IOSR Journal of Electronics and Communication Engineering (IOSR-JECE)
e-ISSN: 2278-2834,p-ISSN: 2278-8735
PP 19-22
www.iosrjournals.org

Low Power Exact Computation for FPGA Based On Memoization Technique

S.Sathya[1], R.Poomurugan[2]
[1](Student,Gnanamani College of technology, Namakkal)
[2](Ap/Ece,Gnanamani College of technology, Namakkal)
Corresponding Authors: sindu.sathya6@gmail.com Phone No--+91-8883682381

Abstract: *Memoization is a fundamental technique in computer science, providing the basis for dynamic programming. This paper goes on to explore the idea of memoization, in which values are aggressively reused from the cache even in the process of computing a value to store in the cache. It studies memoization on FPGA and analyzes different architectural and design parameters that should be considered.In this paper I take image multiplication to prove the process held at minimum time period with small area.The proposed design flow leverages on memo table to enable memoization-based microarchitecture generation. When compared with the previous approaches of bit-width truncation and approximate multipliers, memoization-based approximate computation on FPGA achieves a significant dynamic power saving with very small area overhead and better power-to-signal noise ratio values for the image processing benchmarks. An optimization is going to done by Altera Quatrus II and Modelsim EDA tools.*

Keywords: *Memo table, FPGA, Quatrus II*

I. Introduction

Memoization on FPGA and analyzes different architectural and design parameters that should be considered. It becomes very important to consider how to utilize many cores effectively. Algorithmic based optimization is a fundamental method which aims at reducing the number of operations used in an algorithm by decreasing the number of steps that the algorithm requires to take. Parallel processing breaks down an algorithm into processes that can be run in parallel while preserving the data dependencies among the operations. In both schemes, the building blocks of the optimization methods are the operations where the former reduces the number of operations and the latter runs them in parallel. In both cases, the input data is somewhat ignored. In other words, the algorithmic optimization will be done in the same way regardless of the input data type

Since memoization requires a comparison of input values, the comparison logic and the read–write operations on a block of memory resource will incur additional area overhead, which must be clearly balanced against the power saving. Additional resources like these also present an additional load on the clock network on the FPGA and may impact the maximum operating frequency of the design if not managed properly. Memoization-based approximate computing is also challenging from an application perspective. Different comparison measures need to be selected according to the application type in order to achieve the maximal power saving while sacrificing minimal quality.

II. Existing System

The Support Vector Machine (SVM) algorithm is probably the most widely used kernel learning algorithm. It achieves relatively robust pattern recognition performance established concepts in optimization theory.

Despite this mathematical classicism, the implementation of efficient SVM solvers has diverged from the classical methods of numerical optimization. This divergence is common to virtually all learning algorithms. The numerical optimization literature focuses on the asymptotical performance: how quickly the accuracy of the solution increases with computing time. In the case of learning algorithms, two other factors mitigate the impact of optimization accuracy. SVMs (Support Vector Machines) are a useful technique for data classification. Although SVM is considered easier to use than Neural Networks, users not familiar with it often get unsatisfactory results at first. Here I outline a "cookbook" approach which usually gives reasonable results.

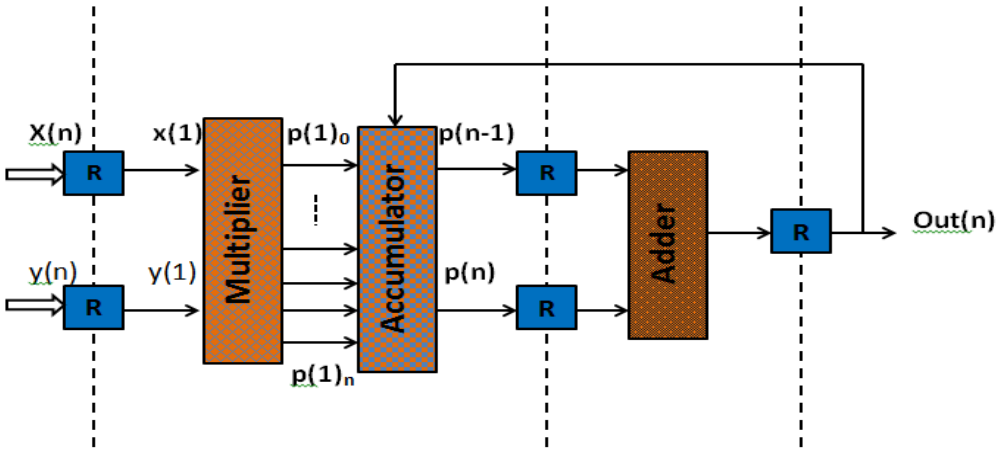


Fig 1:pipelined architecture of dot product

Limitations

Memoization requires a comparison of input values, the comparison logic and the read–write operations on a block of memory resource will incur additional area overhead, which must be clearly balanced against the power saving. Additional resources like these also present an additional load on the clock network on the FPGA and may impact the maximum operating frequency of the design if not managed properly .

- Additional area overhead
- Dynamic power dissipation is high

III. Proposed System

The operands and the operation which want to perform are stored in the memorization block. If any repeated entry presents then the output read from the memorization block via the mux. It is easy and speedup the process.

In memoization block it contain the MEMO TABLE . This implement stage of the pipeline can greatly accelerate the speed in which multi cycle instructions complete. And thus reduce the number of occurences of out of order completions. Unforunately a compiler or run time scheduler sometimes espects an instruction to complete in multiple cycles. Since with our technique it may complete much sooner than expected there is no instruction that uses the same CU that is ready to be issued. This problem is compounded in multiplication units which are themselves pipelined in order to achieve high throughput. New compiler design and run time scheduling is beyond the scope of this work.

The two indicators that measure the success of the MEMO TABLE technique are:

Hit ratio: the hit ratio of a MEMO table will show how many multiple cycle operations are avoided. A higher hit ratio implies that less instances of multiple cycle operations are performed.

Speedup the end goal of using MEMO TABLE is to accelerate processing, if the enhancement has no impact on performance. The extra complexity of adding it isn't worth the effort.

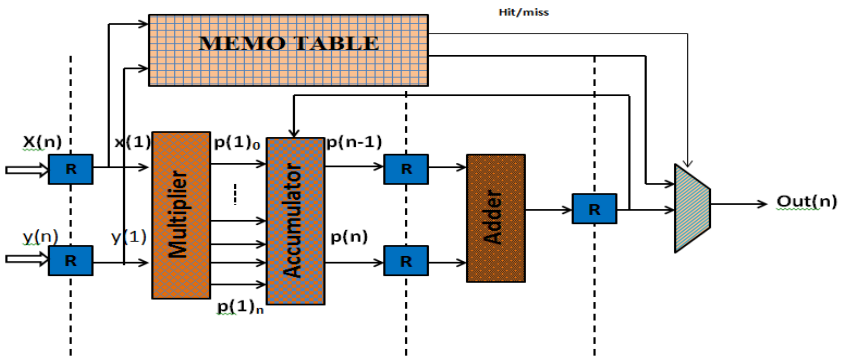


Fig. 2 Proposed architecture

IV. Simulation& Synthesis Results

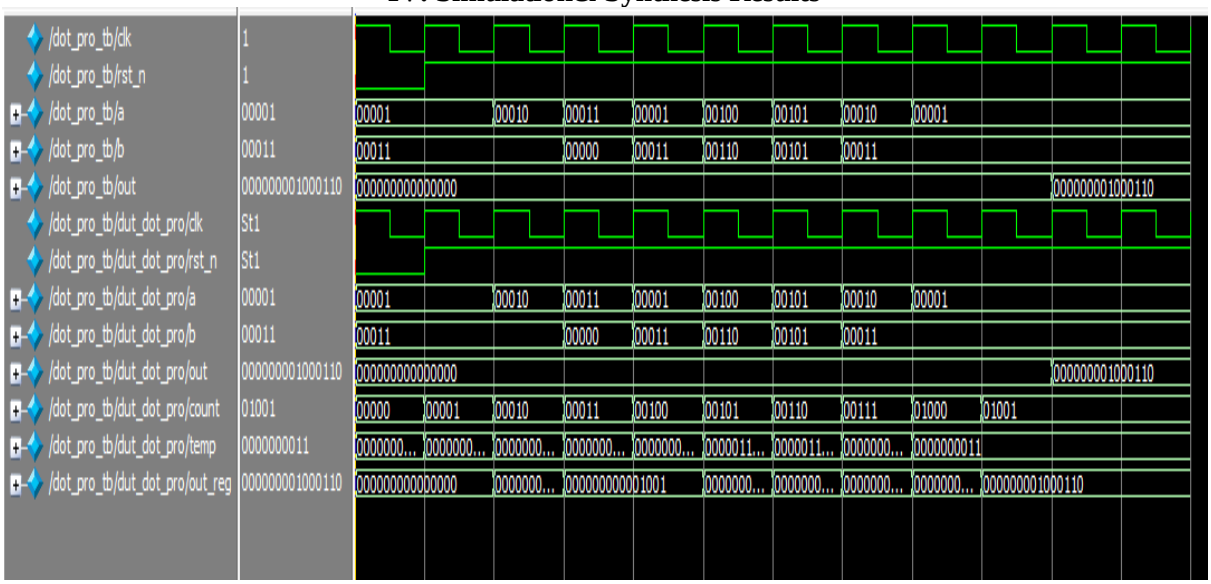


Fig.3 Memoization with dot product

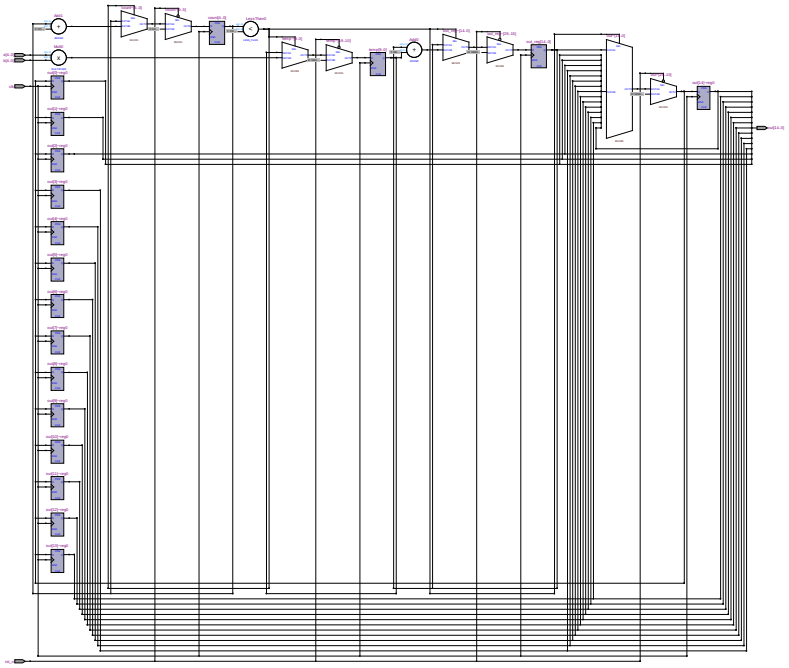
Area Optimization result

Flow Status	Successful - Tue Aug 16 11:31:06 2016
Quartus II Version	9.0 Build 132 02/25/2009 SJ Web Edition
Revision Name	dot_pro
Top-level Entity Name	dot_pro
Family	Cyclone II
Device	EP2C35F672C6
Timing Models	Final
Met timing requirements	Yes
Total logic elements	50 / 33,216 (< 1 %)
Total combinational functions	40 / 33,216 (< 1 %)
Dedicated logic registers	45 / 33,216 (< 1 %)
Total registers	45
Total pins	27 / 475 (6 %)
Total virtual pins	0
Total memory bits	0 / 483,840 (0 %)
Embedded Multiplier 9-bit elements	1 / 70 (1 %)
Total PLLs	0 / 4 (0 %)

POWER REPORT

PowerPlay Power Analyzer Status	Successful - Tue Aug 16 11:35:38 2016
Quartus II Version	9.0 Build 132 02/25/2009 SJ Web Edition
Revision Name	dot_pro
Top-level Entity Name	dot_pro
Family	Cyclone II
Device	EP2C35F672C6
Power Models	Final
Total Thermal Power Dissipation	113.11 mW
Core Dynamic Thermal Power Dissipation	0.00 mW
Core Static Thermal Power Dissipation	79.93 mW
I/O Thermal Power Dissipation	33.17 mW
Power Estimation Confidence	Low: user provided insufficient toggle rate data

RTL VIEW



V. Conclusion

I have proposed memoization based exact computing and applied it to dot product (computation) on FPGAs. This should be used under the image processing application also. Here I show that it is possible to achieve the power saving and area overhead. In addition, approximate designs of functional units, such as multipliers and targeting ASIC technologies, do not necessarily lend to FPGAs, as shown in this paper. I can use this memorization technique in designs of approximate mirror adders, voltage scaling, under designed multiplier.

References

[1]. J. Huang, J. Lach, and G. Robins, "A methodology for energy-qualitytradeoff using imprecise hardware," in Proc. 49th Annu. Design Autom.Conf., 2012, pp. 504–509.

[2]. J. Han and M. Orshansky, "Approximate computing: An emergingparadigm for energy-efficient design," in Proc. 18th IEEE Eur. TestSymp., Avignon, France, May 2013, pp. 1–6.

[3]. H. Esmaeilzadeh, A. Sampson, L. Ceze, and D. Burger, "Architecturesupport for disciplined approximate programming," in Proc. 17th Int.Conf. ASPLOS, 2012, pp. 301–312.

[4]. V. Gupta, D. Mohapatra, S. P. Park, A. Raghunathan, and K. Roy,"IMPACT: Imprecise adders for low-power approximate computing,"in Proc. Int. Symp. Low Power Electron. Design (ISLPED), Aug. 2011,pp. 409–414.

[5]. K. Shi, D. Boland, and G. A. Constantinides, "Imprecise datapathdesign: An overclocking approach," ACM Trans. ReconfigurableTechnol. Syst., vol. 8, no. 2, Apr. 2015, Art. ID 6.

[6]. D. Citron, D. Feitelson, and L. Rudolph, "Accelerating multi-mediaprocessing by implementing memoing in multiplication and divisionunits," in Proc. 8th Int. Conf. ASPLOS, 1998, pp. 252–261.

[7]. F. Khalvati and M. D. Aagaard, "Window memoization: An efficienthardware architecture for high-performance image processing,"J. Real-Time Image Process., vol. 5, no. 3, pp. 195–212, Sep. 2010.

[8]. D. Michie, "'Memo' functions and machine learning," Nature, vol. 218,pp. 19–22, Apr. 1968.

[9]. L. Sterling and E. Shapiro, The Art of Prolog, 2nd ed. Cambridge, MA,USA: MIT Press, Mar. 1994.

[10]. A. K. Mishra, R. Barik, and S. Paul, "iACT: A software-hardwareframework for understanding the scope of approximate computing,"in Proc. Workshop Approx. Comput. Across Syst. Stack (WACAS),Salt Lake City, UT, USA, 2014.