Implementation of 64-Bit Multiplier Unit using Modified Wallace Structure

SADHU HIMA BINDU¹, GANESH KUMAR. P²

¹PG Scholar, Kakinada Institute of Engineering & Technology, Korangi, AP, India, Email: binduhima485@gmail.com.
²Asst Prof, Kakinada Institute of Engineering & Technology, Korangi, AP, India, Email: pganeshkumar66@gmail.com.

Abstract: A multiplier is one of the key hardware blocks in most digital and high performance systems such as FIR filters, micro processors and digital signal processors etc. A system's performance is generally determined by the performance of the multiplier because the multiplier is generally the slowest element in the whole system and also it is occupying more area consuming. The Carry Select Adder (CSLA) provides a good compromise between cost and performance in carry propagation adder design. A Carry Select Adder is introduced but it offers some speed penalty. However, conventional CPA is still area-consuming due to the dual ripple carry adder structure. In the proposed work, generally in Wallace multiplier the partial products are reduced as soon as possible and the final carry propagation path carry select adder is used. In this paper, modification is done at gate level to reduce area and power consumption. The Modified Carry Select-Adder (MCSA) is designed using Common Boolean Logic and then compared with regular CSA respective architectures, and this MCSA is implemented in Wallace Tree Multiplier. This work gives the reduced area compared to normal Wallace tree multiplier. Finally an area efficient Wallace tree multiplier is designed using common Boolean logic based carry select adder.

Keywords: Carry Select Adder, Modified Wallace Tree Multiplier, Xilinx ISE, Verilog.

I. INTRODUCTION

Multi-input addition is an important operation for many DSP and video processing applications. On FPGAs, multi-input addition has traditionally been implemented using trees of carry-propagate adders. This approach has been used because the traditional lookup table (LUT) structure of FPGAs is not amenable to compressor trees, which are used to implement multi-input addition and parallel multiplication in ASIC technology. In prior work, we developed a greedy heuristic method to map compressor trees onto the general logic of an FPGA. Although redundant addition is widely used to design parallel multi operand adders for ASIC implementations, the use of redundant adders on Field Programmable Gate Arrays (FPGAs) has generally been avoided. The main reasons are the efficient implementation of carry propagate adders (CPAs) on these devices (due to their specialized carry-chain resources) as well as the area overhead of the redundant adders when they are implemented on FPGAs. Field Programmable Gate Arrays (FPGAs) are semiconductor devices that are based around a matrix of configurable logic blocks (CLBs) connected via programmable interconnects. FPGAs can be reprogrammed to desired application or functionality requirements after manufacturing. This feature distinguishes FPGAs from Application Specific Integrated Circuits (ASICs), which are custom manufactured for specific design tasks. The FPGA configuration is generally specified using a hardware description language (HDL), similar to that used for an application-specific integrated circuit (ASIC). This paper presents different approaches to the efficient implementation of generic carry-save compressor trees on FPGAs.

They present a fast critical path, independent of bit width, with practically no area overhead compared to CPA trees. Along with the classic carry-save compressor tree, we present a novel linear array structure, which efficiently uses the fast carry-chain resources. This approach is defined in a parameterizable HDL code based on CPAs, which makes it compatible with any FPGA family or vendor. A detailed study is provided for a wide range of bit widths and large number of operands. Compared to binary and ternary CPA trees, increases speedups for 16-bit width. Multiplier unit is an inevitable component in many digital signal processing (DSP) applications involving multiplications. Modified Wallace multiplier unit is used for high performance digital signal processing systems. The DSP applications include filtering, convolution, and inner products. Most of digital signal processing methods use nonlinear functions such as discrete cosine transform (DCT) or discrete wavelet transforms (DWT). Because they are basically accomplished by repetitive application of multiplication and addition, the speed of the Multiplication and addition arithmetic determines the execution speed and performance of the entire calculation. Multiplication-and-accumulate operations are typical for digital filters. Therefore, the functionality of the Multiplier unit enables high-speed filtering and other processing typical for DSP applications.

II. COMPRESSOR

A multiplier is one of the key hardware blocks in most digital and high performance systems such as FIR filters, digital signal processor, microprocessors etc. With advances in technology, many researchers have tried and strive to
design multipliers which offer either of the following—high speed, low power consumption, less area combination of them in multipliers, thus making them compatible for various high speed, low power, and compact VLSI implementations. However, area and speed are two conflicting constraints. Therefore, improving speed always results in larger area. The most efficient multiplier structure will vary depending on the throughput requirement of the application. The first step of the design process is the selection of the optimum circuit structure. The combined factors of low power, low transistor count and minimum delay makes the 5:2 and 4:2 compressors, the appropriate choice. In these compressors, the outputs generated at each stage are efficiently used by replacing the XOR blocks with multiplexer blocks. The select bits to the multiplexers are available much ahead of the inputs so that the critical path delay is minimized. The various adder structures in the conventional architecture are replaced by compressors.

**Fig1. 4:2 Compressor.**

The use of two full adders would introduce a delay of 4 whereas the use of 4:2 compressors reduces the latency to 3. Two full adders are replaced by a single 4:2 compressor.

**III. WALLACE TREE MULTIPLIER**

A Wallace tree multiplier is an efficient hardware implementation of a digital circuit that multiplies two integers devised by an Australian computer scientist Chris Wallace. Wallace tree reduces the no. of partial products and use carry select adder for the addition of partial products.

**A. Multiplier**

In this figure 2 blue circle represent full adder and red circle represent the half adder. Wallace tree has three steps. Multiply each bit of multiplier with same bit position of multiplicand. Depending on the position of the multiplier bits generated partial products have different weights. Reduce the number of partial products to two by using layers of full and half adders. After second step we get two rows of sum and carry, add these rows with conventional adders. As long as there are three or more rows with the same weight add a following layer. Take any three rows with the same weights and input them into a full adder. The result will be an output row of the same weight i.e sum and an output row with a higher weight for each three input wires i.e carry. If there are two rows of the same weight left, input them into a half adder. If there is just one row left, connect it to the next layer. The advantage of the Wallace tree is that there are only \( O(\log n) \) reduction layers (levels), and each layer has \( O(1) \) propagation delay. As making the partial products is \( O(1) \) and the final addition is \( O(\log n) \), the multiplication is only \( O(\log n) \), not much slower than addition (however, much more expensive in the gate count). For adding partial products with regular adders would require \( O(\log n^2) \) time.

**Fig2. 8 Bit×8 Bit Wallace Tree.**

**IV. MODIFIED WALLACE TREE MULTIPLIER**

A modified Wallace multiplier is an efficient hardware implementation of digital circuit multiplying two integers. Generally in conventional Wallace multipliers many full adders and half adders are used in their reduction phase. Half adders do not reduce the number of partial product bits. Therefore, minimizing the number of half adders used in a multiplier reduction will reduce the complexity. Hence, a modification to the Wallace reduction is done in which the delay is the same as for the conventional Wallace reduction. The modified reduction method greatly reduces the number of half adders with a very slight increase in the number of full adders. Reduced complexity Wallace multiplier reduction consists of three stages. First stage the \( N \times N \) product matrix is formed and before the passing on to the second phase the product matrix is rearranged to take the shape of inverted pyramid. During the second phase the rearranged product matrix is grouped into non-overlapping group of three as shown in the figure 2, single bit and two bits in the group will be passed on to the next stage and three bits are given to a full adder. The number of rows in the in each stage of the reduction phase is calculated by the formula.
Implementation of 64-Bit Multiplier Unit using Modified Wallace Structure

If \( r_{j+1} = 2r_j/3 + r_j \mod 3 \)

If \( r_j \mod 3 = 0 \), then \( r_{j+1} = 2r/3 \)

If the value calculated from the above equation for number of rows in each stage in the second phase and the number of row that are formed in each stage of the second phase does not match, only then the half adder will be used. The final product of the second stage will be in the height of two bits and passed onto the third stage. During the third stage the output of the second stage is given to the carry propagation adder to generate the final output.

Thus 64 bit modified Wallace multiplier is constructed and the total number of stages in the second phase is 10. As per the equation the number of row in each of the 10 stages was calculated and the use of half adders was restricted only to the 10th stage. The total number of half adders used in the second phase is 8 and the total number of full adders that was used during the second phase is slightly increased that in the conventional Wallace multiplier. Since the 64 bit modified Wallace multiplier is difficult to represent, a typical 10-bit by 10-bit reduction shown in figure 2 for understanding. The modified Wallace tree shows better performance when carry save adder is used in final stage instead of ripple carry adder. The carry save adder which is used is considered to be the critical part in the multiplier because it is responsible for the largest amount of computation.

V. REGULAR CS COMPRESSOR TREE DESIGN

The classic design of a multi operand CS compressor tree attempts to reduce the number of levels in its structure. The 3:2 counters or the 4:2 compressors are the most widely known building blocks to implement it.

![Fig3. Modified Wallace Reduction Process.](image)

![Fig4. Block Diagram of Modified Wallace Tree Multiplier.](image)

![Fig5. N-Bit Width Cs 9:2 Compressor Tree Based On A Linear Array.](image)
VI. SIMULATION RESULTS

Fig 6. Time Model of The Proposed CS 9:2 Compressor Tree.

Fig 7. Critical Path of the Proposed 9:2 Compressor Tree for Linear Array Behavior.

Fig 8. Transformation of N-Bit Width 9:2 Linear Array Compressor Tree.

VI. SIMULATION RESULTS

Fig 9. RTL Schematic for Modified Wallace Multiplier.

Fig 10. Modified Wallace Multiplier Waveforms.

VII. CONCLUSION

Efficiently implementing Modified Wallace Multiplier on FPGA, in terms of area and speed, is made possible by using the specialized carry-chains of these devices in a novel way. Similar to what happens when using ASIC technology, the proposed CS linear array compressor trees lead to marked improvements in speed compared to CPA approaches and, in general, with no additional hardware cost. Furthermore, the proposed high-level definition of CSA arrays based on CPAs facilitates ease-of-use and portability, even in relation to future FPGA architectures, because CPAs will probably remain a key element in the next generations of FPGA. AAs compare to conventional multiplier number of hardware components are less there by area over head can be reduced cost is less. In future we can extend this by using MAC architecture and can be extended as to implement as ALU. The functionality is verified through XILINX ISE using VERILOG HDL.
VIII. REFERENCES


