Stand 05.04.2026

Multithread-/Multicore-Programmierung - Präsenz-Training

**Ziele -
Ihr Nutzen**

Dieses Training zeigt, wie die Eigenschaften von Multitask-/Multithread-Systemen eingesetzt werden können bzw. welche Probleme gelöst werden müssen, um die Prozessorleistung, insbesondere bei modernen Multicore-Systemen, bestmöglich nutzen zu können.

Es richtet sich an Entwickler, die existierende Applikationen optimieren, oder neue Applikationen entwerfen und implementieren müssen.

Die Themen werden allgemeingültig behandelt, so dass das erworbene Wissen prinzipiell auf jede Multithreading- oder Multitasking-Plattform anwendbar ist.

Gleichzeitig werden die Lehrinhalte durch eine Fülle von C- und C++-Code-Beispielen konkretisiert.

Dabei werden sowohl plattformspezifische Lösungen, basierend auf Windows, Linux und RTEMS, als auch plattformunabhängige Lösungen auf Basis der C++-Multithread-Bibliothek vorgestellt, die seit C++11 zur Verfügung steht.

Teilnehmer

Software-Entwickler, Software-Architekten, Projektleiter.

Voraussetzungen

Kenntnis der Programmiersprache C. Da auch viele C++-Beispiele gezeigt werden, sind C++-Kenntnisse hilfreich, aber nicht Voraussetzung. Somit eignet sich dieser Kurs auch für Teilnehmer, die ausschließlich mit C programmieren.

Multithread-/Multicore-Programmierung - Präsenz-Training

Inhalt

Multitask-/Multithread Grundlagen

- Konzept Task/Thread (allg. Thread)
- Thread vs. Prozess
- Zuordnung von Code und Daten
- Scheduler und Thread-Zustandsmodell
- Allgemeine Schedulingmodelle
- Grundsätzliche Schedulingmechanismen
- Singlecore-/Multicore-Multithreading

Thread-Programmierung (1)

- Threads und Funktionen
- Threads und Objekte
- Erzeugen und Starten von Threads
- Der Stack eines Threads
- Threadpriorität
- Sonstige Konfigurationsparameter
- Parameterübergabe
- Thread-Beendigung

Synchronisation (1)

- Synchronisation mit Flag

- Volatile
- Speichermodell
- Speicherbarrieren
- Polling
- Ereignissynchronisation (Events)
- Zugriffssynchronisation (Mutexe)
- Atomare Operationen

Thread-Programmierung (2)

- Threadlokaler Speicher (TLS)
- Thread-Unterbrechung (Signale, Asynchrone Service Routinen)
- Thread-Terminierung (Thread-Cancellation)

Synchronisation (2)

- Typische Probleme und Lösungsansätze
- Deadlocks und Livelocks
- Mutex-Implementierungsdetails mit Laufzeitauswirkungen
- Zugriff auf Ressourcen mit mehreren Einheiten (Semaphore)
- Ressource-Zugriff mit Ereignisauslösung (Condition Variable)
- Synchronisation von Threads mit Interrupt Service Routinen

Multicore-Programmierung

- Wann lohnt sich der Einsatz von Multicore?
- Beispiel: Laufzeitmessungen klassischer Lösungen für ein Synchronisationsproblem
- Lösung mithilfe von Task-Variablen
- False Sharing
- Prozessor-/Thread-Affinität
- Spinlocks
- Portierung von Singlecore-Applikationen auf Multicore-Systeme
- Wichtige Kriterien für gutes Multithread-Design

MicroConsult Plus:

- Sie erhalten von uns Ihre Übungsverzeichnisse und Lösungsbeispiele für alle Übungsaufgaben.

Präsenz-Training

Preis *	Dauer
-	3 Tage

Anmeldecode: MMP
* Preis je Teilnehmer, in Euro zzgl. USt.

Live-Online - Deutsch

Dauer
3 Tage

Coaching

Unsere Coaching-Angebote bieten den großen Vorteil, dass unsere Experten ihr Wissen und ihre Erfahrungen direkt in Ihren Lösungsprozess einbringen und damit unmittelbar zu Ihrem Projekterfolg beitragen.

Für Ihre Anfrage oder weiterführende Informationen stehen wir Ihnen gern zur Verfügung.