DOUBLE PRECISION FLOATING POINT UNIT USING ISim SIMULATOR

NURUL HIDAYAH BT MAKSOM

FACULTY OF ELECTRICAL ENGINEERING

UNIVERSITY TEKNOLOGI MARA

MALAYSIA

ACKNOWLEDGMENT

Many people have contributed to my education through their guidance and support. I especially wish to thank my final year project supervisor, Pn. Tuan Norjihan Tuan Yaakob for his suggestion and ideas on research. She also reviewed my manuscript carefully. This thesis cannot be done without his help and support.

I would like to acknowledge Miss Shalwani Mohd Nor of master degree in electrical engineering, for assisted me in synthesize and simulation process. She also reviewed my Verilog HDL code.

Thoroughly I would like to thank my colleagues for their assistance and support. Finally, I would like to thank my husband and parents for their love, warmth and encouragement.

ABSTRACT

This paper presents to design Verilog code double precision floating point unit using the arithmetic operation. A method for representation of double precision floating-point numbers with four common operations is proposed in this project. Sometimes the computer result of a calculation that reflects some error was made. Possibly the magnitude of the result of a calculation was larger or smaller than this format would seem to be able to support. Possibly attempted to divide by zero and trying to represent zero. These issues is special cases of floating point numbers, specifically when the exponent field is all 1 bits (2047) or all 0 bits (0).Double precision floating point unit will divide into two numbers such as normalize and denormalize number to find out the implicit leading of the accuracy answer. The simple mathematical equation use to solve the various problem in computer design more accuracy. Simple operation equations with the 64 bits floating point were synthesized and analyze using Xilinx ISE software. Some areas of project are simulating Verilog code using ISim simulator to get the answer of operation from the wave shown.

CONTENTS

Acknowledgement.....iv

Abstract.....v

1. **Introducton**.....1
   1.1. Introduction.....1
   1.2. Objectives.....2
2. **Literature review**.....3
   2.1. Introduction.....3
   2.2. Double precision floating point case.....4
      2.2.1. Denomalize number.....4
      2.2.2. Zero.....5
      2.2.3. Infinity.....5
      2.2.4. NaN (Not a number).....5
   2.3. Summary of special case.....6
3. **Methodology**.....7
   3.1. Double precision floating point arithmetic.....7
      3.1.1. Addition.....8
      3.1.2. Subtraction.....8
      3.1.3. Multiplication.....8
      3.1.4. Division.....9
4. **Result and discussion**.....13
   4.1. Introduction.....13
      4.1.1. Top level module.....13
      4.1.2. Addition module.....15
      4.1.3. Subtraction module.....15
      4.1.4. Multiplication module.....15

CHAPTER 1

INTRODUCTION

1.1 INRODUCTION

Floating point arithmetic is widely used in many applications across multiple market segments. These applications frequently need a large number of calculations and are frequent in financial analytics, seismic imaging, radar, molecular dynamics, and bioinformatics. Apart from integer and single-precision 32 bits floating-point arithmetic, many applications demand higher precision, forcing the use of double-precision 64 bits operations [1].

Double precision floating point unit forms a significant component of many reassemble computing applications. All of the double precision floating point unit can be generated with a variable number of 64 bits for the sign, mantissa and exponent. In digital systems it used to represent numbers over a much larger range than would be possible in a fixed point representation. In IEEE 754-2008, the value of a finite binary floating-point number is calculated as (-1)sign × C × 2(E-bias), where sign is a single bit that indicates the sign of the number, C is the mantissa, E is the exponent value, and bias is a predetermined value used allow the positive E value represent exponents symmetric around 0 [2].

Software tool for synthesis and analysis of HDL designs is a Xilinx ISE, which enables the developer to synthesize the designs, perform timing analysis, examine RTL