THESIS TITLE

FName LName

Submitted in Fulfilment of the Requirements for the Degree of
Doctor of Philosophy

University of Glasgow
College of Science and Engineering
School of Computing Science

September 2014

Abstract

Write abstract here..

Dedication

To my Mother and Father.

Acknowledgements

Write here...

Declaration

I declare that, except where explicit reference is made to the contribution of others, that this dissertation is the result of my own work and has not been submitted for any other degree at the University of Glasgow or any other institution.

Write your name here

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Context

How to use Macros. The Glasgow Haskell Compiler (GHC) is write.

How to cite. According to Akhter and Roberts [1] write. According to Shende et al. [2] write.

How to use glossaries. The number of Processing Elements (PEs) write. The Haskell on a Shared Memory Multiprocessor (GHC-SMP) is write.

1.2 Thesis Statement

1.3 Contributions

1. **Write.**. detail here.

2. **Write.**. detail here.

1.4 Authorship and Publications

1.5 Thesis Structure

The structure of this thesis is as follows:

Chapter 2 gives....

Chapter 2

Background

2.1 Introduction

2.2 Parallel Architectures

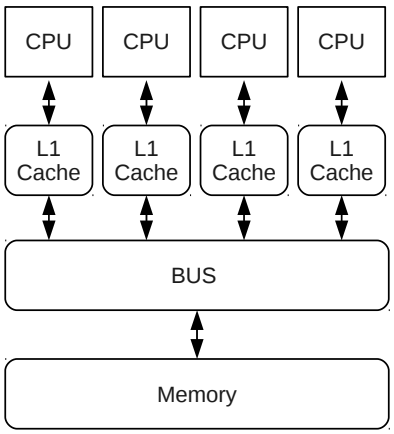

Figure 2.1: Shared Memory SMP Architecture

2.3 Summary

Chapter 3

Conclusion and Future Work

3.1 Summary

3.2 Limitations

3.3 Future Work

Appendix A

Benchmarks

Listing A.1: Fibonacci Benchmark Implementation.

```
1
2 -- / sequential Fibonacci
3
4 fib :: Int -> Integer
5 fib n | n <= 1    = 1
6       | otherwise = fib (n-1) + fib (n-2)
7
8
9 -- / parallel Fibonacci; shared memory
10
11 par_fib :: Int -> Int -> Par Integer
12 par_fib seqThreshold n
13   | n <= k    = force $ fib n
14   | otherwise = do v <- new
15                    let job = par_fib seqThreshold (n - 1) >>=
16                              force >>=
17                              put v
18                    fork job
19                    y <- par_fib seqThreshold (n - 2)
20                    x <- get v
21                    force $ x + y
22   where k = max 1 seqThreshold
```

A.1 Sum Euler

Listing A.2: Sum Euler Benchmark Implementation.

```
1 -- / Euler's totient function (for positive integers)
2
3 totient :: Int -> Integer
4 totient n = toInteger $ length $ filter (\ k -> gcd n k == 1) [1 .. n]
5
6 -- / sequential sum of totients
7
8 sum_totient :: [Int] -> Integer
9 sum_totient = sum . map totient
```

Glossary

GHC-SMP Haskell on a Shared Memory Multiprocessor.

PE Processing Element.

Bibliography

[1] S. Akhter and J. Roberts. *Multi-Core Programming: Increasing Performance Through Software Multi-threading*. Richard Bowles, 2006.

[2] S. Shende, A. D. Malony, J. Cuny, P. Beckman, S. Karmesin, and K. Lindlan. Portable Profiling and Tracing for Parallel, Scientific Applications Using C++. In *Proceedings of the SIGMETRICS Symposium on Parallel and Distributed Tools*, SPDT '98, pages 134–145, New York, NY, USA, 1998. ACM.