

Architectural Implementation of Optimized DSP Accelerator with Modified Booth Recoder

Linu M Jiji¹, Ragimol²

¹M.Tech Student, Department of ECE, Sree Buddha College of Engineering, Alappuzha, Kerala, India

²Professor, Department of ECE, Sree Buddha College of Engineering, Alappuzha, Kerala, India

Abstract: Hardware acceleration has been proved as an extremely promising implementation strategy for the digital signal processing (DSP) domain. An accelerator is a hardware module that can be attached to a processor core. It enhances the performance or functionality by executing certain function in the accelerator instead of executing in the processor core. The accelerator module mainly consists of flexible computational units (FCUs). The structure of the flexible computational unit is designed to enable high performance flexible operation chaining based on a set of operation templates found in DSP kernels. The number of flexible computational units is determined at the design time based on the instruction level parallelism and area constraints imposed by the designer. In this work a high performance architectural scheme is designed by combining both the architectural and arithmetic levels of abstraction. The proposed solution forms an efficient design delay tradeoff of 46% delivering optimized latency/area and energy implementations. It also provides high computing performance, real time processing and power efficiency to variety of applications ranging from sensors to servers. The accelerator module can be used in applications where high performance computation is needed. It can be used in video encoding or decoding for high performance encoding and multichannel transcoding. It can also be used in various image processing applications.

Keywords: DSP Accelerator, Digital signal processor, Modified Booth Algorithm, Multiplier

1. Introduction

Embedded systems use specialized accelerators to improve performance and reduce energy consumption especially in areas such as signal processing, video processing, communications. Algorithms can be applied to merge the hardware implementations of accelerators that are used independently to reduce the area [1]. This approach can create multioperational data paths for a given set of applications [2]; however it alone does not provide greater flexibility as the user of such system wish to create additional applications. Flexibility can be achieved with the use of different types of operation templates [3].

The proposed work mainly focus on an efficient implementation of optimized accelerator architecture for digital signal processors to enhance the performance. Most of the operations performed with the help of templates can be executed within the accelerator module without interfering the processor [4]. A template may be defined as a specialized hardware unit or a group of chained unit. A data-flow graph (DFG) is a graph which represents a data dependencies between a number of operations. Any algorithm consists of a number of ordered operations.

In order to cover a part of data flow graph (DFG) using a given set of templates, the subset of templates that match this part must be identified called as template matching and then the most efficient template must be selected [5]. Template selection has major impact on the performance, and many optimization techniques are used to address the problem. The selection of most appropriate template for the required chain of operation based on the data flow graph is important.

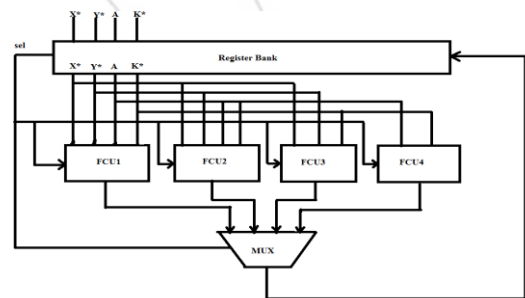


Figure 1: Proposed Optimized Accelerator Architecture for DSP

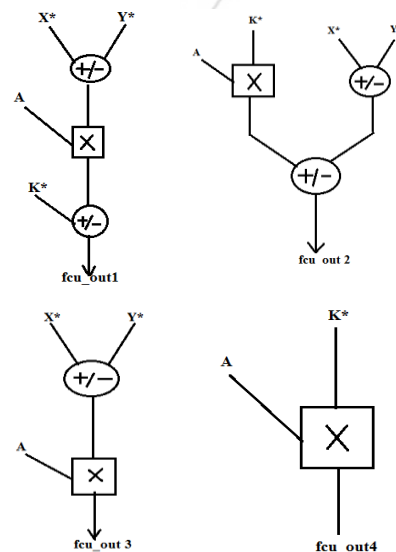


Figure 2: Templates used in the flexible computational unit

2. Flexible Accelerator Architecture

An optimized accelerator architecture for DSP using modified booth is shown in figure1. The architecture mainly consists of flexible computational units (FCUs). Each FCU operates directly on CS operands and produces data in the same form for the direct use of intermediate result. The number of computational unit is determined at the design time based on instruction level parallelism and area constraints imposed by the designer. Each FCU can be configured to operate based on a set of operation templates. The most suitable FCU is selected with the help of multiplexer. Register bank is mainly used to store the intermediate results and sharing values of operands among the FCUs[6].

The different types of templates are shown in figure2. Each template may consist of an adder/subtractor module and a multiplier section. The multiplier here used is a modified booth multiplier. Booth algorithm for multiplication is a simple method in which multiplication is carried out with repeated addition operation. To overcome the main limitations of booth algorithm modified booth algorithm is used.

Modified booth multiplier[7] is used to perform high speed multiplication using modified booth algorithm. The main advantage is that we can reduce the number of partial products to half. The detailed diagram of multiplier is shown in figure 3.

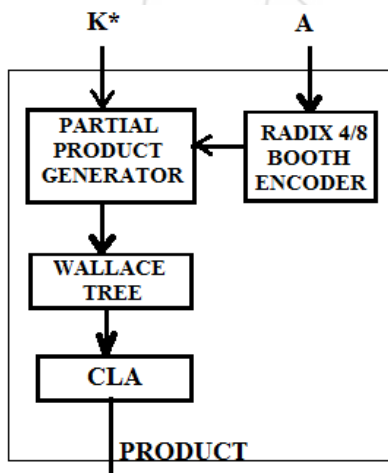


Figure 3: Modified Booth Encoded Multiplier

The Radix8 booth encoder within the multiplier performs the process of encoding the multiplicand based on the multiplier bits. It will compare three bits at a time based on the overlapping technique. For the purpose of generation of the partial products partial product generator is used. For large multipliers the performance of the modified booth algorithm is limited. For that purpose booth encoding together with the Wallace tree structure can be used. Wallace tree adders are used in high speed designs to produce two rows of partial products that can be added in the final stage. Critical path and the number of adders are reduced as compared to the parallel adders. The speed, area and the power consumption of the multipliers are directly proportional to the efficiency of the compressors.

A carry save adder is a type of digital adder used in the computer architecture to compute the sum of three or more n-bit numbers in binary. It differs from other digital adders in the sense that it outputs two numbers of the same dimension as that of the inputs, one which is a sequence of the sum bit and the other which is a sequence of carry bits. This kind of carry save adder trees are used for high speed implementation of multiple operand addition. The adder/subtractor module is mainly designed with the help of carry select adders instead of using the ripple carry adders. Thus the carry propagation in the ripple carry adder can be completely overcome by using the carry select adder. This will significantly reduces the delay and enhances the performance.

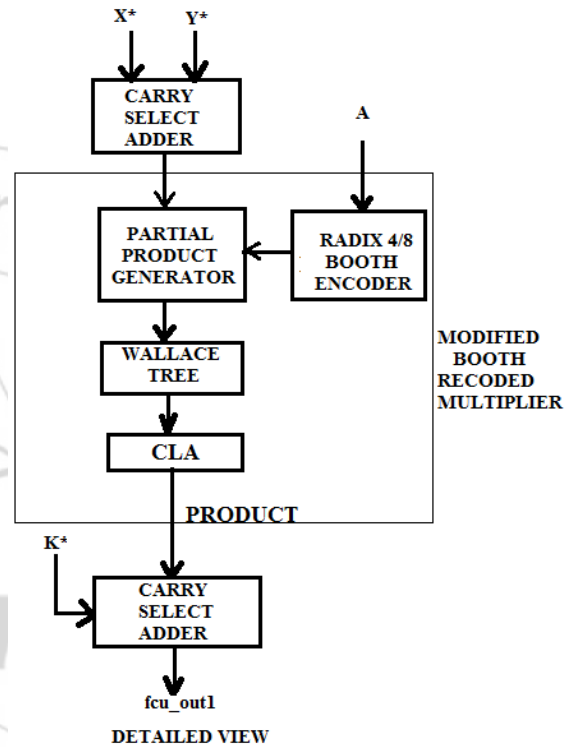


Figure 4: Detailed view of computational unit 1

3. Experimental Results

The model is simulated using Xilinx ISE Design Suite 13.2. Figure 5 shows the simulation result for multiplier. Figure 6 shows the simulation result for 16bit carry look ahead adder. Figure 7,8,9,10 shows the simulation result for different flexible computational units.

Modified Booth Encoded Multiplier

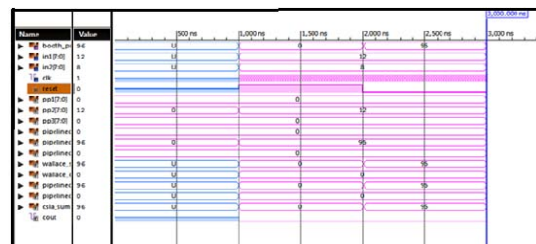


Figure 5: Simulation result for modified booth encoded multiplier

CARRY SELECT ADDER



Figure 6: Simulation result for carry select adder FCU1



Figure 7: Simulation result for FCU1

FCU 2

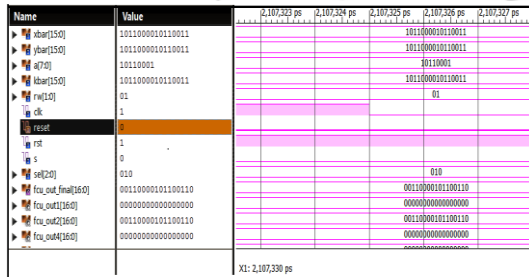


Figure 8: Simulation result for FCU2

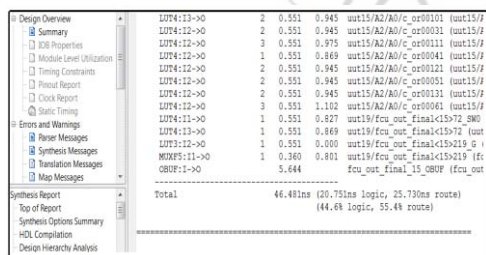


Figure 9: Total delay estimation

4. Conclusion

In this brief, an optimized accelerator architecture for DSP that exploits the merits of carry save arithmetic in case of multiplication and fast addition operations to enable fast chaining of additive and multiplicative operations is introduced. From the result analysis it is clear that the delay can be reduced by optimizing the architecture and incorporating the modified booth. The proposed flexible accelerator architecture is able to operate on both conventional two's complement and carry save formatted

data operands, thus enabling high degrees of computational density to be achieved. The accelerator architecture implementation makes the operation of digital signal processors much faster. Experimental analyses have shown that the proposed solution forms an efficient design tradeoff point delivering optimized latency/area and energy implementations.

5. Acknowledgment

This journey of self actualization would not have been fulfilled without the guidance and support of several individuals. My deep appreciation is extended to my Thesis Guide Prof. Ragimol, Department of Electronics and communication for her guidance, encouragement and appreciation.

References

- [1] Kostas Tsoumanis, Sotirios Xydis, Georgios Zervakis, and Kiamal Pekmestzi, "Flexible DSP Accelerator Architecture Exploiting Carry-Save Arithmetic," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, 2015
- [2] S. Xydis, G. Economakos, D. Soudris, and K. Pekmestzi, "High performance and area efficient flexible DSP datapath synthesis," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 19, no. 3, pp. 429–442, Mar. 2011
- [3] Andrea Lodi, Mario Toma, Fabio Campi, Andrea Cappelli, Roberto Canegallo, and Roberto Guerrieri, "A VLIW Processor With Reconfigurable Instruction Set for Embedded Applications," *IEEE Journal Of Solid-state Circuits*, Vol. 38, No. 11, November 2003
- [4] Hadi Parandeh-Afshar, Ajay Kumar Verma, Philip Brisk, and Paolo Ienne, "Improving FPGA Performance for Carry-Save Arithmetic," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, Vol. 18, No. 4, April 2010
- [5] Peter A. Milder, Franz Franchetti, Formal Datapath Representation and Manipulation for Implementing DSP Transforms June 2008
- [6] Giovanni Ansaloni, Paolo Bonzini and Laura Pozzi, Member, IEEE, "EGRA: A Coarse Grained Reconfigurable Architectural Template" *IEEE Trans. Very Large Scale Integr. (VLSI) Systems*, Vol. 19, No. 6, June 2011.
- [7] Kostas Tsoumanis, Student Member, IEEE, Sotiris Xydis, Constantinos Efstathiou, Nikos Moschopoulos, and Kiamal Pekmestzi "An Optimized Modified Booth Recoder for Efficient
- [8] Design of the Add-Multiply Operator" *IEEE Transactions On Circuits And Systems—I: Regular Papers*, Vol. 61, No. 4, April 2014