

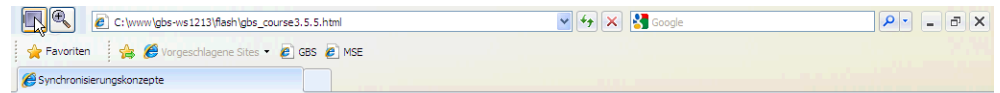
Script generated by TTT

Title: Grundlagen_Betriebssysteme (14.11.2012)

Date: Wed Nov 14 13:20:41 CET 2012

Duration: 41:05 min

Pages: 13



Ziel : Einführung wichtiger Realisierungskonzepte zur Synchronisierung paralleler Abläufe. Dazu Konkretisierung des Prozessbegriffs.

Prozess - Konkretisierung

Konzepte für wechselseitigen Ausschluss

Die Netz-Modellierung hat bereits gezeigt, dass man zur Synchronisation von Prozessen spezifische **Kontrollkomponenten** benötigt (z.B. zusätzliche Stellen im Petri-Netz, oder Kapazitätsbeschränkungen, die implizit durch die abstrakte Kontrollkomponente, die die Transitionsbereitschaft von Transitionen prüft, kontrolliert werden).

Anforderungen

Jede Realisierung des w.A. benötigt eine **Basis**, mit festgelegten **atomaren**, d.h. nicht teilbaren Operationen.

Unterbrechungssperre

Test-and-Set Operationen

Dienste mit passivem Warten

Semaphor-Konzept

Das Semaphor-Konzept ermöglicht die Realisierung des w.A. auf einem höheren Abstraktionslevel als die bereits angesprochenen Hardwareoperationen.

Monitor-Konzept

Generated by Targeteam



Anforderungen



Folgende Anforderungen sind an eine Realisierung des wechselseitigen Ausschlusses (**w.A.**) zu stellen:

Die kritischen Abschnitte der Prozesse sind wechselseitig auszuschließen.

Eine Realisierung des w.A. darf nicht von einer Annahme über die Reihenfolge, in der die Prozesse ihre kritischen Abschnitte ausführen, abhängen.

Eine Realisierung des w.A. darf nicht von Annahmen über die Ausführungszeit der Prozesse abhängen.

Unter der Annahme, dass Prozesse nach endlicher Zeit ihre kritischen Abschnitte wieder verlassen, muss gelten, dass kein Prozess einen anderen Prozess unendlich lange daran hindern darf, seinen kritischen Abschnitt auszuführen.



Generated by Targeteam



Ziel : Einführung wichtiger Realisierungskonzepte zur Synchronisierung paralleler Abläufe. Dazu Konkretisierung des Prozessbegriffs.

Prozess - Konkretisierung

Konzepte für wechselseitigen Ausschluss

Die Netz-Modellierung hat bereits gezeigt, dass man zur Synchronisation von Prozessen spezifische **Kontrollkomponenten** benötigt (z.B. zusätzliche Stellen im Petri-Netz, oder Kapazitätsbeschränkungen, die implizit durch die abstrakte Kontrollkomponente, die die Transitionsbereitschaft von Transitionen prüft, kontrolliert werden).

Anforderungen

Jede Realisierung des w.A. benötigt eine **Basis**, mit festgelegten **atomaren**, d.h. nicht teilbaren Operationen.

Unterbrechungssperre

Test-and-Set Operationen

Dienste mit passivem Warten

Semaphor-Konzept

Das Semaphor-Konzept ermöglicht die Realisierung des w.A. auf einem höheren Abstraktionslevel als die bereits angesprochenen Hardwareoperationen.

Monitor-Konzept

Generated by Targeteam



Unterbrechungssperre



Der Ablauf des Prozesses darf nicht unterbrochen werden.

In Ein-Prozessorsystemen (1 CPU) kann es ausreichend sein, mit **Enable** und **Disable Interrupt** Operationen dafür zu sorgen, dass der Prozess, der einen kritischen Abschnitt ausführt, dabei nicht unterbrochen wird.

Realisierung mit Unterbrechungssperre ist nützlich für den Betriebssystemkern, aber sollte nicht für allgemeine Benutzerprogramme zur Verfügung stehen.

Generated by Targeteam



Test-and-Set Operationen (Test-And-Set-Lock, TSL) erlauben auf Hardware-Ebene (Maschinenbefehl) das Lesen und Schreiben einer Speicherzelle als atomare Operation.

Semantik eines Test-and-Set-Befehls

Die Instruktion liest den Inhalt eines Speicherwortes in ein Register und schreibt einen Wert ungleich Null an die Adresse dieses Wortes. Diese beiden Operationen werden als atomare, unteilbare Sequenz ausgeführt.

Sei a die zu untersuchende Speicherzelle; der Wert 1 in der Speicherzelle kann dahingehend interpretiert werden, dass der Zugang in den kritischen Bereich nicht möglich ist.

Befehl "TSL R, a" entspricht dann

```
Register R = inhalt von a;
```

```
a = 1
```

Falls $R = 1$ ist, ist der kritische Bereich bereits belegt, andernfalls war er frei und er wurde vom ausführenden Prozess belegt.

Problem: wie Atomarität bei Rechensystem mit mehreren CPUs gewährleistet?

Lösung: Die CPU, die die TSL-Instruktion ausführt, blockiert den Speicherbus, um andere CPU's (Multi-Prozessorumgebung) daran zu hindern, auf die Speichereinheit zuzugreifen.

Generated by Targeteam



Synchronisierungskonzepte



Ziel: Einführung wichtiger Realisierungskonzepte zur Synchronisierung paralleler Abläufe. Dazu Konkretisierung des Prozessbegriffs.

Prozess - Konkretisierung

Konzepte für wechselseitigen Ausschluss

Die Netz-Modellierung hat bereits gezeigt, dass man zur Synchronisation von Prozessen spezifische **Kontrollkomponenten** benötigt (z.B. zusätzliche Stellen im Petri-Netz, oder Kapazitätsbeschränkungen, die implizit durch die abstrakte Kontrollkomponente, die die Transitionsbereitschaft von Transitionen prüft, kontrolliert werden).

Anforderungen

Jede Realisierung des w.A. benötigt eine **Basis**, mit festgelegten **atomaren**, d.h. nicht teilbaren Operationen.

Unterbrechungssperre

Test-and-Set Operationen

Dienste mit passivem Warten

Semaphor-Konzept

Das Semaphor-Konzept ermöglicht die Realisierung des w.A. auf einem höheren Abstraktionslevel als die bereits angesprochenen Hardwareoperationen.

Monitor-Konzept

Generated by Targeteam



Unterscheidung zwischen

aktivem Warten: Prozess muss periodisch selber prüfen, ob die Voraussetzungen zum Weiterarbeiten erfüllt sind.

passivem Warten: Prozess wird in Warte-Zustand versetzt und aufgeweckt, wenn das Ereignis, auf das er wartet, eingetreten ist.

Methoden oder Dienste, so dass unter Mithilfe des Betriebssystems ein Prozess in den Zustand **wartend** übergeführt wird.

beim Eintreten eines passenden "Weckereignisses" wird der Prozess vom Betriebssystem vom Zustand **wartend** in den Zustand **rechenbereit** übergeführt.

Beispiel: Java wait und notify.

Generated by Targeteam



Synchronisierungskonzepte



Ziel: Einführung wichtiger Realisierungskonzepte zur Synchronisierung paralleler Abläufe. Dazu Konkretisierung des Prozessbegriffs.

Prozess - Konkretisierung

Konzepte für wechselseitigen Ausschluss

Die Netz-Modellierung hat bereits gezeigt, dass man zur Synchronisation von Prozessen spezifische **Kontrollkomponenten** benötigt (z.B. zusätzliche Stellen im Petri-Netz, oder Kapazitätsbeschränkungen, die implizit durch die abstrakte Kontrollkomponente, die die Transitionsbereitschaft von Transitionen prüft, kontrolliert werden).

Anforderungen

Jede Realisierung des w.A. benötigt eine **Basis**, mit festgelegten **atomaren**, d.h. nicht teilbaren Operationen.

Unterbrechungssperre

Test-and-Set Operationen

Dienste mit passivem Warten

Semaphor-Konzept

Das Semaphor-Konzept ermöglicht die Realisierung des w.A. auf einem höheren Abstraktionslevel als die bereits angesprochenen Hardwareoperationen.

Monitor-Konzept

Generated by Targateam



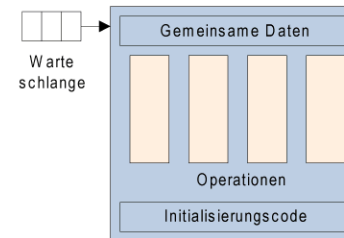
Monitor-Konzept



Das Monitor-Konzept basiert auf der Idee, die in einem kritischen Bereich bearbeiteten Daten zusammen mit den darauf definierten Zugriffsoperationen in einer sprachlichen Einheit - dem Monitor - zusammenzufassen.

soll Probleme, wie sie bei der Programmierung von Semaphoren auftreten, vermeiden.

zu einem Zeitpunkt höchstens ein Prozess innerhalb des Monitors aktiv.



Definition von **Bedingungsvariablen** zur Spezifikation anwendungsspezifischer **Bedingungen**.

Operationen auf **Bedingungsvariable cond**

`cond.wait()`: Prozess wird wartend gesetzt.

`cond.signal()`: ein wartender Prozess wird aktiviert.

Generated by Targateam



Synchronisierungskonzepte



Ziel: Einführung wichtiger Realisierungskonzepte zur Synchronisierung paralleler Abläufe. Dazu Konkretisierung des Prozessbegriffs.

Prozess - Konkretisierung

Konzepte für wechselseitigen Ausschluss

Die Netz-Modellierung hat bereits gezeigt, dass man zur Synchronisation von Prozessen spezifische **Kontrollkomponenten** benötigt (z.B. zusätzliche Stellen im Petri-Netz, oder Kapazitätsbeschränkungen, die implizit durch die abstrakte Kontrollkomponente, die die Transitionsbereitschaft von Transitionen prüft, kontrolliert werden).

Anforderungen

Jede Realisierung des w.A. benötigt eine **Basis**, mit festgelegten **atomaren**, d.h. nicht teilbaren Operationen.

Unterbrechungssperre

Test-and-Set Operationen

Dienste mit passivem Warten

Semaphor-Konzept

Das Semaphor-Konzept ermöglicht die Realisierung des w.A. auf einem höheren Abstraktionslevel als die bereits angesprochenen Hardwareoperationen.

Monitor-Konzept

Generated by Targateam



Synchronisation



Eine wichtige Systemeigenschaft betrifft die Synchronisation paralleler Ereignisse. Mehrere Prozesse konkurrieren um eine gemeinsame **Ressource (CPU)**, oder greifen auf gemeinsame Daten zu.

Beispiele

Die beiden Beispiele basieren auf der speicherbasierten Prozessinteraktion, d.h. Prozesse (oder auch Threads) interagieren über gemeinsam zugreifbare Speicherzellen.

Beispiel: gemeinsame Daten

Erzeuger-Verbraucher-Problem

Definition: Wechselseitiger Ausschluss

Modellierung

Synchronisierungskonzepte

Semaphore

Synchronisierung von Java Threads

Generated by Targateam



Die Operationen P und V sind **atomar**; sie sind unteilbar und sie werden wechselseitig ausgeschlossen ausgeführt.

Sei s die Koordinierungsvariable, dann sind die P und V Operationen wie nachfolgend definiert.

Mögliche Realisierung auf Hardware-Ebene mittels des Test-and-Set Maschinenbefehls.

Informelle Charakterisierung

```
public void P (int s) {
    s = s - 1;
    if ( s < 0 ) { Prozess in die Menge der bzgl. s wartenden Prozesse
    einreihen }
}

public void V (int s) {
    s = s + 1;
    if ( s ≤ 0 ) { führe genau einen der bzgl. s wartenden Prozesse in den
    Zustand rechenwillig über }
}
```

Binäres Semaphore: die Kontrollvariable s nimmt nur boolesche Werte an.

man spricht auch von Mutex.

[Mutex in Posix](#)

Generated by Targeteam



einfache Sperre zur Absicherung gemeinsamer Ressourcen

pthread_mutex_init(mutex) ⇒ initialisiert und konfiguriert ein Mutex.

pthread_mutex_lock(mutex) ⇒ entspricht der P-Operation; sperrt ein Mutex.

pthread_mutex_unlock(mutex) ⇒ entspricht der V-Operation; gibt ein Mutex frei.

Mutex Objekte und Semaphore mit Integer Werten stehen auch in Windows zur Verfügung.

Generated by Targeteam

