Publication date : 02/09/2024

Course : C/C++, programming applications in Multicore

Practical course - 3d - 21h00 - Ref. MUC
Price : 1940 CHF E.T.

4,7 / 5

You'll learn about multicore architectures and their programming, techniques for implementing a multithread or multiprocess approach, and languages dedicated to parallel programming. You'll also learn about data access synchronization constraints and the precautions to be taken.

Teaching objectives

At the end of the training, the participant will be able to:

- ✓ Mastering the challenges of multicore programming
- ✓ Design and develop thread- and process-based applications
- ✓ Master parallel programming models and available libraries
- ✓ Debugging and profiling Multicore applications

Intended audience

Developers, software architects, project managers.

Prerequisites

Good knowledge of C or C++. Basic knowledge of Multicore application concepts.

Practical details

Hands-on work

Practical work will be carried out in C/C++ under Visual Studio in a Windows environment.

Course schedule

PARTICIPANTS

Developers, software architects, project managers.

PREREQUISITES

Good knowledge of C or C++. Basic knowledge of Multicore application concepts.

TRAINER QUALIFICATIONS

The experts leading the training are specialists in the covered subjects. They have been approved by our instructional teams for both their professional knowledge and their teaching ability, for each course they teach. They have at least five to ten years of experience in their field and hold (or have held) decision-making positions in companies.

ASSESSMENT TERMS

The trainer evaluates each participant's academic progress throughout the training using multiple choice, scenarios, hands-on work and more.
Participants also complete a placement test before and after the course to measure the skills they've developed.

1 Introduction

- The challenges of multicore programming.
- Table of usable technologies: process, thread and parallelism.
- Description of how a processor works.
- Architecture in "Hyperthreading".
- INTEL and AMD processor architectures.
- NVidia architectures and APIs.
- Shared vs. distributed memory architecture.

2 Application modeling

- Importance of modeling aspects.
- Parallel processing patterns.
- Use of asynchronous mechanisms.
- Developing a new application: precautions and modeling. Avoiding "singletons".
- Modify an existing application in Multicore.
- Choice of architecture: a compromise between synchronization and performance. Multiprocess/multithread choices.

3 Threads

- Threads in an industrial application.
- Thread scheduling.
- Management of stacks and "call stack" in threads.
- Multithreaded debuggers.
- Management of synchronization objects: critical sections, Mutexes and Semaphores.
- Develop "thread safe".
- Thread API TBB, Clik++, C++11, boost threads, pthreads.

Hands-on work

Threads and synchronization in C/C++.

4 Process

- Process address spaces, organization.
- Criteria for choosing a multi-process approach.
- Inter-process communication (IPC) techniques.
- Multi-process debugging tools.
- Advantages and disadvantages of multi-process techniques.

Hands-on work

Manage asynchronous processing with the C/C++ API.

TEACHING AIDS AND TECHNICAL RESOURCES

• The main teaching aids and instructional methods used in the training are audiovisual aids, documentation and course material, hands-on application exercises and corrected exercises for practical training courses, case studies and coverage of real cases for training seminars.
• At the end of each course or seminar, ORSYS provides participants with a course evaluation questionnaire that is analysed by our instructional teams.
• A check-in sheet for each half-day of attendance is provided at the end of the training, along with a course completion certificate if the trainee attended the entire session.

TERMS AND DEADLINES

Registration must be completed 24 hours before the start of the training.

ACCESSIBILITY FOR PEOPLE WITH DISABILITIES

Do you need special accessibility accommodations? Contact Mrs. Fosse, Disability Manager, at psh-accueil@orsys.fr to review your request and its feasibility.

5 Parallel programming

- Contribution and objectives of parallel programming.
- The "OpenMP" C++ library (shared memory programming).
- The "OpenMPI" library (distributed memory programming).
- Use GPUs on graphics cards for calculation.
- Kits from NVidia (CUDA) and ATI.
- The "OpenAcc" library for GPU programming.
- The "OpenCL" library for CPU and GPU parallel programming.

Hands-on work

Parallelizing algorithms with "OpenMP" in C++. Using the OpenCL API.

6 Summary and conclusion

- Conclusion of the techniques studied.
- The future of C++ with multicore.

Dates and locations

REMOTE CLASS
2026 : 1 June, 16 Sep., 16 Dec.