J.Natn.Sci.Foundation Sri Lanka 2008 36 (3):235-237

SHORT COMMUNICATION

Performance evaluation of multipliers in reconfigurable hardware

W.A.S. Wijesinghe[1], M.K. Jayananda[2] and D.U.J. Sonnadara[2*]

[1] *Department of Electronics, Faculty of Applied Sciences, Wayamba University of Sri Lanka, Mankandura.*

[2] *Department of Physics, Faculty of Science, University of Colombo, Colombo 03.*

Revised: 20 December 2007 ; Accepted: 20 June 2008

Abstract: Hardware multiplier is a critical element in many computing intensive sub-systems that are implemented in Field Programmable Gate Arrays (FPGAs). In this study, performance comparison between several different types of array-based unsigned 8-bit multipliers (both combinational and pipelined) for two types of FPGAs (XC4005XLPC84-3C and XC3S1000FT256-4C Spartan 3) in terms of resource utilization and critical path delays was carried out. For combinational multipliers, the embedded multiplier in the high density Spartan 3 FPGA has the lowest critical path delay. MUX-based Carry Save Array (CSA) multiplier when implemented in the low density XC4005 FPGA, utilized 24% less resources and resulted is 42% improvement in the latency than the standard multiplier available in the hardware description language (VHDL) library. For pipelined multipliers, single stage pipelined multiplier of the same architecture, for the same chip, utilized 18% more resources but produced a 84% improvement in the latency. Thus, to obtain the optimum performance of FPGA hardware in high speed applications, MUX-based pipelined CSA multipliers are recommended.

Keywords: FPGA, Multiplier hardware architecture, reconfigurable computing, VHDL

INTRODUCTION

Configurable hardware devices, especially Field Programmable Gate Arrays (FPGAs) are becoming popular among system designers due to their flexibility and reconfigurability. Most of the application-specific processors that are implemented in FPGAs, such as digital signal processors, require fast execution time for mathematical operations[1,2]. The multiplication operation, among other operations, is a critical operation which consumes considerable amount of logic resources and affect the throughput of the system.

When systems are implemented in reconfigurable hardware, either the standard multipliers available in the hardware description language (VHDL) library or the embedded multipliers are often used. However, this may not be the best solution in terms of resource utilization and speed. In literature, various methods of implementing hardware multipliers in configurable logic are available[3]. These methods provide varying degrees of benefits with respect to utilization of logic resources and latency[4,5]. For hardware designers, it is essential to know the advantages and disadvantages of available multipliers in order to design a system to achieve the optimum performance at the lowest cost. In this study, performance comparisons of several multiplier methods in terms of logic utilization and critical path delay were carried out for two Xilinx FPGA devices.

The hardware multipliers found in literature can be categorized into two groups according to their structure; array multipliers and tree multipliers. Although some argue that tree multipliers are faster than array multipliers[6], the layout of the tree multiplier is highly complex and irregular. This leads to poor design and eventually consume large amounts of logic resources from the configurable device[7,8]. On the other hand, array multipliers are less complex and they have regular structures. This makes it possible to implement them easily in configurable hardware. Furthermore, regular structures are scalable as well as easily pipelined to improve the latency of the hardware.

METHODS AND MATERIALS

Three types of array-based unsigned multipliers (both combinational and pipelined) are considered in this work, namely, Basic Binary multiplier, Cellular multiplier and Carry Save Adder (CSA) multiplier. The implementation of these multipliers was carried out with the VHDL hardware description language. Thus, they can be ported

*Corresponding author

to any type of programmable logic device without making any changes to the developed code.

Two types of FPGAs were used in the evaluation. The first device was a high density XC3S1000FT256-4C Spartan-3 FPGA with 1 million equivalent logic gates, available in the XSA-3S1000 board manufactured by XESS Corporation. The second device was a low density XC4005XLPC84-3C FPGA with 5000 equivalent logic gates in XS40 board from the same manufacturer. The Spartan-3 chip has several 18-bit multipliers. Being a low density device, the XC4005 did not have sufficient resources for implementing 18-bit multipliers. Therefore, for this investigation, 8-bit multipliers were implemented and tested. Each of these multipliers was implemented as a combinational circuit while some of them were later re-implemented as pipelined multipliers. All the multipliers were simulated with the Modelsim Xilinx Edition (version 5.8c) and physically implemented on the two boards for verifications.

The comparison of the performance was based on the resource utilization, which was measured in terms of the total equivalent gate count, and the speed of the multiplier, which was measured in terms of the maximum delay in the critical path. Both these pieces of information were extracted from the reports generated by the Xilinx WebPACK software as and when each circuit was configured for the selected FPGA. The total equivalent gate count is directly available from the Post Map reports. The 'Post Place and Route Static Timing' reports provide the maximum frequency each circuit can perform with the guaranteed accuracy. This is the maximum frequency at which the signal delay in the critical path does not prevent the circuit from functioning correctly. Thus, it provides an indirect measure of the critical path delay.

For the implementation of the circuits, ISE WebPACK version 7.1i was used for the Spartan 3. However, this version of WebPACK did not support XC400 family and therefore version .2.03i had to be used for the XC4005. The delay for the critical path was taken by "Post-Routing Timing Analysis Tool" which comes with the Xilinx ISE WebPACK software. The critical delay equals the execution time of the system when operated at the maximum frequency with a guaranteed accuracy.

Since most array-based multiplier structures are regular structures, the addition of partial products can be carried out sequentially. The cell, the basic building block of the multiplier consists of a partial product generation circuit. The cells are arranged in an array to add row by row to produce the final result. The cell architecture and the cell arrangement are different for each implemented multiplier and the details are described elsewhere[9].

Table 1: Resource utilization and delay in XC4005 and Spartan 3 FPGA

8-bit Multiplier architecture	Total equivalent gates		Delay of critical path (ns)	
	XC4005XL	Spartan 3	XC4005XL	Spartan 3
Basic Binary	961	714	126.58	36.09
Cellular	712	714	105.26	27.49
CSA	969	708	50.25	27.07
MUX based CSA	794	713	40.59	24.28
Standard/Embedded	1040	*4000	69.44	16.54

*The embedded multipliers utilized fix 4000 logic gates. However, these resources are not available for general purpose configurations and thus effectively the resource utilization of the embedded multipliers can be taken as zero.

Table 2: Resource utilization and delay in MUX-based CSA pipeline multiplier for XC4005 and Spartan 3 FPGA

8-bit Multiplier Architecture	Total equivalent gates		Delay of Critical Path (ns)	
	XC4005XL	Spartan 3	XC4005XL	Spartan 3
Single stage	1228	1268	10.51	7.29
Four-stage	1506	1648	10.56	7.30

RESULTS & DISCUSSION

Table 1 shows the logic utilization and critical path delay of implemented 8-bit multipliers in low density XC4005 FPGA and high density Spartan 3 FPGA. Table 2 shows the results for different pipeline stages of implemented MUX-based CSA multipliers.

The embedded multiplier in Spartan-3 had 16.54 ns critical path delay, which is the lowest among the combinational multipliers. Further, the multiplier blocks were implemented in separate logic resources without consuming any resources available to implement general purpose functions. Thus, in Spartan-3, the use of the embedded multipliers will be the best choice, unless one runs out of them in a particular design.

In the XC4005, the best timing performance was seen for the CSA multiplier. The CSA multiplier had 28% lower latency and 31% lower logic gate count compared to the standard multiplier. The MUX-based CSA multiplier implemented in the XC4005 FPGA showed further improvements. Compared to the standard multiplier, logic utilization and latency were improved by 24% and 42% respectively by implementing MUX-based CSA multiplier. Thus, MUX-based CSA multipliers are suitable for the FPGAs without embedded multipliers.

The pipelined multipliers tested in this work showed low critical path delays compared to their non-pipeline counterparts, though they consumed extra logic gates. The MUX-based pipelined CSA multiplier had the lowest critical path delay of 7.29 ns. The increase in pipeline stages did not show improvement in latency though they consumed more logic resources of the device. The single stage MUX-based pipeline CSA multiplier showed 84% improvement in latency compared to the standard multiplier in the XC4005.

In conclusion, the use of embedded multipliers is the best choice in Spartan-3 and MUX-based CSA pipelined multipliers offer the highest speed in XC4005. However, these results may not be extendable to devices from other manufacturers due to different architectures of configurable logic blocks.

Acknowledgement

Financial assistance by the IPPS, Uppsala University, Sweden (research grant number SRI: 01/3) and Wayamba University of Sri Lanka, Makandura (research grant number RG/2005/07) are acknowledged.

References

1. Boemo E., Lopez-Buedo S. & Meneses J. (1998). Some experiments about wave pipelining on FPGAs. *Institute of Electrical and Electronics Engineers*, *Inc. (IEEE) Transactions on Very Large Scale Integration* (VLSI) *Systems* **6**(2): 232-237.
2. Lee H. & Sobelman G.E. (2002). FPGA-based digit-serial CSD FIR filter for image signal format conversion. *Microelectronics Journal* **33** (5-6): 501-508.
3. Jones M., Nakad Z., Plassmann P. & Yi Y. (2006). The use of configurable computing for computational kernels in scientific simulations. *Future Generation Computer Systems* **22** (1-2): 67-79.
4. Yu W.W. & Xing S. (1999). Fixed-point multiplier evaluation and design with FPGA. *Proceedings of the International Society for Optical Engineers* (SPIE). **3844**: 153-161.
5. Rodellar V., Sacristan M.A., Gomez P., Diaz A. & Peinado V. (2000). Performance evaluation of reusable multiplier for rapid prototyping. *Proceedings of the 43rd IEEE Institute of Electrical and Electronics Engineers, Inc. Midwest Symposium on Circuits and Systems*. **1**: 358-361.
6. Huang Z. & Ercegovac M.D. (2003). High-performance left-to-right array multiplier design. *16th IEEE Institute of Electrical and Electronics Engineers, Inc. Symposium of Computer Arithmetic*. 4-11.
7. Kolla Y., Kim Y. & Carter J. (2003). A Novel 32-bit Scalable Multiplier Architecture. *Great Lake Symposium on VLSI*. 241-244.
8. Ciminiera L. & Montuschi P. (1996). Carry-save multiplication schemes without final addition. *IEEE Institute of Electrical and Electronics Engineers, Inc. Transactions on Computers*. **45**(9): 1050-1055.
9. Wijesinghe W.A.S. (2007). Numerical Computation in Configurable Hardware, *M.Phil Thesis*, Department of Physics, University of Colombo, Colombo 03.