Chip Multiprocessors

Todd Davies

February 1, 2016

Overview

Due to technological limitations, it is proving increasingly difficult to maintain a continual increase in the performance of individual processors. Therefore, the current trend is to integrate multiple processors on to a single chip and exploit the resulting parallel resources to achieve higher computing power. However, this may require significantly different approaches to both hardware and software particularly for general purpose applications. This course will explore these issues in detail.

Syllabus

Introduction Trends in technology, limitations and consequences. The move to multi-core parallelism in programs, ILP, Thread Level, Data Parallelism.

Parallel Architectures SIMD, MIMD, Shared Memory, Distributed Memory, strengths and weaknesses.

Parallel Programming Multithreaded programming, Data parallel programming, Explicit vs Implicit parallelism, automatic parallelisation. The Case for Shared Memory. When to share?

Shared Memory Multiprocessors Basic structures, the cache coherence problem. The MESI protocol. Limitations. Directory based coherence.

Programming with Locks and Barriers The need for synchronisation. Problems with explicit synchronisation

Other Parallel Programming Approaches MPI and OpenMP

Speculation The easy route to automatic parallelisation?

Transactional Memory Principles. Hardware and Software approaches

Memory Issues Memory system design. Memory consistency

Other Architectures and Programming Approaches GPGPUs, CUDA

Data Driven Parallelism Dataflow principles and Functional Programming

Attribution

These notes are based off of both the course notes (`http://studentnet.cs.manchester.ac.uk/ugt/2015/COMP35112/`). Thanks to John Gurd for such a good course! If you find any errors, then I'd love to hear about them!

Contribution

Pull requests are very welcome: `https://github.com/Todd-Dav`
`third-year-notes`

Contents