

NCT-100 Programming Multicore Processors

Three day intensive training course covering all aspects of programming multicore processors using advanced methods and techniques

Course Overview

This course covers concepts and approaches related to programming multicore processors in C/C++. From recognizing parallelism opportunities, to multithreaded programming and designing multithreaded algorithms, this course teaches you how to deal with the many aspects of synchronization, multithreading libraries, processor caches, shared memory and specialized methods for multithreaded programming using modern approaches such as OpenMP and Intel Threading Building Blocks.

Course Objectives

- In-depth theoretical background covering multicore processor architecture, concurrent programming, parallel programming concepts and considerations.
- Critical concepts such as implicit and explicit parallelism, atomicity, synchronization, shared memory, cache coherency, Amdahl's Law, Flynn's processor classifications and Little's law are covered in-depth.
- Recognize the best parallelism opportunities and explain the advantages of using threads to obtain concurrency using various analysis techniques, compositional approaches and parallel design patterns.
- How to avoid synchronization pitfalls such as starvation, deadlock, live lock and data races.
- How to implement and tune parallel algorithms.
- Define and use different synchronization methods effectively including mutexes/critical sections and conditional variables.
- Explain operating system interactions and the relationship between shared memory and threads.
- Explain what aspects of the operating system affect programming, how to deal with shared memory effectively, CPU selection, CPU-specific binding of threads, thread specific data and kernel-level scheduling.
- Understand and use threads with specific technologies and programming methods such as the Windows API, POSIX pthreads, Intel TBB and OpenMP using C/C++ and the Intel Compiler.

Benefits

- A comprehensive training workshop: This course offers an in-depth overview of fundamental concepts while offering advanced training and practical advice on C/C++ programming of multicore processors using modern methods.
- Gain critical insights on how to improve your software's performance: This course is designed to give you key skills and using specialized tools to help you to correctly architect, design and develop efficient parallel applications for multicore processors.
- Additional hands-on learning: This course provides laboratory sessions in writing multithreaded programs and exercises on practical parallelization of legacy software. It also includes walk-through laboratory exercises designed to increase your understanding of multithreading.

Who should attend

Software architects, software developers, software team leaders and managers seeking to understand and implement efficient software running multicore processors. Knowledge of the C++ programming language and C++ software development experience is a pre-requisite for this course.

About nCore

nCore Design LLC develops high performance software and systems for multicore processors, GPUs and embedded systems. nCore has significant expertise on wide variety of computing platforms and specialized knowledge of applied software optimization techniques. nCore is a leading provider of hands-on, instructor-led training courses focused on programming multicore and GPU processors.

nCore is a privately held company with offices in the US and Japan. nCore works with local partners to deliver state-of-the-art solutions to multinational technology companies.

Schedule - Day One

Part One

- **Introduction**
- **Motivation** — The march to multicore
- **Architecture** — parallelism, processors, shared memory, Little's law, Flynn's classification
- **Concurrent programming** — synchronization, atomicity, dead locks and data races
- **Parallel programming concepts** — shared and distributed memory, performance — scalability & granularity vs. performance, Amdahl's Law, examples
- **Programming considerations** — parallel analysis, decomposition, guidelines, re-engineering

Part Two

- **Multicore computers** — architecture, SMT, memory, caches
- **Multicore computers** — programming, affinity, process migration, Linux kernel scheduler
- **Shared memory programming in C++**
- **Introduction to OpenMP and TBB**
- **Exercise:** Parallelization of an application using OpenMP

Schedule - Day Two

Part One

- **Multithreading APIs (Windows/POSIX/TBB)** — Creation and destruction
- **Exercise:** Creation of N threads
- **Synchronization** — API comparison and C++
- **Exercise:** Synchronization using a mutex
- **Exercise:** Synchronization using a conditional variable
- **Private thread data**

Part Two

- **Thread unsafe APIs**
 - STL
 - Thread safe wrappers
 - Exercise: Asynchronous queue
- **Exercise:** Producer consumer
- **Exercise:** Parallelization of an independent application using N threads
- **Comparison with OpenMP version**

Schedule - Day Three

Part One

- **Reader Writer** — description and implementation
- **Exercise:** Implementation and test
- **Application:** multi-thread efficient memory allocator
- **Pipeline** — description and implementation
- **Exercise:** Implementation and test

Part Two

- **Thread pool**
 - Exercise: implementation and test
- **Lock free algorithms**
 - Exercise: implement a producer
 - Consumer using the previous asynchronous queue and a provided lock
 - Free queue
- **Multicore Performance**
- **Relevant Tools**
- **Profilers, Debuggers, Intel Thread checker**

Course Registration



<http://www.ncoredesign.com/training/>

Length: 3 Days

Cost: \$2495

Registration Methods

Online: <http://www.ncoredesign.com/training/>

By E-mail: training@ncoredesign.com

By Fax: 971-228-0707

