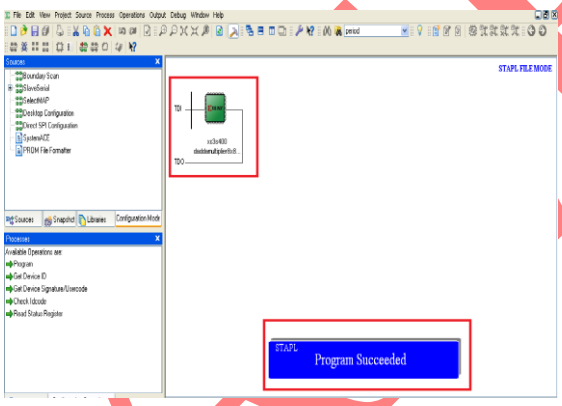

Fig 14. Xilinx® FPGA IDE.

8. CONCLUSION

Computer arithmetic offers significant operational advantages for inexact computing; an extensive literature exists on approximate adders. However, this paper has initially focused on compression as used in a multiplier. This paper has shown that by an appropriate design of an approximate compressor, multipliers can be designed for inexact computing; these multipliers offer significant advantages in terms circuit-level. Although not discussed and beyond the scope of this manuscript, the proposed designs may also be useful in other arithmetic circuits for applications in which inexact computing can be used. In our paper, the proposed design is only for 8- bit multiplier design using approximate compressors. And in our future we can try to implement the 16-bit multiplier design by using exact or approximate compressors and the design is implemented by using Modelsim and/or Xilinx software Tools.

REFERENCES

[1] J. Liang, J. Han, F. Lombardi, "New Metrics for the Reliability of Approximate and Probabilistic Adders," IEEE Transactions on Computers,vol. 63, no. 9, pp. 1760 - 1771, 2013.

[2] V. Gupta, D. Mohapatra, S. P. Park, A. Raghunathan, K. Roy, "IMPACT: IMPrecise adders for low-power approximate computing," Low Power Electronics and Design (ISLPED) 2011 International Symposium on. 1-3 Aug. 2011.

[3] S. Cheemalavagu, P. Korkmaz, K.V. Palem, B.E.S. Akgul, and L.N. Chakrapani, "A probabilistic CMOS switch and its realization by exploiting noise," in Proc. IFIP-VLSI SoC, Perth, Western Australia, Oct. 2005.

[4] H.R. Mandeni, A. Ahmadi, S.M. Fakhraie, C. Lucas, "Bio-Inspired Imprecise Computational Blocks for Efficient VLSI Implementation of Soft-Computing Applications," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 57, no. 4, pp. 850-862, April 2010.

[5] M. J. Schulte and E. E. Swartzlander, Jr., "Truncated multiplication with correction constant," VLSI Signal Processing VI, pp. 388–396, 1993.

[6] E. J. King and E. E. Swartzlander, Jr., "Data dependent truncated scheme for parallel multiplication," in Proceedings of the Thirty First Asilomar Conference on Signals, Circuits and Systems, pp. 1178–1182, 1998.

[7] P. Kulkarni, P. Gupta, and MD Ercegovac, "Trading accuracy for power in a multiplier architecture", Journal of Low Power Electronics, vol. 7, no. 4, pp. 490--501, 2011.

[8] C. Chang, J. Gu, M. Zhang, "Ultra Low-Voltage Low- Power CMOS 4-2 and 5-2 Compressors for Fast Arithmetic Circuits," IEEE Transactions on Circuits & Systems, Vol. 51, No. 10, pp. 1985-1997, Oct. 2004.

[9] D. Radhakrishnan and A. P. Preethy, "Low-power CMOS pass logic 4-2 compressor for high-speed multiplication," in Proc. 43rd IEEE Midwest Symp. Circuits Syst., vol. 3, 2000, pp. 1296–1298.

[10] Z. Wang, G. A. Jullien, and W. C. Miller, "A new design technique for column compression multipliers," IEEE Trans. Comput., vol. 44, pp. 962–970, Aug. 1995.

[11] J. Gu, C. H. Chang, "Ultra Low-voltage, low-power 4-2 compressor for high speed multiplications," in Proc. 36th IEEE Int. Symp. Circuits Systems, Bangkok, Thailand, May 2003.

[12] M. Margala and N. G. Durdle, "Low-power low-voltage 4-2 compressors for VLSI applications," in Proc. IEEE Alessandro Volta Memorial Workshop Low-Power Design, 1999, pp. 84–90.

[13] B. Parhami, "Computer Arithmetic: Algorithms and Hardware Designs," 2nd edition, Oxford University Press, New York, 2010.

[14] K. Prasad and K. K. Parhi, "Low-power 4-2 and 5-2 compressors," in Proc. of the 35th Asilomar Conf. on Signals, Systems and Computers, vol. 1, 2001, pp. 129–133.