International Journal for Innovative Engineering and Management Research
A Peer Revieved Open Access International Journal
www.ijiemr.org

COPY RIGHT

ELSEVIER
SSRN

IJIEMR Transactions, online available on 5th Dec 2018. Link
:http://www.ijiemr.org/downloads.php?vol=Volume-07&issue=ISSUE-12

Title: PERFORMANCE AND ANALYSIS OF 16-BIT, 32-BIT MULTIPLY-ACCUMULATE UNITS USING VEDIC MULTIPLIER FOR DIGITAL SIGNAL PROCESSING

Volume 07, Issue 12, Pages: 946–954.

Paper Authors

MOPARTHI WILSANI ASHISH, P. NAGARAJU
KAKINADA INSTITUTE OF ENGINEERING & TECHNOLOGY-.KAKINADA

USE THIS BARCODE TO ACCESS YOUR ONLINE PAPER

To Secure Your Paper As Per UGC Guidelines We Are Providing A Electronic Bar Code

International Journal for Innovative Engineering and Management Research
PEER REVIEWED OPEN ACCESS INTERNATIONAL JOURNAL
www.ijiemr.org

PERFORMANCE AND ANALYSIS OF 16-BIT, 32-BIT MULTIPLY-ACCUMULATE UNITS USING VEDIC MULTIPLIER FOR DIGITAL SIGNAL PROCESSING

[1]MOPARTHI WILSANI ASHISH, [2]P. NAGARAJU
[1]PG SCHOLAR,KAKINADA INSTITUTE OF ENGINEERING & TECHNOLOGY-.KAKINADA
[2]ASSOCIATE PROFESSOR DEPT OF ECE. KAKINADA INSTITUTE OF ENGINEERING & TECHNOLOGY-.KAKINADA
[2]nagaraju.ece.jrg@gmail.com

ABSTRACT Most complex operations in digital signal processing involve a multiply-accumulate operation. MAC Unit performs multiplication of two numbers and stores it in a register. Almost every Processor has a MAC unit. MAC units are also used in FPGA and Certain PLCs along with other processors. But MAC Unit is one of the slowest modules present in the processors. So, there is a non-negotiable need for improving the speed of MAC Unit, which apparently enhances the performance of the processor. This thesis presents the implementation of low area, high speed Multiply-Accumulate Unit.

The main objective of this project is Multiply and Accumulate (MAC) unit design using Vedic multiplier, based onUrdhva-Tiryagbhyam Sutra. An efficient 16-bit, 32-bit Multiply-Accumulate (MAC) architecture and performance results are presented in comparison with Booth and Conventional multiplier architectures and also compared with respect to different adders.The 32-bit Multiply and Accumulate (MAC) unit reduces the area by reducing the number of multiplication and addition in the multiplier unit. Increase in the speed of operation is achieved by the hierarchical nature of the Vedic multiplier unit. The Multiply-Accumulate unit designed with Vedic multiplier and Carry skip adder has less delay of 26.102 ns. The modules have been designed in VHDL, simulated and synthesized using Xilinx 14.7.The efficiency in terms of device utilization (area) and speed of 16-bit, 32-bit Multiply and Accumulate (MAC) unit architecture is observed through reduced area, low critical delay and low hardware complexity.

1. INTRODUCTION

1.1 MOTIVATION AND OVERVIEW

The general Multiply-Accumulate (MAC) architecture consists of a conventional multiplier, adder and an accumulator. Where the output is added to the previous MAC output result by an accumulate adder. The Multiply-Accumulate (MAC) unit is extensively used in microprocessors and digital signal processors for data-intensive applications, such as filtering,

convolution, and inner products. Most digital signal processing methods use nonlinear functions such as discrete cosine transform (DCT) or discrete wavelet transform (DWT) or FFT/IFFT computations that can be efficiently accelerated by dedicated MAC units. Because they are basically accomplished by repetitive application of multiplication and addition, the speed of the multiplication and addition determines the execution speed and performance of the entire computation. As the multiplier exhibits inherently long delay among the basic operational blocks in digital system, the multiplier determines the critical path.The demand for high speed processing has been increasing as a result of expanding computer and signal processing applications. Higher throughput arithmetic operations are important to achieve the desired performance in many real-time signal and image processing applications. One of the key arithmetic operations in such applications is multiplication and the development of fast multiplier circuit has been a subject of interest over decades. Reducing the time delay and power consumption are very essential requirements for many applications. This work presents different multiplier architectures. Multiplier based on Vedic Mathematics is one of the fast and low power multiplier**.**In order to improve the speed of the MAC unit, there are two major bottlenecks. The first is the partial products reduction network that is used in the multiplication block and the second is the accumulator. Both of these stages require addition of large operands that involve long paths for carry propagation. The main key to the proposed architecture is using the Vedic multiplier to design the MAC unit and compare the performance with the conventional MAC units in terms of area, speed and number of resources. It is well known fact that the speed of MAC is governed by the speed of the multiplier. The Vedic multiplier uses "Urdhva-Tiryagbhyam" algorithm. Urdhva-Tiryakbhyam Sutra is first applied to the binary number system and is used to develop digital multiplier architecture. This is shown to be very similar to the popular array multiplier architecture. This Sutra also shows the effectiveness of to reduce the NXN multiplier structure into an efficient 4X4 multiplier structures. The proposed multiplication algorithm is then illustrated to show its computational efficiency by taking an example of reducing a 4X4-bit multiplication to a single 2X2-bit multiplication operation. This work presents a systematic design methodology for fast and area efficient digit multiplier based on Vedic mathematics .The Multiplier Architecture is based on the Vertical and Crosswise algorithm of ancient Indian Vedic Mathematics.

International Journal for Innovative Engineering and Management Research
PEER REVIEWED OPEN ACCESS INTERNATIONAL JOURNAL
www.ijiemr.org

1.2 PROBLEM DEFINITION

To study different algorithms for the designing of Adder circuits for the implementation of MAC unit.

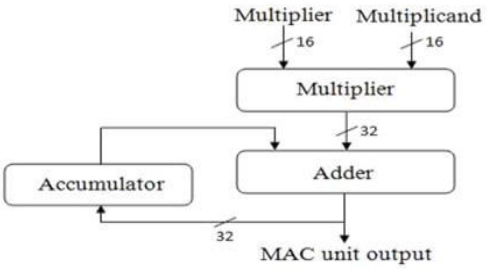

Figure 1.1 Block diagram of MAC Unit

- To study different algorithms for the designing of Multiplier circuits for the implementation of MAC unit.
- To study the performance of the different Multiply-Accumulate unit by combing different Multiplier and Adder circuits.
- To compare the performance results of the different MAC unit by combing Multi-plier and Adder circuits this has lower delay and area.
- To do FPGA implementation of the obtained MAC unit.

1.3 THESIS ORGANISATION

In chapter 2 the literature survey on various Multiplier, Adder circuits, and MAC units is presented. In chapter 3 the algorithms and implementation of various Adders circuits which are used in the design of MAC unit are discussed. The algorithms and implementation of different Multiplier circuits which are used in the design of MAC unit are discussed in chapter 4. In chapter 5 the tools that are used to implement the MAC unit are explained.

Schematic and Simulation results of multipliers, adders and multiply-accumulate units are detailed in chapter 6 and the comparison of different MAC units with respect to area and speed are studied. The thesis is concluded with chapter 7, where the conclusions from entire work done in this project are presented.

2. VEDIC MULTIPLICATION

The proposed Vedic multiplier is based on the Vedic multiplication formulae (Sutras). These Sutras have been traditionally used for the multiplication of two numbers in the decimal number system. In this work, we apply the same ideas to the binary number system to make the proposed algorithm compatible with the digital hardware. Vedic multiplication based on some algorithms, is discussed below:

Urdhva Tiryakbhyam sutra

The multiplier is based on an algorithm Urdhva Tiryakbhyam (Vertical & Crosswise) of ancient Indian Vedic Mathematics. Urdhva Tiryakbhyam Sutra is a general multiplication formula applicable to all cases of multiplication. It literally means “Vertically and crosswise”. It is based on a novel concept through which the generation of all partial products can be done and then, concurrent addition of these partial products can be done. Thus parallelism in generation of partial products and their summation is obtained using Urdhava Tiryakbhyam. The algorithm can be generalized for n x n bit number.

International Journal for Innovative Engineering and Management Research
PEER REVIEWED OPEN ACCESS INTERNATIONAL JOURNAL
www.ijiemr.org

Since the partial products and their sums are calculated in parallel, the multiplier is independent of the clock frequency of the processor. Thus the multiplier will require the same amount of time to calculate the product and hence is independent of the clock frequency. The net advantage is that it reduces the need of microprocessors to operate at increasingly high clock frequencies. While a higher clock frequency generally results in increased processing power, its disadvantage is that it also increases power dissipation which results in higher device operating temperatures. By adopting the Vedic multiplier, microprocessors designers can easily circumvent these problems to avoid catastrophic device failures. The processing power of multiplier can easily be increased by increasing the input and output data bus widths since it has a quite a regular structure. Due to its regular structure, it can be easily layout in a silicon chip. The Multiplier has the advantage that as the number of bits increases, gate delay and area increases very slowly as compared to other multipliers. Therefore it is time, space and power efficient. It is demonstrated that this architecture is quite efficient in terms of silicon area/speed [3,5].

Multiplication of two decimal numbers- 43*68

To illustrate this multiplication scheme, let us consider the multiplication of two decimal numbers (43*68). The digits on the both sides of the line are multiplied and added with the carry from the previous step. This generates one digit of result and a carry digit. This carry is added in the next step and hence the process goes on. If more than one line are there in one step, all the results are added to the previous carry. In each step, unit's place digit acts as the result bit while the higher digits act as carry for the next step. Initially the carry is taken to be zero. The working of this algorithm has been illustrated in

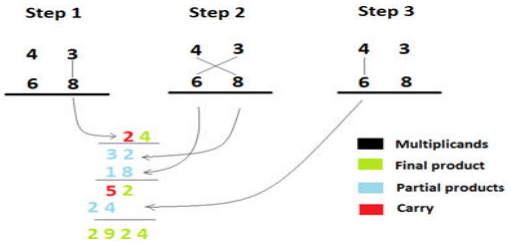

Fig 2.2 Multiplication of 2 digit decimal numbers using Urdhva Tiryakbhyam Sutra

Now we will see how this algorithm can be used for binary numbers. For example (1101 * 1010) as shown in Fig 2.3.

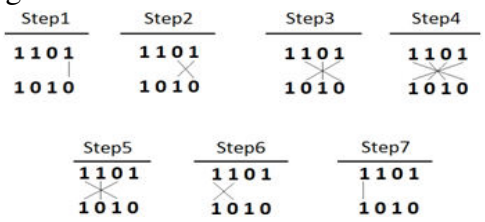

Fig 2.3 Using Urdhva Tiryakbham for Binary numbers

Firstly, least significant bits are multiplied which gives the least significant bit of the product (vertical). Then, the LSB of the multiplicand is

International Journal for Innovative Engineering and Management Research
PEER REVIEWED OPEN ACCESS INTERNATIONAL JOURNAL
www.ijiemr.org

multiplied with the next higher bit of the multiplier and added with the product of LSB of multiplier and next higher bit of the multiplicand (crosswise). The sum gives second bit of the product and the carry is added in the output of next stage sum obtained by the crosswise and vertical multiplication and addition of three bits of the two numbers from least significant position. Next, all the four bits are processed with crosswise multiplication and addition to give the sum and carry. The sum is the corresponding bit of the product and the carry is again added to the next stage multiplication and addition of three bits except the LSB. The same operation continues until the multiplication of the two MSBs to give the MSB of the product. For example, if in some intermediate step, we get 110, then 0 will act as result and 11 as the carry. It should be clearly noted that carry may be a multi-bit number.

16x16 bit Multiplier

The 16x16 Multiplier is made by using 4 , 8x8 multiplier blocks. Here , the multiplicands are of bit size (n=16) where as the result is of 32 bit size. The input is broken into smaller chunks size of n/2 = 8, for both inputs, that is a and b. These newly formed chunks of 8 bits are given as input to 8x8 multiplier block, where again these new chunks are broken into even smaller chunks of size n/4 = 4 and fed to 4x4 multiply block, just as in case of 8x8 Multiply block. Again, the new chunks are divided in half, to get chunks of size 2, which are fed to 2x2 multiply block. The result produced, from output of 8x8 bit multiply block which is of 16 bits, are sent for addition to an addition tree , as shown in the Fig 3.7.

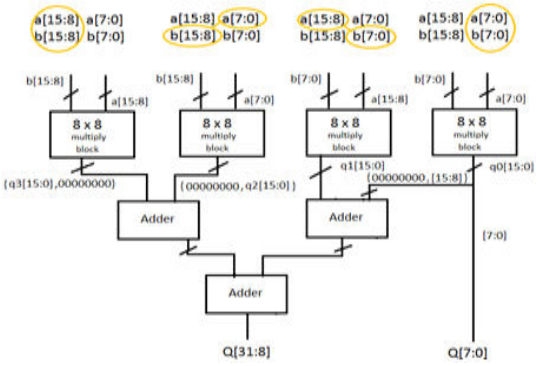

Fig 3.7Block diagram of 16x16 Multiply block

Here, as shown in Fig 16, the lower 8 bits of q0 directly pass on to the result, while the higher bits are fed for addition into the addition tree. The addition of partial products is shown in Fig 3.8.

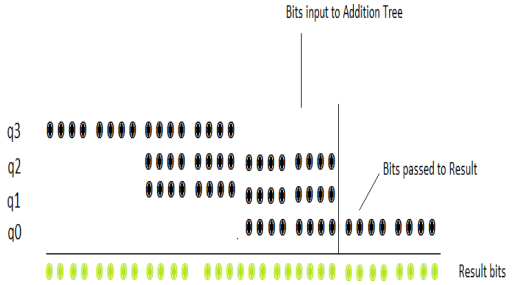

Fig 3.8 Addition of Partial products in 16x16 block

3.2 Booth Multiplier

The Booth multiplier is also known as Recoded booth multiplier, in which the multiplicand is kept as it is and the multiplier is recoded as a recoded multiplier and then the multiplication is done with multiplicand

International Journal for Innovative Engineering and Management Research
PEER REVIEWED OPEN ACCESS INTERNATIONAL JOURNAL
www.ijiemr.org

and recoded multiplier. To reduce the number of partial products in the multiplier, the Multiplier uses Radix 2*r* multipliers, which produces *N/r* partial products, each of which depends on *r* bits of the multiplier. Fewer partial products lead to a smaller and faster CSA (Carry Save Adder) array. For example, a radix-4 multiplier produces N/2 partial products. Each radix-4 multiplier produces N/2 partial products. Each partial product is 0, Y, 2Y, or 3Y,depending on a pair of bits of X. Computing 2Y is a simple shift, but 3Y is a hard multiple requiring a slow carry-propagate addition of Y + 2Y before partial product generation begins. Higher-radix Booth encoding is possible, but generating the other hard multiples appears not to be worthwhile for multipliers of fewer than 32 bits. The figure 2 shows the basic architecture of Booth multiplier.

3.3 Conventional Multiplier

The conventional multiplier of width N x N bits will generate the N number of partial products. The partial products are generated by bit wise ANDing one multiplier bit with another multiplier. Hence, the N x N bit multiplier uses 2N-multiplications and N-Adders in the architecture of Conventional multiplier. Fig. 4 below shows the basic architecture of conventional multiplier. The Multiplications and Additions are significantly reduced in case of Vedic Multiplier compared to Booth and conventional multiplier. Table 1 gives details about the Hardware resources used i.e., Multiplications and Additions in case of Vedic, Booth and conventional multiplier for 8x8-bit, 16x16-bit & 32x32-bit.

3. SIMULATION RESULTS

3.1 32-Bit MAC Unit using Booth Multiplier

RTL Schematic:

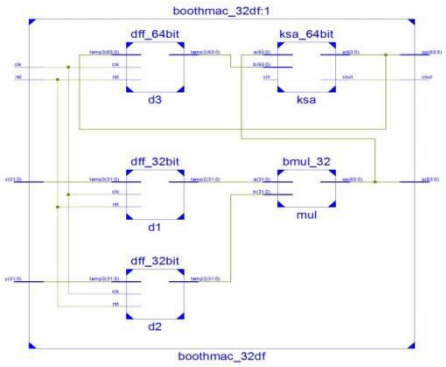

Fig 3.1 RTL Schematic of 32-bit MAC unit using Booth multiplier and Kogg-stone adder

SIMULATION WAVEFORM:

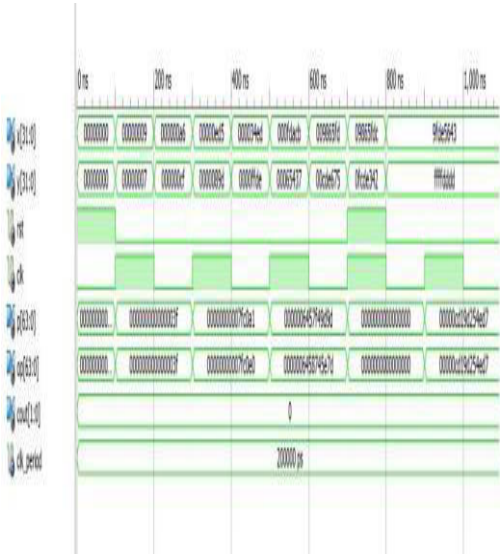

Fig 3.1.1 Simulation Waveform of 32-bit MAC unit using Booth multiplier and Kogg-stone adder

International Journal for Innovative Engineering and Management Research
PEER REVIEWED OPEN ACCESS INTERNATIONAL JOURNAL
www.ijiemr.org

DEVICE UTILIZATION SUMMARY:

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slice Registers	205	54576	0%
Number of Slice LUTs	5711	27288	20%
Number of fully used LUT-FF pairs	64	5852	1%
Number of bonded IOBs	195	218	89%
Number of BUFG/BUFGCTRLs	1	16	6%

Fig 3.2 Device Utilization Summary of 32-bit MAC unit using Booth multiplier and Kogg-stone adder

TIMING REPORT:

Logic Delay – 16.111 ns
Routing Delay- 57.412 ns
Total Delay- 73.524ns

3.2 32-Bit MAC Unit using Array Multiplier

RTL Schematic:

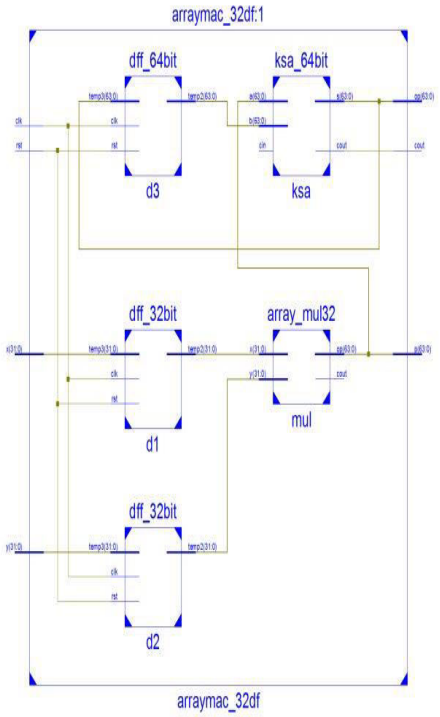

Fig 3.3 RTL Schematic of 32-bit MAC unit using Array multiplier and Kogg-stone adder

SIMULATION WAVEFORM:

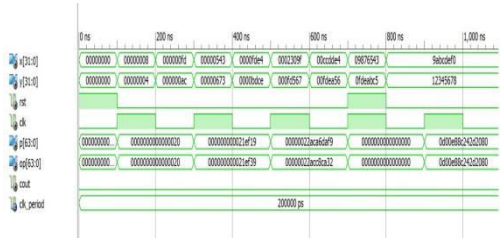

Fig 3.4 Simulation Waveform of 32-bit MAC unit using Array multiplier and Kogg-stone adder

DEVICE UTILIZATION SUMMARY:

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slice Registers	250	54576	0%
Number of Slice LUTs	2927	27288	10%
Number of fully used LUT-FF pairs	63	3114	2%
Number of bonded IOBs	195	218	89%
Number of BUFG/BUFGCTRLs	1	16	6%

Fig 3.5 Device Utilization Summary of 32-bit MAC unit using Array multiplier and Kogg-stone adder

TIMING REPORT:

Logic Delay – 6.821 ns
Routing Delay- 23.640 ns
Total Delay- 30.461 ns

3.3 DELAY AND UTILIZATION COMPARISION:

Inferences:

- Vedic Multiply-Accumulate has low logic delay, routing delay and total delay also, and Booth Multiplier has high delay among these three Multiply-Accumulate units using Kogg-Stone adder.
- Array Multiply-Accumulate unit has less no of Slice LUTs are utilised and Booth Multiply-Accumulate unit has more no of Slice LUTs are utilised i.e double the number of Array multiplier.

International Journal for Innovative Engineering and Management Research
PEER REVIEWED OPEN ACCESS INTERNATIONAL JOURNAL
www.ijiemr.org

3.4 OVERALL COMPARISION 32-BIT MAC UNITS

Inferences:

- Vedic Multiply-Accumulate unit using with Carry skip adder has very high speed among all the nine combinations of MAC Units.
- Booth Multiply-Accumulate Unit using Kogg-Stone Adder has high delay among these Nine Multiply-Accumulate units with different adders and multipliers.
- Array Multiply-Accumulate unit using Carry Skip Adder has less no of Slice LUTs are utilised and Booth Multiply-Accumulate unit using Carry Save Adder has more no of Slice LUTs are utilised among all the combinations of MAC Units using different multipliers and adders.

Table 6.10 Overall 32-Bit MAC Units synthesis results comparison

Adders	Multipliers	No of Slice LUTs	Logic Delay (ns)	Routing Delay (ns)	Total Delay (ns)
1.Carry Skip Adder	Vedic	3609	5.798	20.104	26.102
	Booth	4461	17.283	50.644	67.927
	Array	2822	6.878	23.996	30.874
2.Carry Save Add er	Vedic	4031	7.503	25.829	33.332
	Booth	5762	13.651	55.299	68.950
	Array	3105	8.469	27.989	36.455
3.Kogg-stone Adder	Vedic	3874	5.859	20.982	26.841
	Booth	5711	16.111	57.412	73.524
	Array	2927	6.821	23.640	30.461

CONCLUSION AND FUTURE SCOPE

In this work 3-different types of 32-bit Multipliers and 3 different types 32-bit Adders are implemented. By using different combinations of Multipliers and Adders, 9-different types of 32-bit Multiply-Accumulate units are designed. The combination with Vedic Multiplier and Carry-skip Adder combination has the delay of 26.102 nS for both logic and routing delay. So here can be say that the UrdhvaTiryagbhyam sutra with 32-bit Vedic Multiplier is the best in the aspect of delay because of this vertical and crosswise sutra no of partial products required for multiplication are reduced.The combination 32-bit Array multiplier with Carry skip adder has used less no of Slice LUTs used for the implementation i.e of 2822 slice LUTs are used. In this design due to using of multi block concept it has required lesser area compared to remaining Multiply-Accumulate units. The design complexity and delay are reduced due to use of the Vedic Multipliers in the linear filtering, correlation, spectrum analysis. This

Multiply-Accumulate Unit can be extended higher bit MAC unit such as full precision floating point MAC unit or double precision floating point MAC unit. The designed MAC unit may be used to design VLSI architectures for digital signal processing such as convolution, Fast Fourier Transform (FFT), Discrete Cosine Transform (DCT), Discrete Wavelet Transform (DWT), Adaptive Signal Processing.

REFERENCES

[1] Bisoyi, Baral, Senapati“ Comparision of a 32-bit Vedic Multiplier with a Conventional Binary multiplier. 2014 IEEE International Conference on Advanced Communication Control and Computing Technologies.

[2] Yeh, W.C. and C.W. Jen., 2003.\High-speed and low-power split-radix FFT," IEEE Trans. Sig. Process., 51:pp.864-874.

[3] Matsui, M, H. Hara, Y. Uetani, L.S. Kim and T. Nagamatsu et al., 1994.\A 200 MHz 13 mm2 2-D DCT macrocell using sense-amplifying pipeline ip- op scheme," IEEE J. Solid-State Circ., 29:pp.1482-1490.

[4] Grossschadl, J. and G.A. Kamendje, 2003. \A single-cycle (32x 32+ 32+ 64)-bit multiply/accumulate unit for digital signal processing and public-key cryptog-raphy, Proceedings of the 10th International Conference on Electronics, Circuits and Systems.,pp.254-257

[5] Clark. L.T, E.J. Ho man, J. Miller, M. Biyani and Y. Liao et al.,2001. \An embedded 32-b microprocessor core for low-power and high-performance applica-tions, ", in : IEEE J. Solid-State Circ.,36:pp.1599-1608.