Standard front page

IT University of Copenhagen

Course exam with submission

Class code:

Name of course:

Course manager:

Course e-portfolio:

Thesis/project exam with project agreement

Supervisor:

Peter Sestoft

Thesis or project title:

CUDA GPGPU programming using F#

Alea.cuBase and Actulus CalcSpech for

parallelized pension reserve estimation

Name(s):	Birthdate and year:	ITU-mail:
1. Nicolai Skovvart	23/12-88	nbsk@
2.		@
3.		@
4.		@
5.		@
6.		@
7.		@

CUDA GPGPU programming using F# Alea.cuBase and Actulus CalcSpec for parallelized pension reserve estimation

Nicolai Skovvart nbsk@itu.dk
Superviser: Peter Sestoft

IT University of Copenhagen

Abstract. Parallelization is especially useful and worthwhile in large-scale numerical computations in which a vast amount of independent, relatively simple, equations need to be solved, for example in estimating required pension reserves. In this thesis we will consider such problems drawn from pension and life insurance mathematics, and in particular how to estimate required pension reserves and cashflows, within the framework of the Actulus research project. Such work in particular can make use of the power of GPGPU's and the CUDA C programming languages.
CUDA C is, however, based on the somewhat archaic C language. For safety and flexibility industry wants to automatically generate GPGPU code and to control its execution via the .NET platform.
This project aims to allow for such calculations to be written in the F# language on the .NET platform using Alea.cuBase which in turn is translated into CUDA C (or PTX, the CUDA C bytecode). Furthermore, I hope to implement a parser of the Actulus CalcSpec language files to be translated to F# kernel-representations. This will allow for business to stick to a .NET workflow while using a modern language (F#) and avoid having to work directly with (CUDA) C.
The generated F# CUDA code will be evaluated to to both an existing single-threaded C# application provided by Peter Sestoft, manually written F# cuBase code and manually written CUDA code.
Through this project I hope to learn how to write high-performance GPGPU software for numerical problems, to demonstrate that I can design, implement, describe and critically evaluate the performance and other quality aspects of such software.

Keywords: CUDA, parallelization, F# Alea.cuBase, pension reserve estimation, Actulus CalcSpech

Table of Contents

1 Introduction 3
2 Background 3
  2.1 Parallelization and the GPU 3
  2.2 Runge-Kutta and Thiele's differential equation 3
  2.3 CUDA 3
  2.4 F# Alea.cuBase 3
3 Solutions 3
  3.1 Initial C# Solution 3
  3.2 CUDA Solution 3
  3.3 F# Alea.cuBase Solution 3
4 Actulus CalcSpec to cuBase Code Generation 3
5 Optimization 4
6 Related Work 4
7 Conclusion 4

Appendices 5
A Test 5

List of Figures

1 Introduction

Content

2 Background

Background introduction

2.1 Parallelization and the GPU

Talk about GPU vs CPU and parallelization in general

2.2 Runge-Kutta and Thiele's differential equation

Runge-Kutta 4 solvers and Thiele's differential equation section[1]

2.3 CUDA

CUDA section

2.4 F# Alea.cuBase

F# Alea.cuBase section

3 Solutions

3.1 Initial C# Solution

Describe original (C#) solution

3.2 CUDA Solution

Describe implemented CUDA solution

3.3 F# Alea.cuBase Solution

Describe implemented F# Alea.cuBase solution

4 Actulus CalcSpec to cuBase Code Generation

Describe how Actulus CalcSpec is translated to F# and its effect on performance

5 Optimization

Describe various optimization strategies and their effect on the project. Lots of graphs and stuff.

6 Related Work

Related work, include Dahl+Harrington for sure

7 Conclusion

Conclude

References

1. Actulus.dk. Actulus, 2013. `http://www.edlund.dk/da/forskning/actulus/`.

A Test

Hey