Design of Step-up Inexact MAC (IMAC) Unit for DSP Applications

P.V.S.R.Bharadwaja, M.Venkata Naresh, Neelima Koppala, J Sai Krishna

Abstract--- Digital Signal Processing is the processing of the analog signal into digital signal. Present epoch of Digital Signal Processor's need a rapid and very compacted processor's. The Digital Signal Processing enables the analog audio and video signals to process, transmit, store, reproduce and manipulate the information from one form to another form. But size of the system increases if the all the processing is to be done and as the size increases the delay also increases. Area of the system is very important parameter to be considered than the accuracy in the present constraint. Hence we concentrate on reducing the area and increasing the speed of the processor. Hence in this paper we state the concept of inexactness and we apply the same to some DSP processor application. This paper was designed using MIPS Verilog HDL. The design was synthesized and simulated in Xilinx 14.2 ISE. The power Calculations were calculated using Isim, a tool of Xilinx Xpower Analyzer.

1. INTRODUCTION

Digital Signal Processing is generally the numerical handling of the signals generally to measure, filter, acquire and compress the analog signals. These are generally characterized by using digital signals to represent the signals in terms of discrete time or discrete frequency in numerical format to permit the digital processing of the signals. These type of models can be created by the process of sampling. Some computations are in generally required to digitalize the signal. These Computations in generally use Analog to Digital Converter (ADC) and they may also require an Digital to Analog Converter (DAC) is required generally to reconstruct the signal back into analog format. The conversion of analog signal to discrete signal may be complex enough but is very advantageous compared to analog processing and has lot many applications. The basic applications of the Digital Signal Processing will be audio and speech signal processing. There were separate processors called as Digital Signal Processors. Now a day's FPGA technology is being used for Digital Signal Processors [1]. A digital Signal processor is a special processor with a special function of Digital Signal Processing. The objectives of Digital Signal Processors are to asses, dribble and constrict the analog signals [2]. There is a necessity for a consecrate processor for the digital signal processing as they have best power efficiency and they can be used in any Portable devices where power consumption is an important constraint. DSP's in generally use special processor architectures that are able to fetch multiple data or instructions in parallel. Digital signal Processing algorithm's generally use large amount of mathematical operations and are to be computed quickly i..e that should have less delay and as they are used in several applications the area utilized by the circuit should be very less.

2. BASIC UNITS USED

There are many multiplier techniques used [3][4][5][6]. After the comparison of several types of multiplier technique the Vedic sutra [7] is used here.

2.1. Urdhava Triyambakam

Vedic sutra which is generally used for the combinations is the Urdhava Triyambakam Sutram. As the name specifies this sutra is helpful in multiplying both vertically and cross wise. The procedure to do the same is showed below steps.

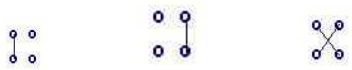

Fig 1(a) Fig 1(b) Fig 1(c)

Figure 1 : Steps showing working of Urdhava Triyambakam

Step 1: Multiply the bits which are vertical as showing in Fig 1(a)

Step 2: Now multiply the cross terms as shown in Fig 1(b)

Step 3: Now multiply the last two terms as shown in Fig 1(c)

If we consider the following steps in terms of the example let us consider we multiply 37 and 33

Step 1: Multiply the first digits of the two number 3*3 = 9

Step 2: Multiply the terms and Add the terms diagonally (3*3) + (3*7) = 30. As 30 is a 2 digit number put the last digit nest to 9 and the other digit below 9.

Step 3: Now multiply the last place and put the values in the similar fashion

Hence the value will be 1221.

This was based on the concept of generation of all the partial products parallel. The sum of the partial products and the sums are calculated in advance. Hence we call this as a concurrent technique which is well suited for all the processor.

The DSP processor, in generally use any of these sutra's.

Revised Manuscript Received on March 10, 2019.

P.V.S.R.Bharadwaja, Assistant Professor, Department of ECE, Sree Vidyanikethan Engineering College, A.Rangampet, Tirupati, Andhra Pradesh, India – 517102 (bharat0507@gmail.com)

M.Venkata Naresh, Assistant Professor, Department of ECE, Sree Vidyanikethan Engineering College, A.Rangampet, Tirupati, Andhra Pradesh, India – 517102 (nareshmvenkat@gmail.com)

Neelima Koppala, Assistant Professor, Department of ECE, Sree Vidyanikethan Engineering College, A.Rangampet, Tirupati, Andhra Pradesh, India – 517102 (neelumtech17@gmail.com)

J Sai Krishna, Assitant Professor, Department of ECE, SVCEW,Tirupati, A.P, India

2.2. Dadda Multiplier

There are lot many number of Multipliers [8][9][10][11]. A sequence of matrix in which the order is determined to have minimum number of reduction stages [12][13]. As Proposed by Dadda the sequence of heights of matrix are determined to give minimum number of reduction stages. To reduce the N by N partial product matrix, dadda multiplier helps in developing a sequence of matrix heights which are found by working back from the final two-row matrix. The dot representation of 7x7 Dadda multiplications is shown in Fig 2.

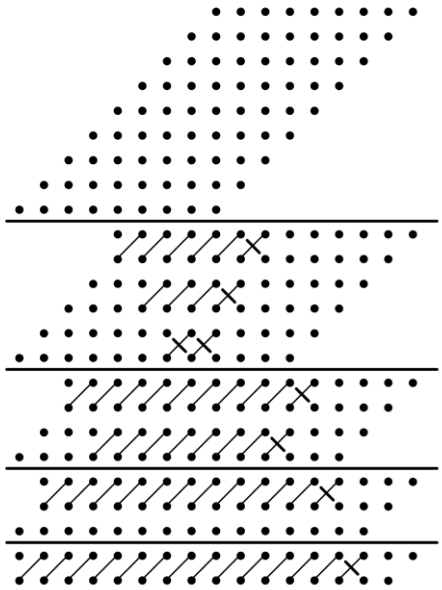

Fig 2: Dot Representation of 8 x 8 Dadda Multiplier

In order to realize the minimum number of reduction stages, the height of each intermediate matrix is limited to the least integer that is no more than 1.5 times the height of its successor. The process of reduction for a Dadda multiplier is developed using the following recursive algorithm

1. Let d1=2 and dj+1 = [1.5*dj], where dj is the matrix height for the jth stage from the end. Find the smallest such that at least one column of the original partial product matrix has more than dj bits
2. In the jth stage from the end, employ (3, 2) and (2, 2) counter to obtain a reduced matrix with no more than dj bits in any column.
3. Let j = j-1 and repeat step 2 until a matrix with only two rows is generated.

This method of reduction, because it attempts to compress each column, is called a column compression technique [14][15]. Another advantage of utilizing Dadda multipliers is that it utilizes the minimum number of (3, 2) counters. Therefore, the number of intermediate stages is set in terms of lower bounds: 2, 3, 4, 6, 9. For Dadda multipliers there are N2 bits in the original partial product matrix and 4.N-3 bits in the final two row matrix. Since each (3, 2) counter takes three inputs and produces two outputs, the number of bits in the matrix is reduced by one with each applied (3, 2) counter therefore}, the Total number of (3,2) counters is #(3, 2) = N2, length of the carry propagation adder is CPA length = 2.N–2.– 4.N+3. The different types of adders used in the system are [16][17].

For Higher Order Dadda Multiplier, since the complexity of representation and design of Dadda Multiplier increases, we utilize the concept of Vedic multiplication i.e., Urdhava Triyambakam Sutram as shown in Fig 3.

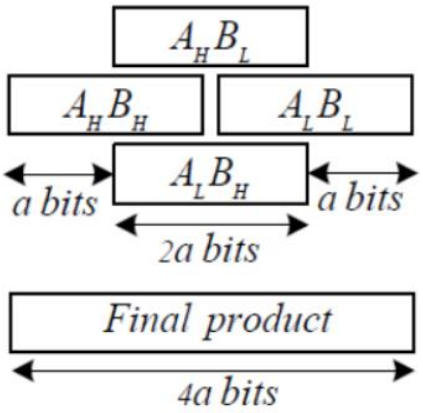

Fig 3: Format for inputs for recursive Vedic multiplication.

The technique shown in Fig 3 is used for designing the 16 x 16 multiplication where the basic multiplier will be 8 x 8 Dadda Multiplier. Here A_H and B_H represent the MSB bits of 16-bit inputs A and B. Also A_L and B_L represent the LSB bits of 16-bit inputs A and B. hence instead of performing 16 x 16 Dadda Multiplication directly, four 8 x 8 multiplications are performed in parallel and are added to obtain the final result. In similar Manner a 32 x 32 Dadda Multiplication is performed using this Vedic Multiplication Concept.

3. INEXACTNESS VS EXACTNESS

From the above discussion it is aware that the Digital Signal Processors should be fast and occupy the less area i...e less area should be occupied and the exactness may not be so required. As the series of pulses will be there if there minor deviation also the system will not damage. So Inexactness is nothing but decrease of the accuracy which can indeed reduce the area and increase the speed of the system. To explain this inexactness the probability of the inexactness is less. Let us consider an example of a Full Adder. We are always evident of the fact that the sum of the Full Adder is S = A^B^C and the carry of the full adder is C_{out} = AB+ BC+ CA. But the area is increasing when we are considering the same $C_{out.}$ When the area is increasing the size of the system is increasing and obviously the delay. But this is not so apt in DSP processor's as we require the fast rather than the accuracy so the mixture of the fastness by minimizing the delay and there by the area occupied by the system. This concept of decrease in accuracy is called inexactness and the circuits built up by using inexactness are called Inexact Circuits. Let us verify the inexactness by using a Full Adder Gate. For inexactness let is consider the sum equation be S=A^B^C and Carry = A & B. Now let us table the values of adder for both previous regular and adder and an inexact adder for the values.

Table 1: Comparison of Exactness and Inexactness of Full Adder

INPUTS			OUTPUTS		Actual Output	
A	B	C	S	COUT	S	COUT
0	0	0	0	0	0	0
0	0	1	1	0	1	0
0	1	0	1	0	1	0
0	1	1	1	0	0	1
1	0	0	1	0	1	0
1	0	1	0	0	0	1
1	1	0	0	1	0	1
1	1	1	1	1	1	1

In the Table 1, when we compare the Actual Output to the Outputs of the Inexactness we can see only two changes in the carry are happening. The percentage of inexactness is only 2/16=0.125.Hence the deviation will be very less or will be equal to the exact values. So this concept of inexactness is explained above. The simulation of Inexact Adder is shown below in Fig 4.

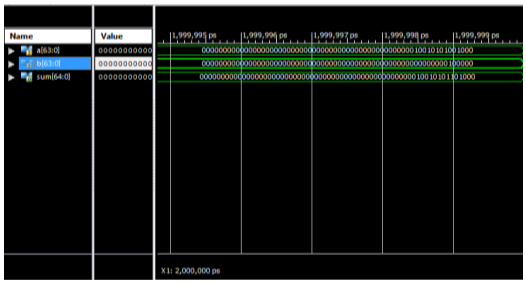

Fig 4: Results of an inexact 64-bit Adder

The same concept of inexactness can also used in Dadda Multiplier. As explained above Dadda is a different type of Multiplier which reduces the time of computation i.e., the Delay and also increases the accuracy. Where the Dadda Multiplier in this paper Multiplier two 32'bit Values and Gives the outputs as a 64-bit Number. The output of the Dadda Multiplier is as shown below in Fig 5.

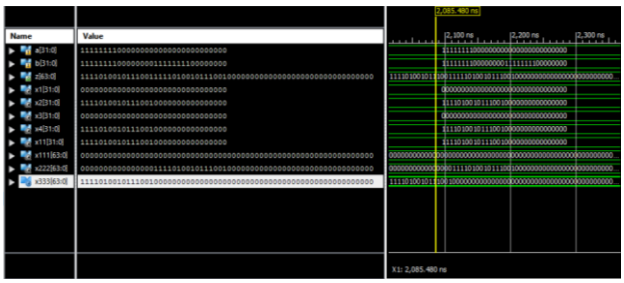

Fig 5: Output of 32-bit Dadda Multiplier

Thus the delay in the previous circuit has been reduced by using inexact Adder.

4. EXACT MAC

The block diagram of the Exact MAC [18] that includes the components Dadda Multiplier, Exact Adder and a register [19][20] to store the data are shown in Fig 6, where the inputs are represented as a and b each of 32-bit length, in addition to these a clock input as clk is considered so that the results are estimated for every clock cycle and the reset signal is used to start over the function if necessary. The output is then a 65-bit out signal that shows the actual output of the MAC Unit for

Every clock cycle. The Simulation waveform of the Exact MAC is shown in Fig 6, where the inputs are given as a=253 and b=28 then the multiplication output is a*b=253*28=7084, which at every rising clock yields and addition of value of 7084 to the result.

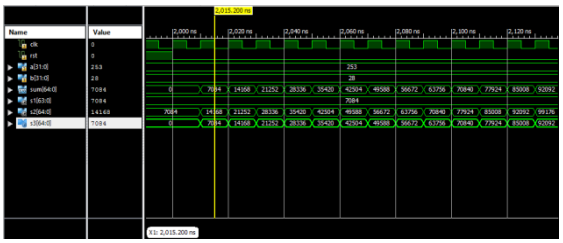

Fig 6: Simulation Results of Exact MAC

The Synthesis is performed for logic optimized FPGA Spartan3E with device XC3S500E-5FG320. The extracted details represent occupied slices as 1502 out of 4656 available slices, 2689 out of 9312 available 4-input LUTs. The power utilized for this design is a logic power of 0.00094W, total signal power of 0.00138W, I/O power of 0.00005W which yields to a total hierarchical power of 2.31mW.

5. INEXACT MAC (IMAC)

The main application of the IMAC units is in Digital Signal Processing. The IMAC unit is designed with inexact adder with a negligible percentage error. It consists of inexact 32-bit Dadda multiplier which uses the inexact full adders considered for the design, an inexact 64-bit adder and a 65-bit register to trigger required changes to the output The Simulation waveform of the IMAC Unit is shown in Fig 7, where the inputs are given as a=8226 and b=20996 then the multiplication output is a*b=8226*20996=172713096, which at every rising clock yields and addition of value of 172713096 to the result.

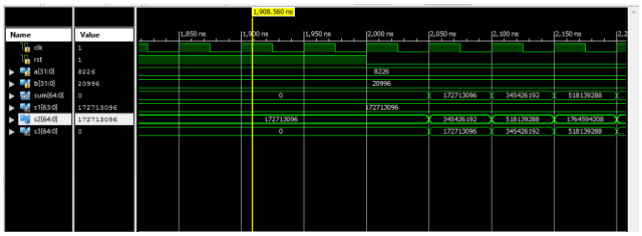

Fig 7: Simulation Results of Inexact MAC Unit

The Synthesis is performed for logic optimized FPGA Spartan3E with device XC3S500E-5FG320. The extracted details represent occupied slices as 1268 out of 4656 available slices, 2431 out of 9312 available 4-input LUTs. The power utilized for this design is a logic power of 0.00015W, total signal power of 0.00022W, I/O power of 0.00001W which yields to a total hierarchical power of 0.37mW.

6. COMPARISON RESULTS

The simulation results show that both exact and IMAC units yield same result and the percentage error is less than 5%. The synthesis results show that the area occupied by IMAC Unit is 15.5% less for slices and 9.5% less for 4-input LUTs when compared to Exact MAC Unit as shown in Fig 8 (a). As shown in Fig 8(b), the total power utilized by IMAC Unit is reduced by 84% when compared to the power utilized by Exact MAC Unit. The Fig 8(c) illustrates that the delay corresponding to the minimum input arrival time before clock signal measured in terms of ns is 25.1% reduced in IMAC Unit than that of in Exact MAC Unit.

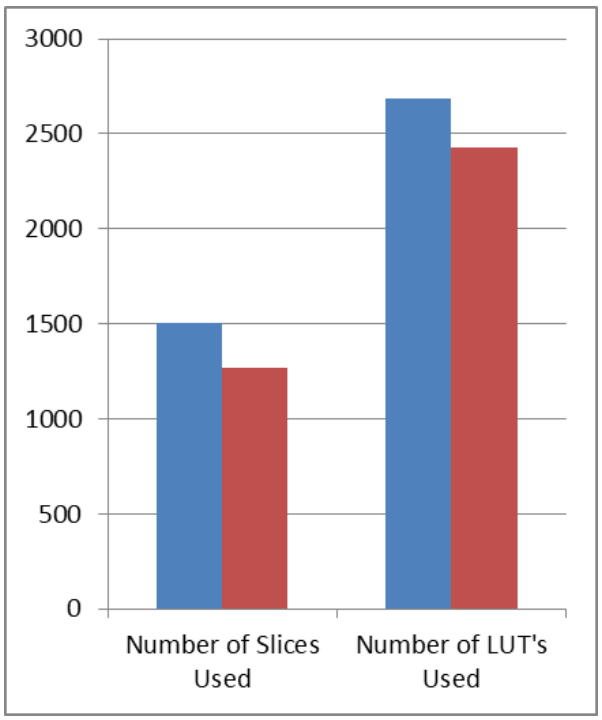

Fig 8 (a): Comparison in terms of Area

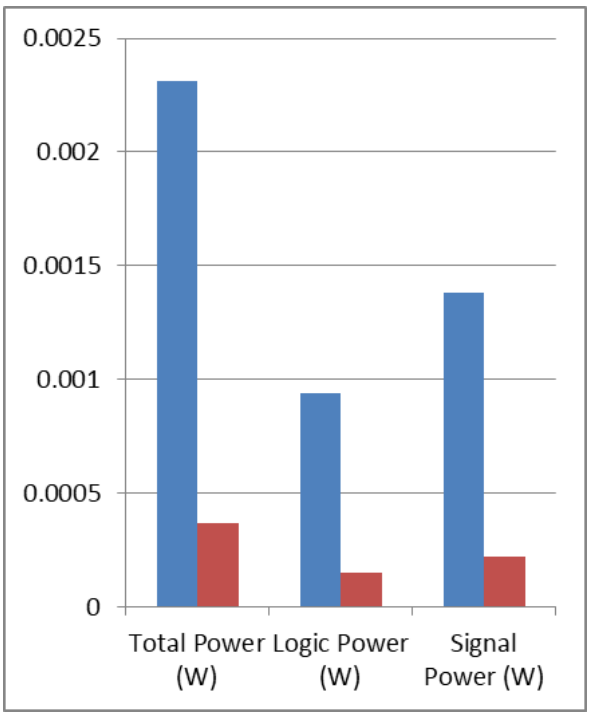

Fig 8 (b): Comparison in terms of Power

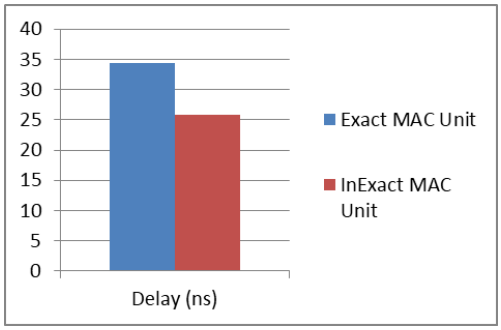

Fig 8(c): Comparison in terms of Delay

Figure 8: Comparison of several parameters of Exact MAC and In Exact MAC

The Table 2 shows the comparision of serveral parameters of Exact and In Exact MAC.

Table 2 : Comparision of several parameters of Exact MAC and In Exact MAC

	Exact MAC	Inexact MAC
Number of Slices Used	1502	1268
Number of LUT's Used	2689	2431
Total Power Utilized (W)	0.00231	0.00037
Logic Power (W)	0.00094	0.00015
Signal Power (W)	0.00138	0.00022
Delay (ns)	34.39	25.756

7. CONCLUSION

Initially there was inevitability of a stubby area utilized and High Speed DSP Processor. Hence we initially had a high area utilized system and to overcome this i..e a system which occupies the less are we introduced the concept of inexactness. We explained the concept of inexactness and explained the basic blocks of Inexact Adder. To check the functionality we are considering an example of the a MAC and comparing both exact and inexact MAC. By comparison we can finally conclude that the amount of the power utilized by the Inexact MAC is 10% when compared with Exact MAC and the area utilized by the Inexact MAC is 84% lesser than of the exact MAC. As the delay is also reduced the Inexact MAC is also fast. The outputs are also accurate enough and only less amount of the deviation, which is not a great concern. Thus we conclude that this inexact MAC will greatly be helpful in the different Digital Signal Processing applications.

8. REFERENCES

1. J. Han and M. Orshansky, "Approximate Computing: An Emerging Paradigm for Energy-Efficient Design," in ETS'13, Avignon, France, May 27-31, 2013, pp. 1 – 6.
2. H.R. Mahdiani, 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.

3. 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.
4. M. J. Schulte and E. E. Swartzlander, Jr., “Truncated multiplication with correction constant,” VLSI Signal Processing VI, pp. 388–396, 1993.
5. J. Ma, K. Man, T. Krilavicius, S. Guan, and T. Jeong, “Implementation of High Performance Multipliers Based on Approximate Compressor Design” in international Conference on Electrical and Control Technologies (ECT), 2011.
6. 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.
7. Kunal Jadhav, Aditya Vibhute ; Shyam Iyer ; R. Dhanabal “Novel Vedic mathematics based ALU using application specific reversibility”, IEEE Sponsored 9th International Conference on Intelligent Systems and Control (ISCO), 2015
8. A. Momeni, J. Han, P. Montuschi, F. Lombardi, "Design and analysis of approximate compressors for multiplication," IEEE Transactions on Computers, in press, 2014.
9. K.Y. Kyaw, W.L. Goh, K.S. Yeo, "Low-power high-speed multiplier for error-tolerant application," IEEE International Conference of Electron Devices and Solid-State Circuits (EDSSC), 2010.
10. C.-H. Lin, I.-C. Lin, "High accuracy approximate multiplier with error correction," IEEE 31st International Conference on Computer Design (ICCD), 2013.
11. C. Liu, J. Han and F. Lombardi, “A Low-Power, High-Performance Approximate Multiplier with Configurable Partial Error Recovery,” DATE 2014, Dresten, Germany, 2014.
12. Townsend, Whitney J., Earl E. Swartzlander Jr, and Jacob A. Abraham. "A comparison of Dadda and Wallace multiplier delays." In *Optical Science and Technology, SPIE's 48th Annual Meeting*. International Society for Optics and Photonics, December 2003.
13. K. Bhardwaj, P.S. Mane, J. Henkel, "Power- and area-efficient Approximate Wallace Tree Multiplier for error-resilient systems," 15th International Symposium on Quality Electronic Design (ISQED), 2014.
14. Bickerstaff, K. A. C., Michael Schulte, and E. E. Swartzlander. "Reduced area multipliers." In *International Conference on Application-Specific Array Processors, Proceedings.*, pp. 478-489. IEEE, October 1993.
15. D. Kelly, B. Phillips, S. Al-Sarawi, "Approximate signed binary integer multipliers for arithmetic data value speculation", in Proc. of the conference on design and architectures for signal and image processing, 2009
16. 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.
17. 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.
18. Teffi Francis, Tresa Joseph, Jobin K Antony, “Modified MAC unit for low power high speed DSP application using multipler with bypassing technique and optimized adders”, Fourth International conference on Computing, Communications and Networking Technologies (ICCCNT) 4-6 July 2013,pp. 1 – 4.
19. A. Abdelgawad, “Low power multiply accumulate unit (MAC) for future Wireless Sensor Networks”, IEEE conference on Sensors Applications Symposium (SAS)19-21 Feb. 2013, pp.129 – 132.
20. A. Abdelgawad ; Magdy Bayoumi, “High Speed and Area-Efficient Multiply Accumulate (MAC) Unit for Digital Signal Prossing Applications”, IEEE International Symposium on Circuits and Systems, 27-30 May 2007, pp. 3199 – 3202.