**Prozessverwaltung**

Begriffe

Sequenzielles Programm - besitzt genau einen Kontrollfluss

Nebenläufiges Programm - mehrere Kontrollflüsse, dürfen parallel ausgeführt werden  
 - auf Mehrprozessorarchitektur Kontrollflüsse eigenen  
 Prozessoren zugeordnet und so parallel

Nebenläufigkeit - maximale Parallelität, die mit unbegrenzt vielen Prozessoren  
 erreichbar ist

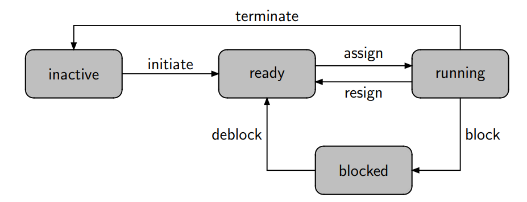
Parallelität - tatsächlich erreichte Grad an paralleler Ausführung  
 (begrenzt durch Prozessoranzahl)

Sequenzieller Prozess - stellt die Ausführungsumgebung für ein sequenzielles Programm  
 Dafür benötigt: Adressraum, das Maschinenprogramm, ein  
 Aktivitätsträger (Prozessorkern), eine Liste der belegten  
 Betriebsmittel

Nebenläufiger Prozess - stellt die Ausführungsumgebung für ein nebenläufiges Programm  
 Unterschied: mehrere Aktivitätsträger

Implementierung von Prozessen

Zustandsmodell Zustände:

- running Dem Prozess ist ein Prozessor zugeordnet

- ready Der Prozess beteiligt sich am Wettbewerb um  
 einen Prozessor (besitzt alle Betriebsmittel)

- blocked Prozess nicht ablaufbereit (wartet)

- inactive Prozess ist keinem Programm zugewiesen

Es liegt ein geschlossenes Modell vor, in dem Prozesse als Objekte auftreten. Zu jedem Zeitpunkt befindet sich ein Prozess in genau einem Zustand.

Übergangsfunktionen

block ( q ) Prozess wird blockiert und in Warteschlange q eingefügt

deblock ( q ) Prozess wird entblockt und in ready-Zustand überführt

assign ( ) Einen bereiten Prozess auswählen (Zuordnung des Prozessors)

resign ( ) Laufenden Prozess in ready überführen

initiate ( context ) Inaktiven Prozess auswählen (Initialisierung der Prozessumgebung  
 mit den Werten aus context) überführt in ready

terminate ( ) laufende Prozess gibt Prozessor ab, da Programmausführung beendet  
 Überführung in inactive

AUF BLOCK, RESIGN UND TERMINATE MUSS IMMER ASSIGN FOLGEN SONST PROZESSOR NICHT BELEGT!

Die Betriebssystemkomponente Prozessverwaltung

Dispatcher und Scheduler

- Einordnung des Prozesszustandsmodells  
 (gehört zur untersten Betriebssystemschicht (elementarer Dienst des Kerns))

Dispatcher  
 - Durchführung der Zustandsübergänge  
 - Implementierung der Übergangsfunktionen

Scheduler  
 - Aufruf beim Übergang von ready nach running  
 - Auswahl des nächsten Prozesses, der den Prozessor erhält

Prozesskontrollblock

- Darstellung eines Prozesses im Dispatcher

- Enthält: - Prozesszustand   
 - Sicherungsbereich für Registerinhalt  
 - Adressrauminformationen  
 - Scheduling-Informationen  
 - belegte Betriebsmittel (z.B. geöffnete Dateien)  
 - weitere Verwaltungsdaten

storeRegister() Abspeichern der aktuellen Registerwerte innerhalb des  
 Sicherungsbereichs des PCBs

loadRegister() Überschreiben der Registerinhalte des Prozessors mit den im PCB  
 gesicherten Werten

initPcb ( Context c ) Der Prozesskontrollblock wird mit allen notwendigen Informationen  
 initialisiert

put ( Pcb process ) Der Prozess wird an das Ende der Warteschlange angehängt

Pcb get () Der Prozess, der vorne in der Warteschlange steht, wird zurückgeliefert

Prozesswarteschlangen

Auf einen Prozess wird durch eine eindeutige PID (Process ID) verwiesen   
 Innerhalb der Prozessverwaltung wird die Referenz auf den PCB verwendet, der dem Prozess  
 zugeordnet ist.

Probleme mit nebenläufigem Code

Doppelter Abruf/Bearbeitung

Unterbrechungssperren

- Unterbrechungen werden während der Ausführung des kritischen Abschnittes verboten  
 - Bei Mehrprozessorsystemen nicht ausreichend  
 ⇒ Anderen Prozessoren könnten weitere ausgeschlossene kritische Abschnitte ausführen

Nachteile  
 - Herabgesetzte Reaktionsfähigkeit auf externe Unterbrechungen  
 - Mögliche Folge: Verlust von Daten bei E/A-Operationen   
 - Unterbrechungen können nur im privilegierten Zustand gesperrt werden  
 - Fehleranfälligkeit: Vergisst ein Prozess, die Unterbrechungen wieder zuzulassen, führt dies  
 zu einer Monopolisierung des Prozessors.

Atomare Speicheroperationen

- Falls mehrere Prozessoren gleichzeitig versuchen, einen Wert in dasselbe Speicherwort zu  
 speichern, trifft die Hardware die Entscheidung, welche zuerst ausgeführt wird.

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung

Dekker Algorithmus

Bewertung

+ Es funktioniert

- Nur zwei Prozesse

- nur kurze Abschnitte

Test&Set-Befehl

Grundlage für eine Realisierung des gegenseitigen Ausschlusses kurzer kritischer Abschnitte bei Mehrprozessorsystemen. ( → Spin Locks)

+ Eintritts- und Austrittsprotokoll sind unabhängig von der Prozessanzahl und  
 Prozessidentifikationen  
 + Durch die Parametrisierung mit einer globalen Variablen kann der gegenseitige Ausschluss  
 auf bestimmte kritische Abschnitte eingeschränkt werden

Ein Bild, das Text enthält.

Automatisch generierte Beschreibung - Aktives Warten (Busy Waiting)  
 - Ein Aushungern (Starvation) eines Prozesses kann auftreten