FPGA-BASED DESIGN OF A MATH CO-PROCESSOR FOR THE AMIR CPU

ARTHUR TAN FOO YEN

A project report submitted in partial fulfilment of the
requirements for the award of the degree of
Master of Engineering (Computer and Microelectronic Systems)

School of Electrical Engineering
Faculty of Engineering
Universiti Teknologi Malaysia

JULY 2020

ACKNOWLEDGEMENT

First and foremost, I would like to take this opportunity to express my sincere gratitude to my project supervisor, Assoc. Prof. Dr. Muhammad Nasir Bin Ibrahim, who offered abundantly helpful, guidance, patience and support throughout the duration of my project. His advice and encouragement have given me the motivation to accomplish feats.

I would also like to thank Dr. Suhaila Isaak, Dr. Usman Ullah Sheikh, and Dr. Zulfakar Aspar for their invaluable advice during the project seminar. Their advice has allowed me to further improve my work on this project.

In addition, I would like to extend my deepest gratitude to my beloved friends and family for their continuous blessings and moral support throughout the duration of this project.

Not forgetting those who had contributed directly or indirectly in the successful completion of this project, I sincerely appreciate their unconditional support and guidance throughout this project.

ABSTRACT

Math coprocessors are vital components in modern computing to improve the overall performance of the system. The AMIR CPU is a homegrown softcore 32-bit CPU that can only handle integer numbers making it inadequate for high-performance real-time systems. The aim of this project is to design and develop a math coprocessor for the AMIR CPU that can perform addition, subtraction, multiplication and division on IEEE-754 single precision floating-point numbers. The design of the math coprocessor is devised and improved based on past works on IEEE 754 floating-point operations and math coprocessor implementations. The architecture of the proposed math coprocessor consists of a control unit with instruction decode, floating-point computation unit and a register file. The architecture type is a serial controller with pipelined data path. The proposed math coprocessor retrieves instruction from the instruction register, decodes it, retrieves operands from the CPU register, performs computation then stores the results into the internal register, pending retrieval from the AMIR CPU. The proposed math coprocessor managed to achieve at least 99.999% accuracy for all four arithmetic operations with a maximum frequency of 63.8 MHz, while utilizing less than 30% of the available resource on board an Intel Cyclone IV EP4CE10E22C8 FPGA. The design is not without flaws as the proposed design has problems with instruction queueing due to the absence of an instruction buffer. Nevertheless, with further improvements and features, the proposed math coprocessor has the potential to enable the AMIR CPU to be used in a wide range of applications.

ABSTRAK

Pemprosesan matematik adalah komponen yang amat penting dalam pengkomputeran moden untuk menambah baik prestasi keseluruhan sistem. AMIR CPU adalah satu 32-bit CPU teras lembut buatan tempatan yang hanya mampu mengendalikan nombor bulat membuatkan ianya tidak mencukupi untuk mengendalikan sistem masa nyata berprestasi tinggi. Tujuan project ini adalah untuk mereka dan memperbaik satu pemprosesan matematik untuk AMIR CPU yang mampu melaksanakan proses penambahan, penolakkan, pendaraban dan pembahagi atas nombor-nombor terapung ketepatan tunggal IEEE-754. Reka bentuk pemprosesan matematik dirangka dan diperbaiki berdasarkan penyelidikan sebelum ini atas operasi IEEE-754 titik-terapung dan implimentasi pomproses-bersama matematik. Senibina pemprosesan matematik yang dicadangkan terdiri daripada unit kawalan dengan dekod arahan, unit pengiraan titik terapung dan fail daftar. Jenis senibina adalah pengawal bersiri dengan jalur data saluran. Pemproses yang dicadangkan akan mengambil arahan dari daftar arahan, menyahkodnya, mengambil operan dari unit penyimpanan CPU, melakukan pengiraan dan menyimpan hasilnya ke dalam unit penyimpanan dalaman, sementara menunggu pengambilan dari AMIR CPU. Pemprosesan matematik yang dicadangkan berjaya mencapai sekurang-kurangnya ketepatan 99,999% untuk keempat-empat operasi aritmetik dengan frekuensi maksimum 63.8 MHz, sambil menggunakan kurang dari 30% sumber yang ada pada Intel Cyclone IV EP4CE10E22C8 FPGA. Reka bentuk mempunyai sedikt kelemahan kerana reka bentuk yang dicadangkan mempunyai masalah dengan pengaturan arahan disebabkan oleh ketiadaan penyangga arahan. Walau demikian, dengan peningkatan dan ciri-ciri yang lebih maju, pemprosesan matematik yang diusulkan berpotensi untuk membolehkan AMIR CPU digunakan dalam pelbagai aplikasi.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

CISC	–	Complex Instruction Set Computer
CORDIC	–	Coordinate Rotation Digital Computer
CPLD	–	Complex Programmable Logic Devices
CPU	–	Central Processing Unit
CU	–	Control Unit
DSP	–	Digital Signal Processing
FF	–	Flip Flops
FIFO	–	First In First Out
FPGA	–	Field Programmable Gate Array
FPU	–	Floating-point Unit
HDL	–	Hardware Description Language
IEEE	–	Institute of Electrical and Electronics Engineers
IO	–	Input/Output
IP	–	Intellectual Property
LOD	–	Leading One Detector
LSB	–	Least Significant Bit
LUT	–	Look Up Table
MATHCO	–	Math Coprocessor
MSB	–	Most Significant Bit
NaN	–	Not a Number
POR	–	Power on Reset
RISC	–	Reduce Instruction Set Computer
RTL	–	Register Transfer Level
VHDL	–	Very High Speed Integrated Circuit HDL

CHAPTER 1

INTRODUCTION

1.1 Problem Background

Data is everywhere these days. They are in your computer, mobile phone, tablet, watch, car, and even in your rice cooker. Modern day computing devices are data-centric devices which process huge amounts of data every second. However, the central processing unit (CPU) of a system could not optimally process the various types of data. A coprocessor is designed to work together with the CPU to effectively process specific types of data, such as digital signal processing, encryption and memory management (Zhao *et al.*, 2016). The presence of a coprocessor would alleviate the load on the CPU and improve the overall performance of the system.

Arithmetic operations are required to manipulate data in solving computational problems (Ardalan and Adibi, 2005). A math coprocessor is a specialized processor that complements the CPU to specifically perform arithmetic operations. Early embedded processors such as the Intel 4004 does not have hardware support for floating-point operations. Eight years later, the famous Intel 8086 still does not have a built-in floating-point unit (FPU). However, the Intel 8087 was announced as a mathematical coprocessor to add hardware-based floating-point computation for the Intel 8086 and 8088. The Intel 8087 managed to achieve 20% to 500% performance improvements in floating-point computations (Zhao *et al.*, 2016). Math coprocessors are commonly found in general purpose computer architectures and are increasingly found in embedded processors. They are an essential module in the modern-day computing world.

1.2 Problem Statement

The AMIR CPU, a homegrown softcore 32-bit CPU, can only handle integer numbers, which are inadequate for high-performance real-time systems. Software-based floating-point computations are slow and inefficient due to the frequent use of load and store operations. The AMIR CPU does not have the necessary hardware to perform floating-point computations, which are more precise, accurate and vital in today's computing world.

1.3 Research Objectives

At present, there are no math coprocessors developed for the AMIR CPU. The purpose of this project is to design and develop a math coprocessor for the AMIR CPU that is capable of manipulating IEEE-754 single precision floating-point numbers. The math coprocessor must be able to perform addition, subtraction, multiplication and division operations on IEEE-754 floating-point numbers. The objectives of this project are:

i. To design an IEEE 754 compliant 32-bit single precision floating point math coprocessor for the AMIR CPU on an FPGA system.

ii. To perform four basic math operations: addition, subtraction, multiplication and division

iii. To achieve an accuracy of 99% for the floating-point computations

1.4 Scope

The design and development of the math coprocessor could cover a very wide range. Therefore, the scope of this project has been restricted to the following aspects:

i. RTL design of a floating-point math coprocessor for the AMIR CPU to perform addition, subtraction, multiplication and division operation.

ii. Floating-point representation is compliant with IEEE 754 32-bit single precision floating-point format.

iii. Define communication protocol and integration guideline between AMIR CPU and the proposed math coprocessor.

iv. RTL development of the math coprocessor using Verilog HDL and synthesized based on Intel Cyclone IV EX FPGA using Intel Quartus Prime software.

v. Develop simulation testbench to measure the result accuracy of the floating-point operations.

1.5 Report Outline

This report is organized into five chapters, which consists of introduction, literature review, methodology, results and discussion, as well as conclusion and future recommendations.

Chapter 1 is the introduction to this project. This chapter covers the general overview which includes the background, problem statement, research objectives and report outline.

Chapter 2 will discuss on the literature review for this project. This chapter covers the relevant technical concepts and algorithms needed to carry out this project. This chapter also covers the previous works done on floating-point operations and math coprocessor implementations.

Chapter 3 is the methodology of the project. This chapter discusses the overall system architecture of the math coprocessor, the integration of the math coprocessor with the AMIR CPU, and the microarchitectures of all the modules used to implement the floating-point features of the math coprocessor.

Chapter 4 details the results and discussion of the project. The main discussion for this chapter is the simulation results of the four main floating-point operations. Besides that, this chapter also discusses the performance and resource utilization of the design after synthesis as well as benchmarking with related works.

Chapter 5 contains the conclusion of this project along with recommendations that could be done for future improvements.

REFERENCES

Ardalan, S. and Adibi, A. (2005) ‘Design, simulation and synthesis of a 32-bit math-processor’, in *Midwest Symposium on Circuits and Systems*, pp. 1469–1472. doi: 10.1109/MWSCAS.2005.1594390.

Arish, S. and Sharma, R. K. (2015) ‘Run-time reconfigurable multi-precision floating point multiplier design for high speed, low-power applications’, in *2nd International Conference on Signal Processing and Integrated Networks, SPIN 2015*. Institute of Electrical and Electronics Engineers Inc., pp. 902–907. doi: 10.1109/SPIN.2015.7095315.

Arun, K. and Srivatsan, K. (2017) ‘A binary high speed floating point multiplier’, in *2017 International Conference On Nextgen Electronic Technologies: Silicon to Software, ICNETS2 2017*. Institute of Electrical and Electronics Engineers Inc., pp. 316–321. doi: 10.1109/ICNETS2.2017.8067953.

Barrabés Castillo Bratislava, A. and Zálusky Viera Stopjaková, R. (2012) *DESIGN OF SINGLE PRECISION FLOAT ADDER (32-BIT NUMBERS) ACCORDING TO IEEE 754 STANDARD USING VHDL*.

Chen, K. T. F. of E. E. (2017) ‘An IEEE-754 compliant floating-point coordinate rotation digital computer coprocessor on field programmable gate array’.

Cornea, M. *et al.* (2009) ‘A software implementation of the IEEE 754R decimal floating-point arithmetic using the binary encoding format’, *IEEE Transactions on Computers*, 58(2), pp. 148–162. doi: 10.1109/TC.2008.209.

Hemmert, K. S. and Underwood, K. D. (2007) ‘Floating-point divider design for FPGAs’, *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 15(1), pp. 115–118. doi: 10.1109/TVLSI.2007.891099.

Hu, H. *et al.* (2007) ‘A floating-point coprocessor configured by a FPGA in a digital platform based on fixed-point DSP for power electronics’, in *Conference Proceedings - IPEMC 2006: CES/IEEE 5th International Power Electronics and Motion Control Conference*. doi: 10.1109/IPEMC.2006.297256.

Ibrahim, M. N. *et al.* (2018) ‘AMIR CPU: World’s First and only 32-bit Softcore Processor in Schematic on Freeware Platform’, in *Journal of Physics: Conference Series*. Institute of Physics Publishing. doi: 10.1088/1742-6596/1090/1/012003.

Institute of Electrical and Electronic Engineering - IEEE (2008) *IEEE Std 754-2008, Standard for Floating-Point Arithmetic*. IEEE. doi: 10.1109/IEEESTD.2008.4610935.

Institute of Electrical and Electronic Engineering, I. (2019) *IEEE Std 754-2019 - IEEE Standard for Floating-Point Arithmetic*. IEEE.

Jaiswal, M. and Cheung, R. C. C. (2011) 'High Performance FPGA Implementation of Double Precision Floating Point Adder/Subtractor', *International Journal of Hybrid Information Technology, SERSC*, 4.

Jun, K. and Swartzlander, E. E. (2012) 'Modified non-restoring division algorithm with improved delay profile and error correction', in *Conference Record - Asilomar Conference on Signals, Systems and Computers*, pp. 1460–1464. doi: 10.1109/ACSSC.2012.6489269.

Kamble, L., Palsodkar, P. and Palsodkar, P. (2018) 'Research trends in development of floating point computer arithmetic', in *Proceedings of the 2017 IEEE International Conference on Communication and Signal Processing, ICCSP 2017*. Institute of Electrical and Electronics Engineers Inc., pp. 329–333. doi: 10.1109/ICCSP.2017.8286371.

Karatsuba, A. and Ofman, Y. P. (1963) 'Multiplication of Many-Digital Numbers by Automatic Computers', in.

Kong, I. and Swartzlander, E. E. (2011) 'A goldschmidt division method with faster than quadratic convergence', *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 19(4), pp. 696–700. doi: 10.1109/TVLSI.2009.2036926.

Krishnan, T. and Saravanan, S. (2020) 'Design of Low-Area and High Speed Pipelined Single Precision Floating Point Multiplier', in *2020 6th International Conference on Advanced Computing and Communication Systems, ICACCS 2020*. Institute of Electrical and Electronics Engineers Inc., pp. 1259–1264. doi: 10.1109/ICACCS48705.2020.9074366.

Kumar, J. V. *et al.* (2014) *FPGA Based Implementation of Pipelined 32-bit RISC Processor with Floating Point Unit, Journal of Engineering Research and Applications www.ijera.com*. Available at: www.ijera.com (Accessed: 18 December 2019).

Kwon, T. J., Sondeen, J. and Draper, J. (2007) 'Floating-point division and square root using a Taylor-series expansion algorithm', in *Midwest Symposium on Circuits and Systems*, pp. 305–308. doi: 10.1109/MWSCAS.2007.4488594.

Lavanya, B. K. and Shetty, S. (2017) ‘Computing Non-Restoring and Newton Raphson’s Method for Division’, *Journal of Electronics and Communication Engineering*. Ver. II, 12(4), pp. 57–60. doi: 10.9790/2834-1204025760.

Malik, A. and Ko, S. B. (2006) ‘A study on the floating-point adder in FPGAS’, in *Canadian Conference on Electrical and Computer Engineering*. doi: 10.1109/CCECE.2006.277498.

Nannarelli, A. (2019) ‘Fused Multiply-Add for Variable Precision Floating-Point’, in *International System on Chip Conference*. IEEE Computer Society, pp. 342–347. doi: 10.1109/SOCC46988.2019.1570555329.

Nasir Ibrahim, M. *et al.* (2015) *The Implementation of a Pipelined Floating-point CORDIC Coprocessor on NIOS II Soft Processor, International Journal of Electrical, Electronics and Data Communication*.

Palekar, S. and Narkhede, N. (2017a) ‘32-bit RISC Processor with floating point unit for DSP applications’, in *2016 IEEE International Conference on Recent Trends in Electronics, Information and Communication Technology, RTEICT 2016 - Proceedings*. Institute of Electrical and Electronics Engineers Inc., pp. 2062–2066. doi: 10.1109/RTEICT.2016.7808202.

Palekar, S. and Narkhede, N. (2017b) ‘High speed and area efficient single precision floating point arithmetic unit’, in *2016 IEEE International Conference on Recent Trends in Electronics, Information and Communication Technology, RTEICT 2016 - Proceedings*. doi: 10.1109/RTEICT.2016.7808177.

Palsodkar, P. and Palsodkar, P. (2017) ‘Three operand fused floating point add-subtract unit using redundant adder’, in *IEEE Region 10 Annual International Conference, Proceedings/TENCON*. Institute of Electrical and Electronics Engineers Inc., pp. 1343–1346. doi: 10.1109/TENCON.2017.8228066.

Patil, V. *et al.* (2015) ‘Out of order floating point coprocessor for RISC v ISA’, in *19th International Symposium on VLSI Design and Test, VDAT 2015 - Proceedings*. Institute of Electrical and Electronics Engineers Inc. doi: 10.1109/ISVDAT.2015.7208116.

San, A. M. and Yakunin, A. N. (2019) ‘Hardware implementation of floating-point operating devices by using IEEE-754 binary arithmetic standard’, in *Proceedings of the 2019 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering, ElConRus 2019*. Institute of Electrical and Electronics Engineers Inc., pp. 1624–1630. doi: 10.1109/EIConRus.2019.8656775.

Shao, J., Ye, N. and Zhang, X. Y. (2008) ‘An IEEE compliant floating-point adder with the deeply pipelining paradigm on FPGAs’, in *Proceedings - International Conference on Computer Science and Software Engineering, CSSE 2008*. doi: 10.1109/CSSE.2008.590.

Sharma, A., Singh, S. and Sharma, A. (2018) ‘Implementation of single precision conventional and fused floating point add-sub unit using Verilog’, in *Proceedings of the 2017 International Conference on Wireless Communications, Signal Processing and Networking, WiSPNET 2017*. Institute of Electrical and Electronics Engineers Inc., pp. 169–171. doi: 10.1109/WiSPNET.2017.8299741.

Sharma, J. *et al.* (2016) ‘Fused floating-point add and subtract unit’, in *IC-GET 2015 - Proceedings of 2015 Online International Conference on Green Engineering and Technologies*. doi: 10.1109/GET.2015.7453797.

Singh, N. and Sasamal, T. N. (2016a) ‘Design and synthesis of goldschmidt algorithm based floating point divider on FPGA’, in *International Conference on Communication and Signal Processing, ICCSP 2016*. Institute of Electrical and Electronics Engineers Inc., pp. 1286–1289. doi: 10.1109/ICCSP.2016.7754360.

Singh, N. and Sasamal, T. N. (2016b) ‘Design and Synthesis of Single Precision Floating Point Division based on Newton-Raphson Algorithm on FPGA’, *MATEC Web of Conferences*. EDP Sciences, 57, p. 01009. doi: 10.1051/MATECCONF/20165701009.

Soni, M. K. and Hemant, M. B. K. (2009) *FPGA Implementation of IEEE 754 Standard Based Arithmetic Unit for Floating Point Numbers VLSI Design & CAD*.

Sureka, N., Porselvi, R. and Kumuthapriya, K. (2013) ‘An efficient high speed Wallace tree multiplier’, in *2013 International Conference on Information Communication and Embedded Systems, ICICES 2013*. doi: 10.1109/ICICES.2013.6508192.

Vikas Krishnan, R., Alwyn Rajiv, S. and Nancy Deborah, R. (2019) ‘A comparative study on the performance of FPGA implementations of high-speed single-precision binary floating-point multipliers’, in *Proceedings of the 2nd International Conference on Smart Systems and Inventive Technology, ICSSIT 2019*. Institute of Electrical and Electronics Engineers Inc., pp. 1041–1045. doi: 10.1109/ICSSIT46314.2019.8987800.

Xunying, Z. and Xubang, S. (2008) ‘A power-efficient floating-point Co-processor design’, in *Proceedings - International Conference on Computer Science and Software Engineering, CSSE 2008*, pp. 75–78. doi: 10.1109/CSSE.2008.795.

Yang, H. J., Yu, F. and Han, D. D. (2013) *High performance FPGA implementation of floating point addition*, *Applied Mechanics and Materials*. doi: 10.4028/www.scientific.net/AMM.380-384.3316.

Zhang, H., Chen, D. and Ko, S. B. (2017) ‘Area- and power-efficient iterative single/double-precision merged floating-point multiplier on FPGA’, *IET Computers and Digital Techniques*. Institution of Engineering and Technology, 11(4), pp. 149–158. doi: 10.1049/iet-cdt.2016.0100.

Zhao, C. *et al.* (2016) ‘A high-efficient floating point coprocessor for SPARC Leon2 embedded processor’, in *Proceedings - 2015 IEEE 11th International Conference on ASIC, ASICON 2015*. doi: 10.1109/ASICON.2015.7517173.