# Introduction

This report aims to serve as an analysis and documentation for the process of parallelisation of the maxwell source code provided by the assignment. Three versions of parallelisation are needed for this assignment, and these are done following three programming models: with the help of OpenMP, The Message Passing Interface (MPI) and CUDA. This document contains information about the attempts made by the developer to parallelise with the given programming models, how the validation of each programming model was done, and if it is valid, the testing environment alongside with the results produced and an evaluation of the performance each version created produces, to the degree of completion each version it is.

# Parallelisation Approach

Parallel computing/programming is a computer programming technique that enables parallel execution of operations. It uses multiple processors in parallel to solve problems more quickly than with a single processor. If you cannot increase the clock, do more operations by one clock. But, we cannot build an infinite processor due to temperature, cooling problems, interconnect bottleneck, etc. Performance gained by multicore processor strongly dependent on the software algorithms and implementation. [1]

## On node parallelisation – OpenMP

## Approach taken for the parallelisation of MPI

## Approach taken for the parallelisation of the CUDA model

# Validation

The validation process used to verify the correctness of your applications.

The validity of each programming model was done by comparing the log files produced by each version against the log file produced by the original code;

The most important values that were compared where those of the variables called: E magnitude and B magnitude from each 100 steps until the end of the execution. which also had a timer added to it with the help of the time.h library;

# Experimental Setup

A summary of the systems used for performance evaluation and an account of the process of collecting results.

The experiments were conducted on different setups.

The setups are as follows:

The university teaching laboratory which its hardware consists of:

The second experimental setup on which all the programming versions were tested on is the university’s supercomputer Viking.

And the machines CSE066 for CUDA testing, which consists of:

# Performance Evaluation

Appropriate data demonstrating the performance and scaling behaviour of your applications.

The performance was tested by doing a comparison between the execution time of each version of the program;

In theory, if all the parallelisation approaches were completely functional, the rankings of the performance should be around:

[1]

# Comparative Analysis – Conclusion for it

A comparative analysis of your three applications. And the conclusion.

# References

[1] K. S. Mohamed, ‘Parallel Computing: OpenMP, MPI, and CUDA’, in *Neuromorphic Computing and Beyond: Parallel, Approximation, Near Memory, and Quantum*, K. S. Mohamed, Ed. Cham: Springer International Publishing, 2020, pp. 63–93. doi: 10.1007/978-3-030-37224-8\_3.