ISSN: 2278 – 909X
International Journal of Advanced Research in Electronics and Communication Engineering (IJARECE)
Volume 4, Issue 8, August 2015

DSP Design using Approximate Multiplier

Maria Sunny
Electronics and Communication Engineering Department,
Sree Narayana Gurukulam College of Engineering, Kadayiruppu, Kerala, India

***Abstract*—Approximate computing is an emerging trend for digital processing computation at nanoscale and used in applications such as multimedia, image processing that can tolerate errors and yet produce meaningful results. For the use in such applications, an inexact 16-bit multiplier using approximate compressor is implemented in this paper. Efficient implementation of such a multiplier, along with an area efficient carry select adder is used in a custom digital signal processor. Experimental results show that the proposed design accomplishes significant reduction in area and thus make it suitable for applications, where size is of prime importance.**

***Index Terms*—Approximate Multiplier, Carry Select Adder, Compressor, Flexible DSP Programmable Truncated Multiplier.**

I. INTRODUCTION

Methodologies for approximate (or inexact) computing rely on the fact that many applications like multimedia, image processing, can tolerate some errors and imprecision in computation and, therefore, the solution can tolerate some degree of uncertainty. Commonly used multimedia applications have Digital Signal Processing (DSP) blocks as their backbone.

Most of these DSP blocks implement image and video processing algorithms, where the final output is either an image or a video for the consumption by humans. The limited perception of human vision allows the outputs of these algorithms to be numerically approximate (inaccurate) rather than accurate. This relaxation on numerical exactness allows us to carry out approximate computation. Deterministic, exact and precise models are not always suitable to be applied on these situations.

Multiplication is commonly required in digital signal processing and act as one of the fundamental building block. Parallel multipliers [8], [11] give high speed method for multiplications, but require large area for VLSI implementation. The increase in use of portable communication, computing devices and advancement in mobile multimedia applications has made power consumption and area critical to optimize in the design of digital signal processing architectures.

Full or direct multiplier implementations of an N X N bit multiplication gives a 2N -bit product. In order to keep the full accuracy, DSP architecture would need an ever-growing bit width that would be impractical to implement. To avoid this, results are usually truncated [5], [6] to keep results within the limits of the architecture bit width.

While many specific applications require the bit width of inputs and the outputs of the multiplier to be the same, general purpose digital signal processors need flexibility to support the generation of large output results in accumulators where the magnitude of the output is bigger than the multiplier inputs. Programmable truncated multipliers are multipliers where parts of the partial product matrix can be disabled at run-time by introducing a truncation bit in the matrix [1]. Though the power is reduced, area increases as the 2-input and-gate is replaced by 3-input and-gate. In the design of a fast multiplier, compressors [7], [9], [10] have been extensively used to speed up the partial product reduction tree and decrease power dissipation. Multipliers based on approximate compressors have been used.

This paper is organized as follows: Section II is a review of the methodology used. Section III shows the proposed method. Results and discussion is presented in Section IV. Conclusion of work is given in Section V.

II. METHODOLOGY

A. *Approximate Compressor*

Originally there are two designs proposed for approximate compressors [2]. In this paper, we concentrate on design 2. In [2], the *carry* and *cout* outputs have the same weight and *cout* is always equal to *cin* while comparing the truth table of exact compressor and design1. Again, since *cin* is zero in the first stage, *cout* and *cin* will be zero in all stages.

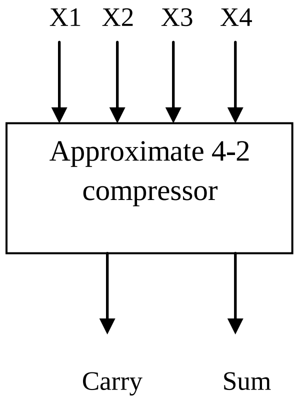

Fig.1. Approximate Compressor, Design 2

Therefore, *cin* and *cout* of exact compressor can be ignored in the hardware design as shown in Fig 1.So the equation for the approximate compressor, design 2 will be as follows:

$$Sum' = (\overline{x1 \oplus x2} + \overline{x3 \oplus x4}) \qquad (1)$$

$$Carry' = (\overline{\overline{x1x2} + \overline{x3x4}}) \qquad (2)$$

Fig.2 shows the gate level implementation of design 2. Design 2 becomes the addition of four inputs as *cin* and *cout* are ignored from the design. Table I shows the truth table of second approximate design of 4-2 compressor.

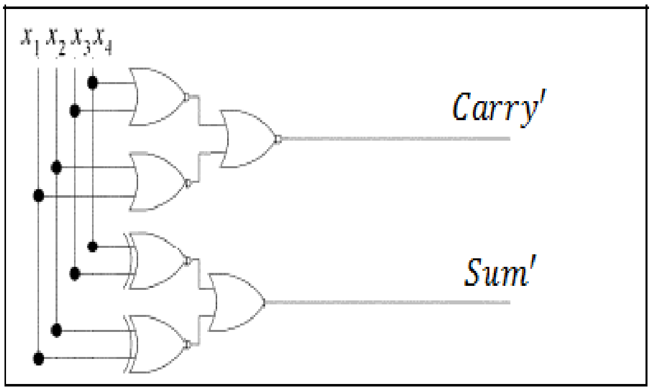

Fig.2. Gate Level Implementation of Design 2

Table I: Design 2, Approximate Compressor

X4	X3	X2	X1	Carry'	Sum'	Difference
0	0	0	0	0	1	1
0	0	0	1	0	1	0
0	0	1	0	0	1	0
0	0	1	1	0	1	-1
0	1	0	0	0	1	0
0	1	0	1	1	0	0
0	1	1	0	1	0	0
0	1	1	1	1	1	0
1	0	0	0	0	1	0
1	0	0	1	1	0	0
1	0	1	0	1	0	0
1	0	1	1	1	1	0
1	1	0	0	0	1	-1
1	1	0	1	1	1	0
1	1	1	0	1	1	0
1	1	1	1	1	1	-1

B. Approximate Multiplier

An exact multiplier is usually comprised of three parts:

•Partial product generation.

•A Carry Save Adder (CSA)

•A Carry Propagation Adder (CPA)

The first part is the generation of partial product matrix. The second part is a Carry Save Adder (CSA) tree to reduce the partial products' matrix to an addition of only two operands. The third part comprises of a Carry Propagation Adder (CPA) for the final computation of the binary result.

In the design of a multiplier, the second module, ie, the carry save adder, plays a pivotal role in terms of delay, power consumption and circuit complexity. Approximate compressors when used in the CSA tree of a multiplier, we get an approximate multiplier.

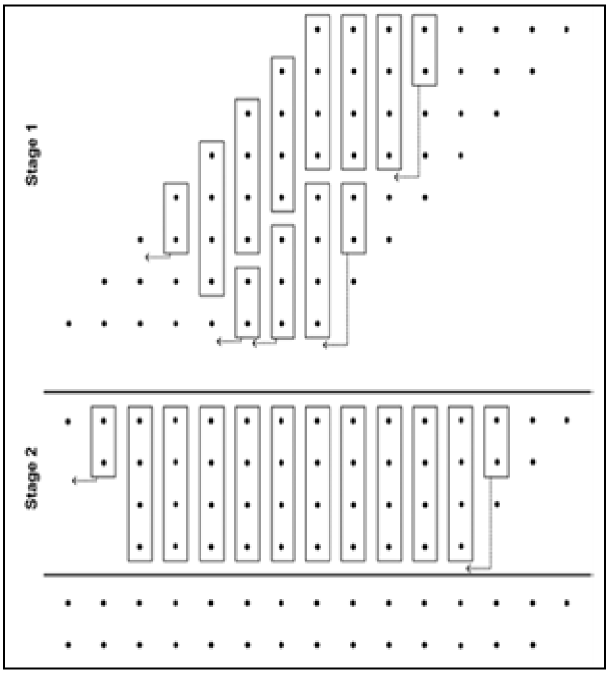

Fig.3. Reduction circuitry of multiplier using Design 2

An *8×8* unsigned Dadda tree multiplier is used to assess the impact of using the approximate compressors in the multiplier. The multiplier uses AND gates in the first part to generate all partial products. In the second part, the approximate compressors of design 2 are used in the CSA tree to reduce the partial products. The last part is an exact CPA to compute the final binary output. Fig.4 shows 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 where each partial product bit is represented by a dot.

Since Design 2 does not have *cin* and *cout*, the reduction circuitry of this multiplier requires a lower number of compressors (Fig.3). Multiplier uses 6 half-adders, 1 full-adder and 17 compressors. It has better performance in terms of delay and power consumption.

III. PROPOSED METHOD

The design 2 approximate multiplier is implemented into a custom DSP. The architecture presented in this paper is a custom-designed DSP structure with minimum control logic and core importance is given to arithmetic unit designed with an accumulator, barrel shifter, approximate multiplier and an area efficient carry select adder.

Fig.4 shows the DSP architecture used in this paper. The main characteristics [1] of the architecture are described as follows:

1) Control Unit: The Control Unit is a small 5 -stage pipeline that fetches and decodes instructions, and controls the data flow, arithmetic unit and the I/O ports DSP includes four general purpose registers within its Control unit.

2) Custom Instruction Set: The set of instructions implemented has been designed to maximize the usage of the Arithmetic Unit, optimizing the power reductions achievable by the multiplier. Instructions are 32 bits wide and allow access to one or both memory blocks in a single instruction. The instruction set is split into different functional areas, as explained below:

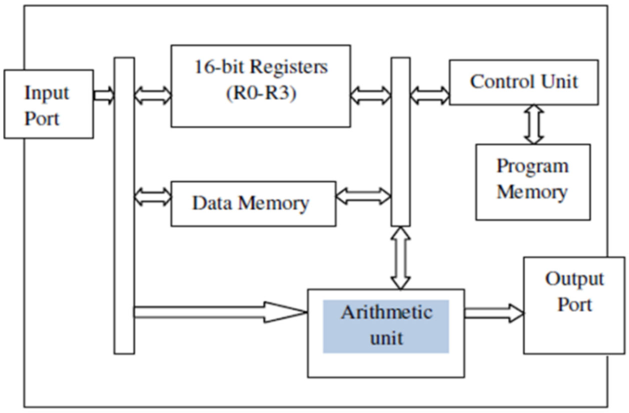

Fig.4. DSP Architecture

a) Arithmetic Instructions: The Arithmetic instructions are displayed in Table 2. It is focused on the arithmetic capabilities of the unit. The range of instructions dedicated to arithmetic operations includes ADD for addition, MULT for multiplication and MAC for multiply-and-accumulate operations.

b) Flow Control Instructions: Flow Control Instructions allow the device to perform relative and absolute jumps in the program and data memory blocks in an independent fashion.

c) Looping Instructions: Looping instructions have been implemented aimed at a reduction in flow-control instructions, maximizing the duty of the arithmetic unit.

d) Data Flow Instructions: Data Flow instructions, make reference to instructions for storing and loading data and also inputting and outputting samples to/from the system.

3) Memory blocks: Memory on the PTMAC architecture is formed by a 1024, 32 bit Program Memory block and two 512, 16 bit Data Memory blocks. All three memories can be accessed on a single clock cycle, as they are directly connected to the control unit.

4) Arithmetic Unit: The Arithmetic Unit is the core unit of this DSP. A block diagram of the Arithmetic Unit is displayed in Fig.5. It consists of a multiply and accumulate structure with main units as:

- a 16-bit approximate multiplier,
- a 40-bit area efficient carry select adder [3],
- an accumulator and ,
- a 40-bit barrel shifter [4] for scaling and rotating the accumulated value.

Table II: Arithmetic instructions

Mnemonics	Commands(Hex)
ADD	10xxxxxx
ADD_RD,A,B	13xABBBB
MULT	20xxxxxx
MAC	30xxxxxx
MAC_ RD,A,B	33Xabbbb
SQ_ACC,A	32xxxxxA
SHIFT_ACC	90xxABCC

Other instructions like looping, flow control,dataflow along with its mnemonic,command and description are given in[1].

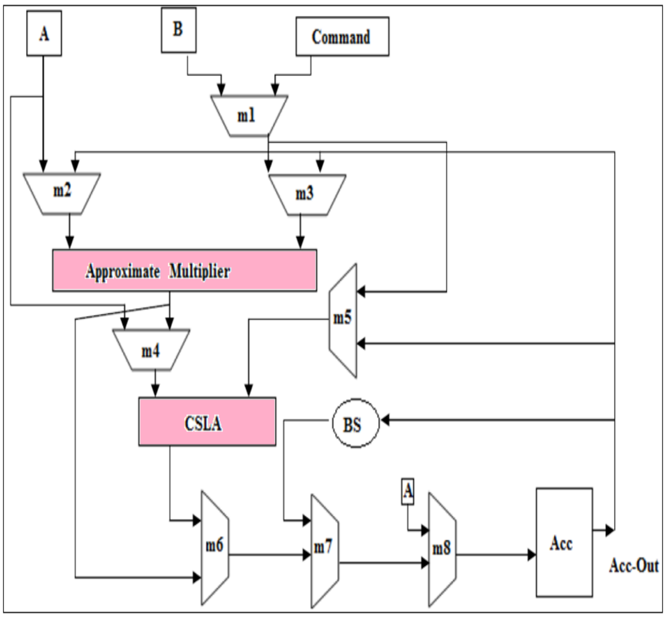

Fig.5. Arithmetic Unit Diagram

IV. RESULTS & DISCUSSION

DSP is implemented in Xilinx 13.3 using Verilog programming language. A 16x16 approximate multiplier is implemented and a comparison is made with programmable truncated multiplier. Performance comparison in terms area are shown in Table III.

Table III: Area Comparison of Multipliers

Metrics compared	Programmable Truncated Multiplier(PTM)	Approximate Multiplier (AM)
No. of Slices (4656)	414	**292**
No .of 4 i/p LUT's (9312)	720	**509**

DSP with programmable truncated multiplier (PTM) uses ripple carry adder as the adder unit while DSP with approximate multiplier uses area efficient carry select adder. Arithmetic units of these DSP's are being compared. Performance comparison in terms area are shown in Table IV and V. Tables show that there is significant area reduction when approximate multiplier is used in place of programmable truncated multiplier. Also comparison of DSP in terms of area indicates that DSP using approximate compressors can be used for various applications that are concerned with size reduction.

Table IV: Performance Comparison of Arithmetic Units

Metrics compared	Arithmetic Unit of PTM	Arithmetic Unit of AM
No. of Slices (4656)	1256	**1026**
No .of 4 i/p LUT's (9312)	2429	**1970**

Table V: Performance Comparison of DSP

Metrics compared	DSP with PTM	DSP with AM
No. of Slices (4656)	2032	**1754**
No .of 4 i/p LUT's(9312)	3964	**3435**

V. CONCLUSION

In this paper, DSP with approximate multiplier is proposed. This is best suited for applications like multimedia, image processing, portable devices etc. where size becomes an important factor. The proposed DSP can be used in applications where inexact (approximate) result is tolerable, yet producing useful results. .Results show that 16 bit approximate multiplier is effective in performance while compared to a programmable truncated multiplier. Also logic unit can be incorporated as a future work, to the arithmetic unit to convert it into an arithmetic and logic unit, so that it can be used in many other applications too.

ACKNOWLEDGMENT

The author is grateful to the reviewers. I am also grateful to Prof. Divya S for helping me understand the complexity of the algorithm. I express my gratitude to Dept. of ECE, Sree Narayana Gurukulam College of Engineering for supporting me generously with their tools.

REFERENCES

[1] Manuel de la Guia Solaz,Wei Han, and Richard Conway, "A Flexible Low Power DSP With A Programmable Truncated Multiplier," IEEE Transaction on Circuits and Systems, Vol 59, n0.11.

[2] A.Momeni,J.Han Member, P.Montuschi, and F. Lombardi, Fellow, Design and Analysis of Approximate Compressors for Multiplication "IEEE Trans. Computer., vol. 56, no. 4, pp, Apr. 2013.

[3] B Ramkumar and Harish M Kittur. " Low Power and Area Efficient Carry Select Adder", IEEE Transaction on VLSI, Vol 20, no 2.

[4] Michael .J. Schulte and E .George Walters, "Design Alternatives Of Barrel Shifter", IEEE journal on Circuits and Systems.

[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] 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.

[8] 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.

[9] 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.

[10] 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

[11] S. Kidambi, F. El-Guibaly, and A. Antoniou, "Area-efficient multipliers for digital signal processing applications," IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process., vol. 43, no. 2, pp. 90–95, Feb.1996