INTERNATIONAL JOURNAL OF RESEARCH
In Advanced Engineering Technologies

Volume 3, Issue 1 OCT 2014

DESIGN AND IMPLEMENTATION OF MODULO ADDER USING VERILOG HDL IN FPGA TECHNOLOGY

[1] **R Rajashekar**, **PG Scholar in VLSI,**
[2]**Burra Ayyappa Swamy, M.Tech, Assoc. Professor, ECE Department**,
[1] Rajaece463@gmail.com,
[2] baswamy.mits@gmail.com,
[1,2]Madhira Institute of Technology & Sciences, Kodad, Nalgonda.

ABSTRACT:

Modulo of the form 2n ± 1, which greatly simplify certain arithmetic operations in residue number systems (RNS), have been of longstanding interest. A steady stream of designs for modulo-(2n
± 1) adders have rendered the latency of such adders quite competitive with ordinary adders. The next logical step is to approach the problem in a unified and systematic manner that does not require each design to be taken up from scratch and to undergo the error-prone and labor-intensive optimization for high speed and low power dissipation. Accordingly, we devise a new redundant representation of mod-(2n ± 1) residues that allows ordinary fast adders and a small amount of peripheral logic to be used for mod-(2n ± 1) addition. Advantages of the building-block approach include shorter design time, easier exploration of the design space (area/speed/power tradeoffs), and greater confidence in the correctness of the resulting circuits. Advantages of the unified design include the possibility of fault-tolerant and gracefully degrading RNS circuit realizations with fairly low hardware redundancy.

Keywords— Modulo adder, parallel-prefix computation,VLSI design.

INTRODUCTION

The modulo 2n+1 adder has the applications in many fields, say pseudorandom number generation, cryptography, convolution computations without round-off errors. It has the applications in residue number system (RNS) also. The RNS is an arithmetic system which decomposes a number into parts (residues) and performs arithmetic operations in parallel for each residue without the need of carry propagation between them, which leads to significant speed-up over the corresponding binary operations. RNS is well suited to applications that are rich of addition/subtraction and multiplication operations and has been adopted in the design of digital signal processors, FIR filters and communication components, offering in several cases apart from enhanced operation speed and low power characteristics.

There are three input representations chosen for the input operands namely, the normal weighted one, the diminished-1 and the signed-LSB representations. But, only the first two representations in the following are considered, since the adoption of the signed-LSB representation does not lead to more efficient circuits in delay or area terms. The input operands and results are limited between 0 and

2n when performing arithmetic operations modulo 2n + 1.

II. EXISTING SYSTEM

We propose improved area-efficient weighted modulo 2n + 1 adder design using diminished-1 adders with simple correction schemes. This is achieved by subtracting the sum of two (n + 1)-bit input numbers by the constant 2n + 1 and producing carry and sum vectors. The modulo 2n + 1 addition can then be performed using parallel-prefix structure diminished-1 adders by taking in the sum and carry vectors plus the inverted end -around carry with simple correction schemes. The area cost for our proposed adders is lower. In addition, our proposed adders do not require the hardware for zero detection that is needed in diminished-1 modulo 2n + 1 addition.

Exploiting Residue Number System for Power-Efficient Digital Signal Processing in Embedded Processors:

Residue Number System (RNS) has long been touted as a solution for application Digital Signal Processing (DSP) hardware. In this application the inherent parallelism of RNS has been well-known for simultaneously yielding both high-performance and modest power consumption. However, the limitations of its expressiveness in terms of arithmetic operations, together with overheads related to interaction with 2's complement arithmetic, makes programmable processor design that takes advantage of these benefits quite challenging. In this paper we meet this challenge by multi-tier synergistic co-design of compilation techniques, architecture, micro-architecture, as well as hardware components. The net result is an RNS extension to a RISC Processor that others remarkable performance and power improvements simultaneously. With efficientautomatic code generation, we have achieved an average performance improvement of 21% and 51% reduction in functional unit power consumption. For the time, we have exposed application programmers to the substantial benefits of RNS-supported embedded DSP.

III.Modulo-($2^n \pm 1$) Adders

The complexity of a modulo 2n+1 arithmetic unit is determined by the representation chosen for the input operands. Three representations have been considered namely, the normal weighted one, the diminished and the signed-LSB representations. We only consider the first two representations in the following, since the adoption of the signed-LSB representation does not lead to more efficient circuits in delay or area terms. In every case, when performing arithmetic operations modulo 2n+1 the input operands and the results are limited between 0 and 2n.In the normal weighted representation, each operand requires n+1 bits for its representation but only utilizes 2n+1 representations out of the 2n+1 that these can provide. A denser encoding of the input operands and simplified arithmetic operations modulo 2n+1 are offered by the diminished-1 representation. In the diminished-1 representation, A is represented as as A where az is a single bit, often called the zero indication bit, and A is an n-bit vector, often called the number part.

IV. METHODOLOGIES

The key parameters of high-speed digital circuits are the propagation delay and power consumption. The maximum operating frequency of a digital circuit is calculated.

F max = 1/(tp LH + tp HL) …… 1

Where tp LH and tp HL are the propagation delays of the low-to-high and high-to-low transitions of the gates, respectively. The total power consumption of the CMOS digital circuits is determined by the switching and short circuit power. The switching power is linearly proportional to the operating frequency and is given by the sum of switching power at each output node as in

Switching= $\sum_{i=1}^{n}$ fclkCLiVdd2 …… 2

Where n is the number of switching nodes, Fclk is the clock frequency, Clip is the load capacitance at the output node of the ith stage, and Add is the supply voltage. Normally, the short-circuit power occurs in dynamic circuits when there exists direct paths from the supply to ground which is given by

P sc = I sc * V dd …… 3

Where Isc is the short-circuit current. The analysis shows that the short-circuit power is much higher in E-TSPC

logic circuits than n TSPC logic circuits. However, TSPC logic circuits exhibit higher switching power compared to that of E-TSPC logic circuits due to high load capacitance. For the E-TSPC logic circuit, the short circuit power is the major problem. The E-TSPC circuit has the merit of higher operating frequency than that of the TSPC circuit due to the reduction in load capacitance, but it consumes significantly more power than the TSPC circuit does for a given transistor size. The following analysis s based on the latest design using the popular and low-cost 0.18-microm CMOS process.

V. PERFORMANCE ONDELAY

The time taken to charge and discharge the load capacitance CL determines the switching speed of the CMOS gate. Rise time is defined during charging time from 10 % to 90% of its steady-state value; this is the same as the fall time. Delay time is defined by the time difference between 50% of charging time and 50% of discharging time. From the equations, in order to improve the individual gate delays, the load impedance CL is reduced or the current gain of the transistors is increased. Increasing the current gain means higher β, approximately equal to the W/L of the transistor. Therefore, by increasing β, the transistor size will increase, thus affecting the size of the chip.

Modules Name:

- Preprocessing unit
- Carry computation unit
- CS block
- Carry prefix operator
- Carry-select block
- Diminished-1modulo operation
- New sparse modulo 2n+ 1 adder

VI. MODULES EXPLANATION AND DIAGRAM:

Preprocessing unit:

The preprocessing stage computes the carry-generate bits Gi, the carry-propagate bits Pi, and the halfsum bits Hi, for every i , $0 \le i \le n-1$. Where . , +, and ⊕ denote logical AND, OR, and exclusive-OR, respectively. Carry generation bits =>Gi = Ai . Bi

Carry propagate bits => Pi = Ai + Bi

Half sum bits => Hi = Ai ⊕ Bi

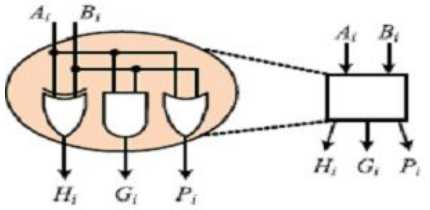


Fig.1. Preprocessing Unit

Carry computation unit:

The second stage of the adder, hereafter called the carry computation unit, computes the carry signals

Ci, for $0 \le i \le n - 1$ using the carry generate and carry propagate bits Gi and Pi. The third stage computes the sum bits according to

Si = Hi ⊕Ci -1

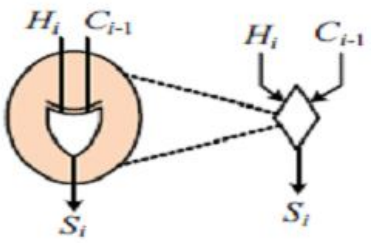


Fig.2. Carry computation unit

Carry computation is transformed into a parallel prefix problem using the "o" operator, which associates pairs of generate and propagate signals and was defined as

(G,P)o(G',P')=(G+P.G',P.P')

In a series of associations of consecutive generate and propagate pairs (G; P) can be represented as

(Gk:j: Pk:j) = (Gk ,Pk)o(Gk-1,Pk-1)o….o(Gj , Pj)

Where k>j Since every carry Ci = Gi:0, a number of algorithms have been introduced for computing all the carries using only "o" operators.

Carry-Select Block:

The design of sparse adders relies on the use of a sparse parallel-prefix carry computation unit and carry-select (CS) blocks. Only the carries at the boundaries of the carry-select blocks are computed, saving considerable amount of area in the carry-computation unit. The carry select block computes two sets of sum bits corresponding to the two possible values of the incoming carry. When the actual carry is computed, it selects the correct sum without any delay overhead.

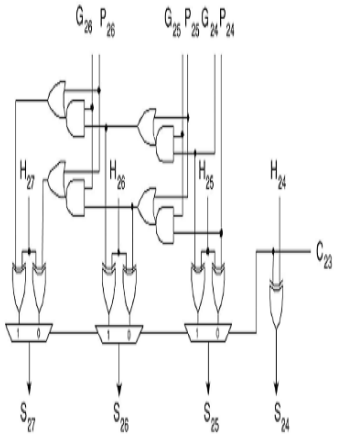


Fig.3. Logic-level implementation of a 4-bit carry-select block

Parallel Prefix Adder:

Suppose that A= An-1An-2….A0and B= Bn-1Bn-2….B0 represent the two numbers to be added and S= Sn-1Sn-2….S0 denotes their sum. An adder can be considered as a threestage circuit. The preprocessing stage computes the carrygenerate bits Gi, the carry-propagate bits Pi, and half-sum bits H, for every i, 0≤i≤n-1, according to

Gi = Ai. Bi Pi = Ai+ Bi Hi = Ai ⊕ Bi

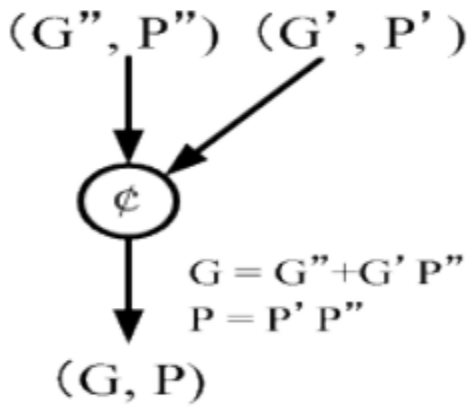


Figure 4. Carry operator

A parallel prefix adder can be represented as a parallel prefix graph consisting of carry operator nodes. Figure 5 is the parallel prefix graph of a Ladner-Fischer adder. This adder structure has minimum logic depth, but has large fan-out requirement up to n/2.

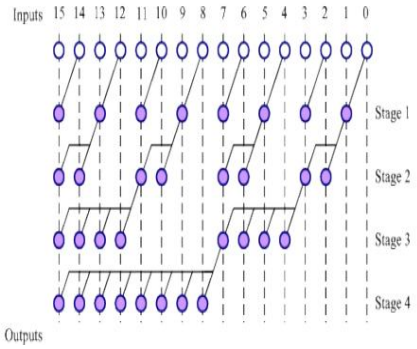


Fig 5. 16-bit Ladner-Fischer adder

Diminished-1 Modulo Operation (Sparse Carry Computation):

The sparse version of the parallel-prefix adders introduced in this paper alleviates a lot the regularity and the area-overhead problem, as it can be verified from Fig 6, there is still a lot of space for improvement. In the following, we attack this problem by introducing a new prefix operator and an even simpler parallel-prefix carry computation unit. The new technique will be presented via an example. Let us consider the design of a sparse-4 diminished-1 modulo 2^{16}+1 adder. In this case, we need a carry computation unit that implements the following prefix equations:

$$C_{15}^{+} \leftrightarrow \overline{(G_{15:0}, P_{15:0})},$$
$$C_{11}^{+} \leftrightarrow (G_{11:0}, P_{11:0}) \circ \overline{(G_{15:12}, P_{15:12})},$$
$$C_{7}^{+} \leftrightarrow (G_{7:0}, P_{7:0}) \circ \overline{(G_{15:8}, P_{15:8})},$$
$$C_{3}^{+} \leftrightarrow (G_{3:0}, P_{3:0}) \circ \overline{(G_{15:4}, P_{15:4})}.$$

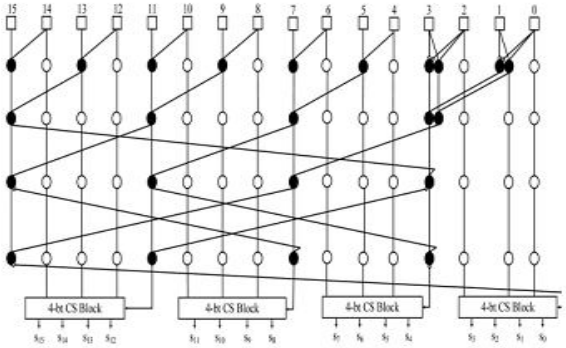


Fig 6 Modulo 2^{16}+1 diminished adders using a sparse carry computation unit.

New Sparse Modulo 2^n+ 1 Adder:

Fig.7presents the resulting architecture for a diminished-1 modulo 2^{16} +1 adder, in which two gray operators are used. The top one which resides at prefix level 3, This operator is used to compute (G3:0; P3:0) o(G15:12,P15:12), which is necessary for the computation of both **C3**+and C11+.

Its vertical successor is also replaced by a gray operator that computes the final

$$C_{3}^{+} \leftrightarrow (G_{3:0}, P_{3:0}) \circ \overline{(G_{15:12}, P_{15:12}) \circ (G_{11:4}, P_{11:4})}.$$

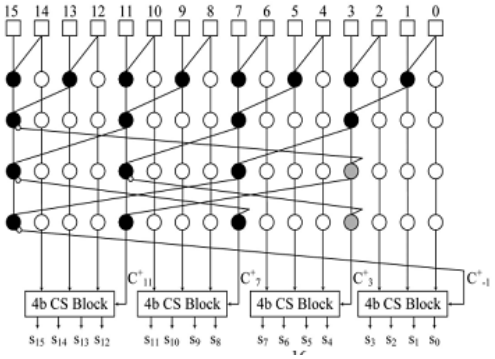
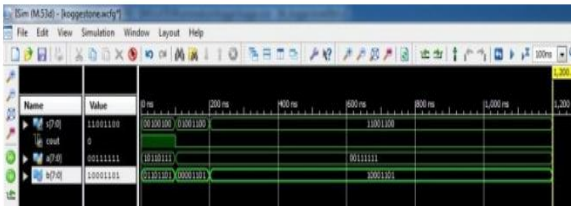


Fig. 7.Proposed sparse-4 modulo 2^{16}+ 1 diminished-1 adder

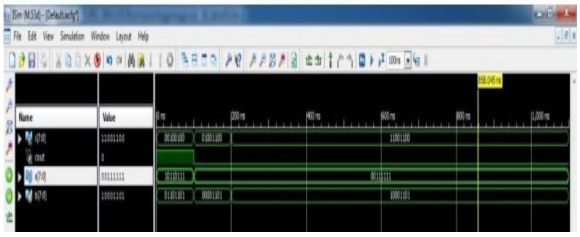
VII.SIMULATION RESULTS:

The Verilog coding is performed to the design and it is then implemented in XILINX ISE 10.1. The implementation is performed with 8-bit input. The experimental results for the parameters namely power, delay, frequency and LUT count obtained for modulo 2^8+1 design are obtained. For the parallel addition operation, three stages are used. Thus the first stage (pre processing stage) are modified has less amount of power consumption compared to the earlier method.

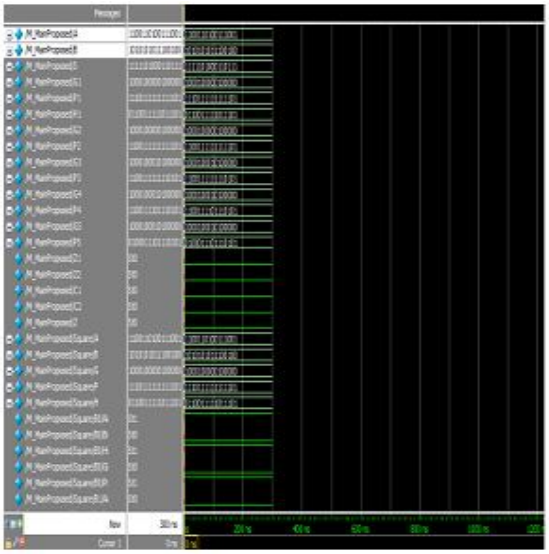
Kogge-Stone adder:



Lander Fisher:



New Sparse Modulo 2^n+ 1 Adder:



XIII. CONCLUSIONS

Efficient modulo 2^n+1 adder are appreciated in a variety of computer applications including all RNS implementations. In this paper, two contributions are offered to the modulo 2^n+1 addition problem. A novel architecture has been proposed that uses a sparse totally regular parallel-prefix carry computation unit. This architecture was derived by proving the inverted circular idem potency property of the parallel -prefix carry operator in modulo ^{2n}+1 addition and by introducing a new prefix operator that eliminates the need for a double computation tree in the earlier fastest proposals. The experimental results indicate that the proposed architecture heavily outperforms the earlier solutions in implementation area and power consumption, while offering a high execution rate. The modulo 2^n+1 addition problem was also shown to be related to the modulo 2^n+1 addition problem. The unifying theory presented in this paper revealed that a simple post processing stage composed of an XOR gate for each output bit needs to be added to a modulo 2^n+1 adder for attaining a modulo 2^n+1 added. As a result, every architecture that has been and more importantly that will be proposed for designing modulo 2^n+1 adders, can be reused for the design of modulo 2^n+1 adders.

REFERENCES

[1] X. Lai and J.L. Massey, "A Proposal for a New Block Encryption Standard,"EUROCRYPT,D.W. Davies, ed., vol. 547, pp. 389-404,Springer, 1991.

[2] R. Zimmermann et al., "A 177 Mb/s VLSI Implementation of theInternational Data Encryption Algorithm," IEEE J. Solid-StateCircuits,vol. 29, no. 3, pp. 303-307, Mar. 1994.

[3] H. Nozaki et al., "Implementation of RSA Algorithm Based onRNS Montgomery Multiplication," Proc. Third Int'l Workshop Cryptographic Hardware and Embedded Systems,pp. 364-376, 2001.

[4] Y. Morikawa, H. Hamada, and K. Nagayasu, "Hardware Realisation of High Speed Butterfly for the Maximal Length Fermat Number Transform,"Trans. IECE,vol. J66-D, no. 1, pp. 81-88,1983.

[5] M. Benaissa, S.S. Dlay, and A.G.J. Holt, "CMOS VLSI Design of a High-Speed Fermat Number Transform Based Convolver/Correlator Using Three-Input Adders," Proc. IEE, vol. 138, no. 2,pp. 182-190, Apr. 1991.

[6] V.K. Zadiraka and E.A. Melekhina,

"Computer Implementation of Efficient Discrete-Convolution Algorithms, "Cybernetics and Systems Analysis,vol. 30, no. 1, pp. 106-114, Jan. 1994.

[7] M.A. Soderstrand et al.,Residue Number System Arithmetic: ModernApplications in Digital Signal Processing.IEEE Press, 1986.

[8] P.V.A. Mohan,Residue Number Systems: Algorithms and Architectures.Springer-Verlag, 2002.

[9] A. Omondi and B. Premkumar,Residue Number Systems: Theoryand Implementations.Imperial College Press, 2007.

[10] J. Ramirez et al., "RNS-Enabled Digital Signal Processor Design,"Electronics Letters,vol. 38, no. 6, pp. 266-268, Mar. 2002.