

<p style="text-align: right;">7 Zugriffskontrolle</p> <h2>Überblick</h2>	<p style="text-align: right;">7 Zugriffskontrolle 7.1 Konkurrenz und Koordination</p> <h2>Nebenläufige Aktivitäten</h2> <p>Nichtsequentielles Programm</p>
<p>Zugriffskontrolle</p> <ul style="list-style-type: none"> <li>Konkurrenz und Koordination</li> <li>Verdrängungssteuerung</li> <li>Prioritätsvererbung</li> <li>Prioritätsobergrenzen</li> <li>Zusammenfassung</li> </ul>	<p><b>Nebenläufigkeit</b> (engl. <i>concurrency</i>) bezeichnet das Verhältnis von nicht kausal abhängigen Ereignissen, die sich also nicht beeinflussen</p>
<p>soech WS 2005/06 E2S 7-1</p>	<p>soech WS 2005/06 E2S 7-2</p>

<p style="text-align: right;">7 Zugriffskontrolle 7.1 Konkurrenz und Koordination</p> <h2>Nebenläufige Aktivitäten</h2> <p>Nichtsequentielles Programm</p>	<p style="text-align: right;">7 Zugriffskontrolle 7.1 Konkurrenz und Koordination</p> <h2>Nebenläufige Aktivitäten</h2> <p>Nichtsequentielles Programm</p>
<p><b>Nebenläufigkeit</b> (engl. <i>concurrency</i>) bezeichnet das Verhältnis von nicht kausal abhängigen Ereignissen, die sich also nicht beeinflussen</p> <ul style="list-style-type: none"> <li>▶ Aktionen können nebenläufig ausgeführt werden, wenn keine das Resultat des anderen benötigt</li> </ul>	<p><b>Nebenläufigkeit</b> (engl. <i>concurrency</i>) bezeichnet das Verhältnis von nicht kausal abhängigen Ereignissen, die sich also nicht beeinflussen</p> <ul style="list-style-type: none"> <li>▶ Aktionen können nebenläufig ausgeführt werden, wenn keine das Resultat des anderen benötigt</li> </ul> <div style="border: 1px solid gray; padding: 5px; margin: 10px 0;"> <pre>1:  foo = 4711; 2:  bar = 42; 3:  foobar = foo + bar; 4:  barfoo = bar + foo; 5:  hal = foobar + barfoo;</pre> </div> <ul style="list-style-type: none"> <li>▶ Zeile 1 kann nebenläufig zu Zeile 2 ausgeführt werden</li> <li>▶ Zeile 3 kann nebenläufig zu Zeile 4 ausgeführt werden</li> </ul>
<p>soech WS 2005/06 E2S 7-2</p>	<p>soech WS 2005/06 E2S 7-2</p>

## Nebenläufige Aktivitäten

Nichtsequentielles Programm

**Nebenläufigkeit** (engl. *concurrency*) bezeichnet das Verhältnis von nicht kausal abhängigen Ereignissen, die sich also nicht beeinflussen

- ▶ Aktionen können nebenläufig ausgeführt werden, wenn keine das Resultat des anderen benötigt

```
1: foo = 4711;
2: bar = 42;
3: foobar = foo + bar;
4: barfoo = bar + foo;
5: hal = foobar + barfoo;
```

- ▶ Zeile 1 kann nebenläufig zu Zeile 2 ausgeführt werden
- ▶ Zeile 3 kann nebenläufig zu Zeile 4 ausgeführt werden

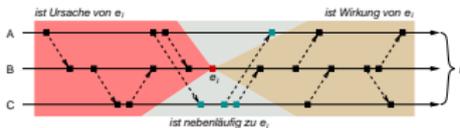
**Kausalität** (lat. *causa*: Ursache) ist die Beziehung zwischen **Ursache** und **Wirkung**, d.h., die ursächliche Verbindung zweier Ereignisse

- ▶ Ereignisse sind nebenläufig, wenn keines Ursache des anderen ist

## Kausalordnung

Nebenläufigkeit als relativistischer Begriff von Gleichzeitigkeit

**Relationen** „ist Ursache von“, „ist Wirkung von“ und „ist nebenläufig zu“:



- ▶ ein Ereignis **ist nebenläufig zu** einem anderen, wenn es im **Anderswo** des anderen Ereignisses liegt
  - ▶ d.h., weder in der Zukunft noch in der Vergangenheit des anderen
- ▶ das Ereignis ist weder Ursache oder Wirkung des anderen Ereignisses

## Kausalordnung (Forts.)

Rangfolge aus Gründen von Daten- und Zeitabhängigkeit

„ist Ursache von“  
„ist Wirkung von“  
„ist nebenläufig zu“ }  $\leadsto$  **Sequentialisierung** von Ereignissen/Aktionen

## Kausalordnung (Forts.)

Rangfolge aus Gründen von Daten- und Zeitabhängigkeit

„ist Ursache von“  
„ist Wirkung von“  
„ist nebenläufig zu“ }  $\leadsto$  **Sequentialisierung** von Ereignissen/Aktionen

Aktionen können im **Echtzeitbetrieb** nebenläufig stattfinden, wenn ...

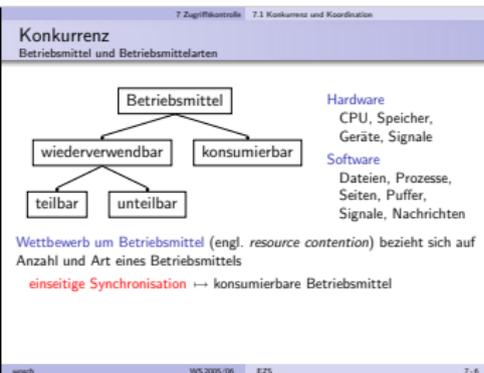
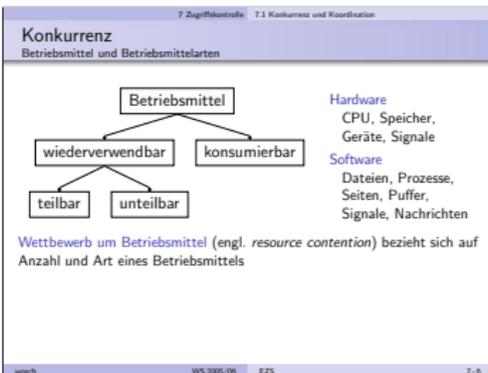
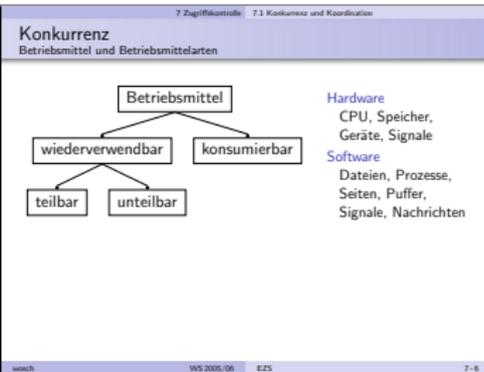
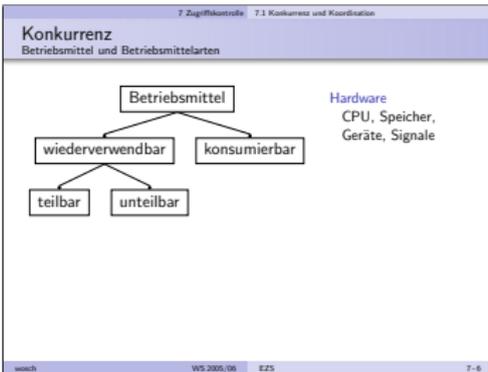
- ▶ keine das Resultat der anderen benötigt (S. 7-2) ✓
- ▶ keine die (strikten) Zeitbedingungen der anderen verletzt
  - ▶ Zeitpunkte dürfen nicht bzw. nur selten verpasst werden
  - ▶ Zeitintervalle dürfen nicht bzw. nur begrenzt zeitlich gedehnt werden
    - ▶ Abstand zwischen Ursache (Ereigniszeitpunkt) und Wirkung (Termin)

7 Zugriffskontrolle   7.1 Konkurrenz und Koordination	7 Zugriffskontrolle   7.1 Konkurrenz und Koordination
<h3>Kausalordnung <small>(Fortz.)</small></h3> <p>Rangfolge aus Gründen von Daten- und Zeitabhängigkeit</p> <p>„ist Ursache von“ „ist Wirkung von“ „ist nebenläufig zu“ } <math>\leadsto</math> Sequentialisierung von Ereignissen/Aktionen</p> <p>Aktionen können im <b>Echtzeitbetrieb</b> nebenläufig stattfinden, wenn ...</p> <ul style="list-style-type: none"> <li>▶ keine das Resultat der anderen benötigt (S. 7-2) ✓</li> <li>▶ keine die (strikten) Zeitbedingungen der anderen verletzt             <ul style="list-style-type: none"> <li>▶ Zeitpunkte dürfen nicht bzw. nur selten verpasst werden</li> <li>▶ Zeitintervalle dürfen nicht bzw. nur begrenzt zeitlich gedehnt werden                 <ul style="list-style-type: none"> <li>▶ Abstand zwischen Ursache (Ereigniszeitpunkt) und Wirkung (Termin)</li> </ul> </li> </ul> </li> </ul> <p>☞ die Relationen unterliegen (weichen bis harten) Echtzeitbedingungen</p>	<h3>Koordinierung</h3> <p>Reihenschaltung nebenläufiger Aktivitäten</p> <p><b>Zugriffskontrolle koordiniert nebenläufige Zugriffe</b> auf gemeinsame aber unteilbare Betriebsmittel <math>\leadsto</math> Synchronisation</p> <ul style="list-style-type: none"> <li>▶ typischerweise blockierend, wenn das Betriebsmittel nicht die CPU ist</li> <li>▶ möglicherweise nicht-blockierend, sonst ...             <ul style="list-style-type: none"> <li>▶ kritische Abschnitte</li> </ul> </li> </ul>
<p>soech WS 2005/06 E25 7-4</p>	<p>soech WS 2005/06 E25 7-5</p>

7 Zugriffskontrolle   7.1 Konkurrenz und Koordination	7 Zugriffskontrolle   7.1 Konkurrenz und Koordination
<h3>Koordinierung</h3> <p>Reihenschaltung nebenläufiger Aktivitäten</p> <p><b>Zugriffskontrolle koordiniert nebenläufige Zugriffe</b> auf gemeinsame aber unteilbare Betriebsmittel <math>\leadsto</math> Synchronisation</p> <ul style="list-style-type: none"> <li>▶ typischerweise blockierend, wenn das Betriebsmittel nicht die CPU ist</li> <li>▶ möglicherweise nicht-blockierend, sonst ...             <ul style="list-style-type: none"> <li>▶ kritische Abschnitte</li> </ul> </li> </ul> <p><b>Synchronisation</b> (gr. <i>syn</i>: zusammen, <i>chrónos</i>: Zeit) bezeichnet das „Herstellen von Gleichzeitigkeit“</p>	<h3>Koordinierung</h3> <p>Reihenschaltung nebenläufiger Aktivitäten</p> <p><b>Zugriffskontrolle koordiniert nebenläufige Zugriffe</b> auf gemeinsame aber unteilbare Betriebsmittel <math>\leadsto</math> Synchronisation</p> <ul style="list-style-type: none"> <li>▶ typischerweise blockierend, wenn das Betriebsmittel nicht die CPU ist</li> <li>▶ möglicherweise nicht-blockierend, sonst ...             <ul style="list-style-type: none"> <li>▶ kritische Abschnitte</li> </ul> </li> </ul> <p><b>Synchronisation</b> (gr. <i>syn</i>: zusammen, <i>chrónos</i>: Zeit) bezeichnet das „Herstellen von Gleichzeitigkeit“</p> <ul style="list-style-type: none"> <li>▶ Koordination der Kooperation und Konkurrenz zwischen Prozessen</li> </ul>
<p>soech WS 2005/06 E25 7-5</p>	<p>soech WS 2005/06 E25 7-5</p>

7 Zugriffskontrolle 7.1 Konkurrenz und Koordination	7 Zugriffskontrolle 7.1 Konkurrenz und Koordination
<b>Koordinierung</b> Reihenschaltung nebenläufiger Aktivitäten	<b>Koordinierung</b> Reihenschaltung nebenläufiger Aktivitäten
<p><b>Zugriffskontrolle koordiniert nebenläufige Zugriffe</b> auf gemeinsame aber unteilbare Betriebsmittel <math>\leadsto</math> Synchronisation</p> <ul style="list-style-type: none"> <li>▶ typischerweise blockierend, wenn das Betriebsmittel nicht die CPU ist</li> <li>▶ möglicherweise nicht-blockierend, sonst...             <ul style="list-style-type: none"> <li>▶ kritische Abschnitte</li> </ul> </li> </ul> <p><b>Synchronisation</b> (gr. <i>syn</i>: zusammen, <i>chrónos</i>: Zeit) bezeichnet das „Herstellen von Gleichzeitigkeit“</p> <ul style="list-style-type: none"> <li>▶ Koordination der Kooperation und Konkurrenz zwischen Prozessen</li> <li>▶ Abgleich von Echtzeituhren (oder Daten) in verteilten Systemen</li> </ul>	<p><b>Zugriffskontrolle koordiniert nebenläufige Zugriffe</b> auf gemeinsame aber unteilbare Betriebsmittel <math>\leadsto</math> Synchronisation</p> <ul style="list-style-type: none"> <li>▶ typischerweise blockierend, wenn das Betriebsmittel nicht die CPU ist</li> <li>▶ möglicherweise nicht-blockierend, sonst...             <ul style="list-style-type: none"> <li>▶ kritische Abschnitte</li> </ul> </li> </ul> <p><b>Synchronisation</b> (gr. <i>syn</i>: zusammen, <i>chrónos</i>: Zeit) bezeichnet das „Herstellen von Gleichzeitigkeit“</p> <ul style="list-style-type: none"> <li>▶ Koordination der Kooperation und Konkurrenz zwischen Prozessen</li> <li>▶ Abgleich von Echtzeituhren (oder Daten) in verteilten Systemen</li> <li>▶ Sequentialisierung von Ereignissen entlang einer Kausalordnung             <ul style="list-style-type: none"> <li>▶ z.B. logische Uhren</li> </ul> </li> </ul>
soech WS 2005/06 E25 7-5	soech WS 2005/06 E25 7-5

7 Zugriffskontrolle 7.1 Konkurrenz und Koordination	7 Zugriffskontrolle 7.1 Konkurrenz und Koordination
<b>Koordinierung</b> Reihenschaltung nebenläufiger Aktivitäten	<b>Konkurrenz</b> Betriebsmittel und Betriebsmittelarten
<p><b>Zugriffskontrolle koordiniert nebenläufige Zugriffe</b> auf gemeinsame aber unteilbare Betriebsmittel <math>\leadsto</math> Synchronisation</p> <ul style="list-style-type: none"> <li>▶ typischerweise blockierend, wenn das Betriebsmittel nicht die CPU ist</li> <li>▶ möglicherweise nicht-blockierend, sonst...             <ul style="list-style-type: none"> <li>▶ kritische Abschnitte</li> </ul> </li> </ul> <p><b>Synchronisation</b> (gr. <i>syn</i>: zusammen, <i>chrónos</i>: Zeit) bezeichnet das „Herstellen von Gleichzeitigkeit“</p> <ul style="list-style-type: none"> <li>▶ Koordination der Kooperation und Konkurrenz zwischen Prozessen</li> <li>▶ Abgleich von Echtzeituhren (oder Daten) in verteilten Systemen</li> <li>▶ Sequentialisierung von Ereignissen entlang einer Kausalordnung             <ul style="list-style-type: none"> <li>▶ z.B. logische Uhren</li> </ul> </li> </ul> <p>☞ analytisch oder konstruktiv: Einplanung vs. Implementierung (S. 4-20)</p>	<pre> graph TD     A[Betriebsmittel] --&gt; B[wiederverwendbar]     A --&gt; C[konsumierbar]     B --&gt; D[teilbar]     B --&gt; E[unteilbar]       </pre>
soech WS 2005/06 E25 7-5	soech WS 2005/06 E25 7-6



7 Zugriffskontrolle 7.1 Konkurrenz und Koordination Konkurrenz Betriebsmittel und Betriebsmittelarten	7 Zugriffskontrolle 7.1 Konkurrenz und Koordination Konkurrenz (Forts.) Serialisierung von Arbeitsabläufen mit begrenzten/unteilbaren Betriebsmitteln
<div data-bbox="216 170 658 310"> <pre> graph TD     BM[Betriebsmittel] --&gt; W[wiederverwendbar]     BM --&gt; K[konsumierbar]     W --&gt; T[teilbar]     W --&gt; U[unteilbar] </pre> <p><b>Hardware</b> CPU, Speicher, Geräte, Signale</p> <p><b>Software</b> Dateien, Prozesse, Seiten, Puffer, Signale, Nachrichten</p> </div> <p><b>Wettbewerb um Betriebsmittel</b> (engl. <i>resource contention</i>) bezieht sich auf Anzahl und Art eines Betriebsmittels</p> <p><b>einseitige Synchronisation</b> <math>\mapsto</math> konsumierbare Betriebsmittel</p> <p><b>mehrsseitige Synchronisation</b> <math>\mapsto</math> wiederverwendbare Betriebsmittel</p> <ul style="list-style-type: none"> <li>▶ Begrenzung, gegenseitiger Ausschluss</li> </ul>	<p>Betriebsmittel, die unteilbar sind, können von gleichzeitigen Prozessen bzw. Arbeitsaufträgen nur nacheinander belegt werden</p>
<p>WS 2005/06 E25 7-6</p>	<p>WS 2005/06 E25 7-7</p>

7 Zugriffskontrolle 7.1 Konkurrenz und Koordination Konkurrenz (Forts.) Serialisierung von Arbeitsabläufen mit begrenzten/unteilbaren Betriebsmitteln	7 Zugriffskontrolle 7.1 Konkurrenz und Koordination Konkurrenz (Forts.) Serialisierung von Arbeitsabläufen mit begrenzten/unteilbaren Betriebsmitteln
<p>Betriebsmittel, die unteilbar sind, können von gleichzeitigen Prozessen bzw. Arbeitsaufträgen nur nacheinander belegt werden</p> <p><b>Vergabe</b> <math>\mapsto</math> das Betriebsmittel sperren und dem Job zuteilen ..... P</p> <ul style="list-style-type: none"> <li>▶ beim Versuch, ein gesperrtes Betriebsmittel erneut zu belegen, wird der anfordernde Job blockiert</li> <li>▶ der blockierende Job erwartet das Ereignis/Signal zur Freigabe des gesperrten Betriebsmittels, ihm wird die CPU entzogen</li> </ul>	<p>Betriebsmittel, die unteilbar sind, können von gleichzeitigen Prozessen bzw. Arbeitsaufträgen nur nacheinander belegt werden</p> <p><b>Vergabe</b> <math>\mapsto</math> das Betriebsmittel sperren und dem Job zuteilen ..... P</p> <ul style="list-style-type: none"> <li>▶ beim Versuch, ein gesperrtes Betriebsmittel erneut zu belegen, wird der anfordernde Job blockiert</li> <li>▶ der blockierende Job erwartet das Ereignis/Signal zur Freigabe des gesperrten Betriebsmittels, ihm wird die CPU entzogen</li> </ul> <p><b>Freigabe</b> <math>\mapsto</math> das Betriebsmittel dem Job wieder entziehen ..... V</p> <ul style="list-style-type: none"> <li>▶ sollten Jobs die Freigabe dieses Betriebsmittels erwarten, wird es sofort der <b>Wiedervergabe</b> zugeführt; das bedeutet:             <ol style="list-style-type: none"> <li>(a) das Betriebsmittel entsperren und alle Jobs deblockieren, die sich dann wiederholt um die Vergabe zu bemühen haben <i>oder</i></li> <li>(b) einen Job auswählen und ihm das Betriebsmittel zuteilen</li> </ol> </li> <li>▶ nur der Job, dem das Betriebsmittel zugeteilt war, kann es freigeben</li> </ul>
<p>WS 2005/06 E25 7-7</p>	<p>WS 2005/06 E25 7-7</p>

7 Zugriffskontrolle 7.1 Konkurrenz und Koordination

### Konfliktsituation

Blockierung von Arbeitsaufträgen

Arbeitsaufträge befinden sich untereinander im **Konflikt**, wenn...

- ▶ nur eine begrenzte Anzahl gemeinsamer Betriebsmitteln vorrätig ist
- ▶ sie unteilbare Betriebsmittel desselben Typs gemeinsam verwenden

weach WS 2005/06 E2S 7-8

7 Zugriffskontrolle 7.1 Konkurrenz und Koordination

### Konfliktsituation

Blockierung von Arbeitsaufträgen

Arbeitsaufträge befinden sich untereinander im **Konflikt**, wenn...

- ▶ nur eine begrenzte Anzahl gemeinsamer Betriebsmitteln vorrätig ist
- ▶ sie unteilbare Betriebsmittel desselben Typs gemeinsam verwenden

Arbeitsaufträge sind im **Streit** (engl. *contention*) um ein Betriebsmittel, wenn einer das Betriebsmittel anfordert, das ein anderer bereits besitzt

- ▶ der anfordernde Job blockiert und wartet auf die Freigabe des Betriebsmittels durch den Job, der das Betriebsmittel belegt
- ▶ der das Betriebsmittel belegende Job löst den auf die Freigabe des Betriebsmittels wartenden Job aus, d.h., deblockiert in wieder

weach WS 2005/06 E2S 7-8

7 Zugriffskontrolle 7.1 Konkurrenz und Koordination

### Konfliktsituation

Blockierung von Arbeitsaufträgen

Arbeitsaufträge befinden sich untereinander im **Konflikt**, wenn...

- ▶ nur eine begrenzte Anzahl gemeinsamer Betriebsmitteln vorrätig ist
- ▶ sie unteilbare Betriebsmittel desselben Typs gemeinsam verwenden

Arbeitsaufträge sind im **Streit** (engl. *contention*) um ein Betriebsmittel, wenn einer das Betriebsmittel anfordert, das ein anderer bereits besitzt

- ▶ der anfordernde Job blockiert und wartet auf die Freigabe des Betriebsmittels durch den Job, der das Betriebsmittel belegt
- ▶ der das Betriebsmittel belegende Job löst den auf die Freigabe des Betriebsmittels wartenden Job aus, d.h., deblockiert in wieder

☞ Nutzung begrenzter/unteilbarer Betriebsmittel impliziert **Kooperation**

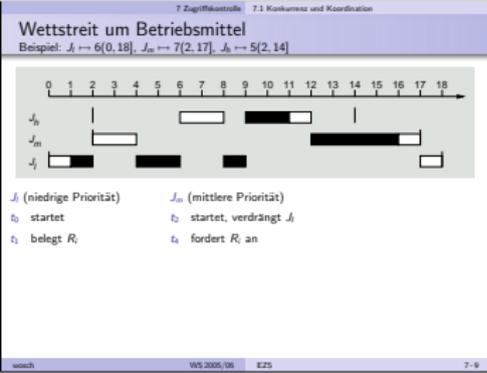
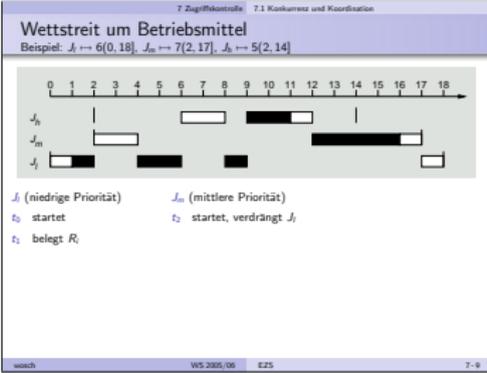
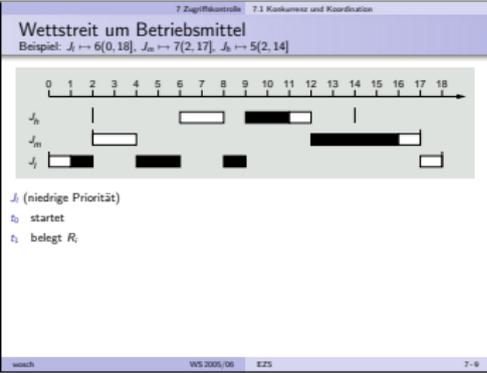
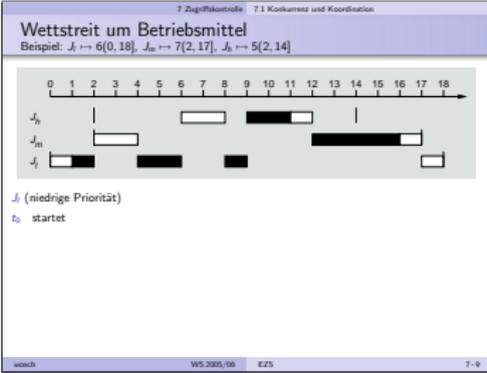
weach WS 2005/06 E2S 7-8

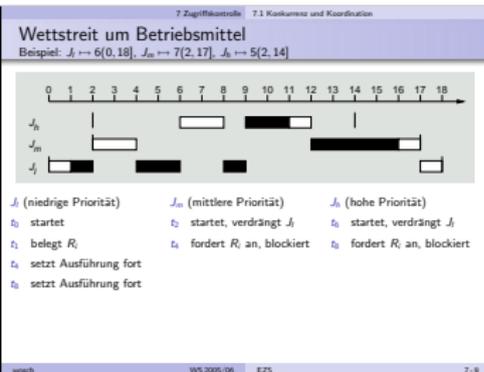
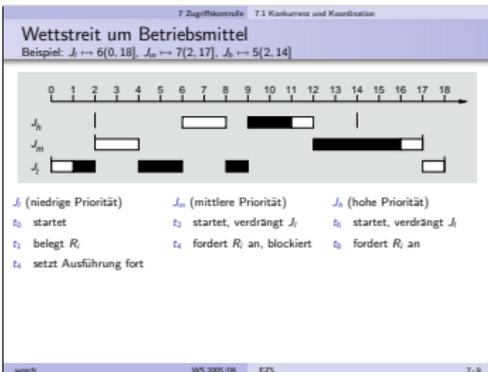
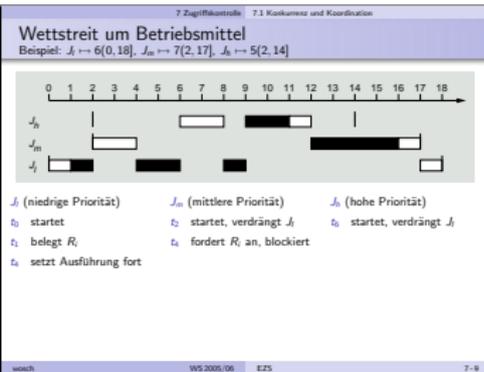
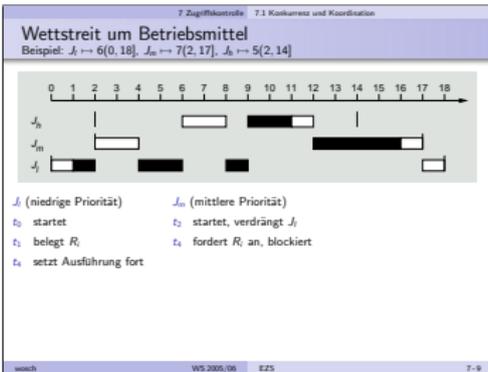
7 Zugriffskontrolle 7.1 Konkurrenz und Koordination

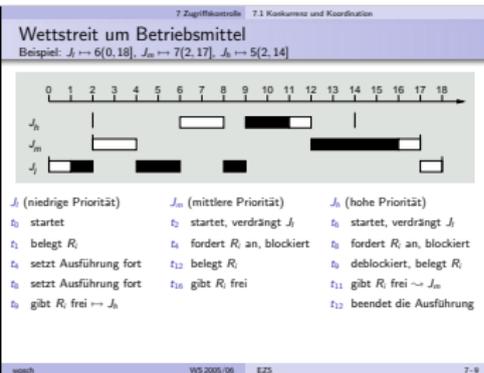
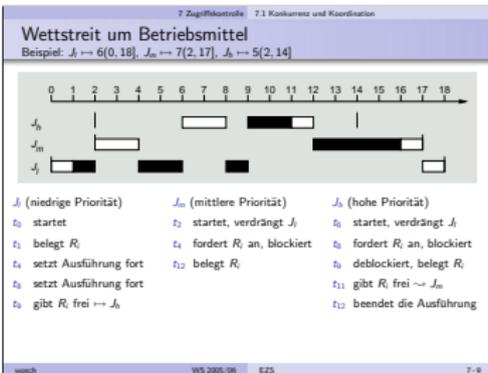
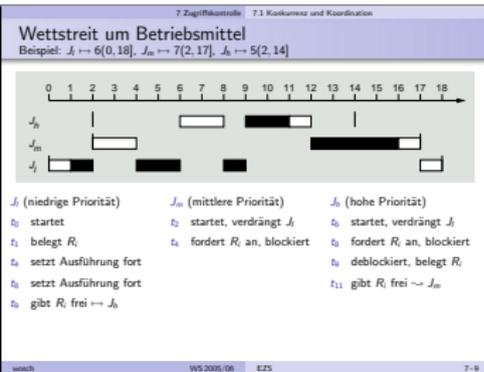
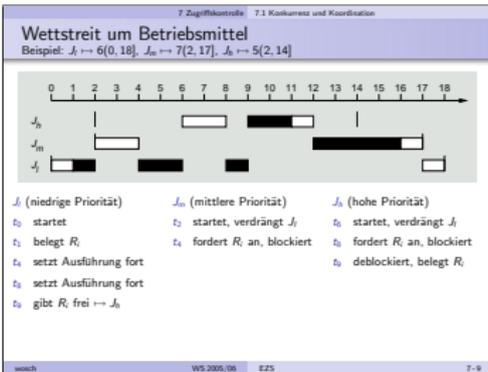
### Wettstreit um Betriebsmittel

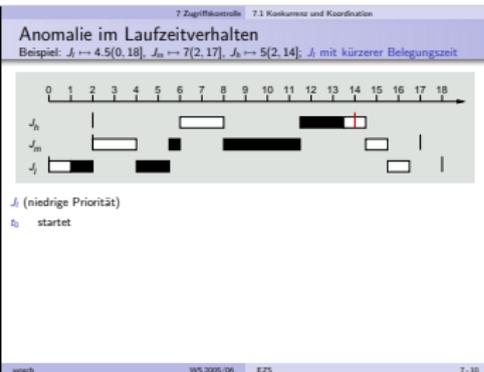
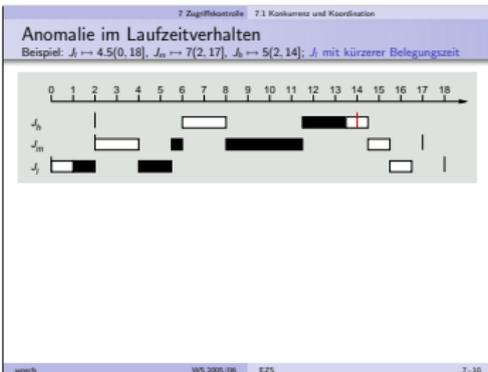
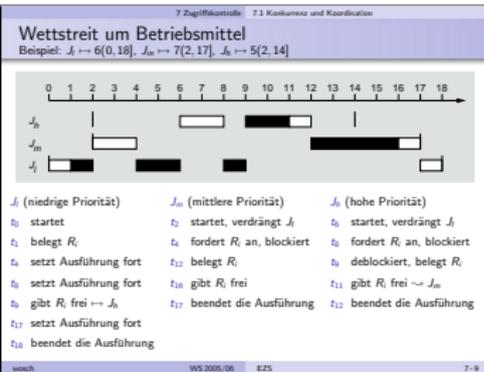
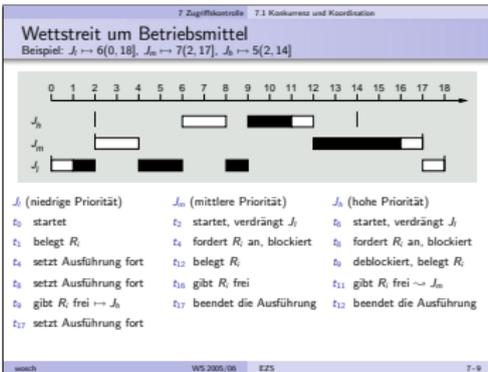
Beispiel:  $J_1 \mapsto 6(0,18]$ ,  $J_m \mapsto 7(2,17]$ ,  $J_b \mapsto 5(2,14]$

weach WS 2005/06 E2S 7-8

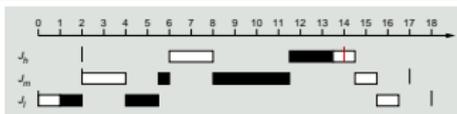








## Anomalie im Laufzeitverhalten

Beispiel:  $J_1 \mapsto 4.5(0,18)$ ,  $J_m \mapsto 7(2,17)$ ,  $J_h \mapsto 5(2,14)$ ;  $J_1$  mit kürzerer Belegungszeit $J_1$  (niedrige Priorität)

- $t_0$  startet
- $t_1$  belegt  $R_1$

## Anomalie im Laufzeitverhalten

Beispiel:  $J_1 \mapsto 4.5(0,18)$ ,  $J_m \mapsto 7(2,17)$ ,  $J_h \mapsto 5(2,14)$ ;  $J_1$  mit kürzerer Belegungszeit $J_1$  (niedrige Priorität)

- $t_0$  startet
- $t_1$  belegt  $R_1$

 $J_m$  (mittlere Priorität)

- $t_2$  startet, verdrängt  $J_1$

## Anomalie im Laufzeitverhalten

Beispiel:  $J_1 \mapsto 4.5(0,18)$ ,  $J_m \mapsto 7(2,17)$ ,  $J_h \mapsto 5(2,14)$ ;  $J_1$  mit kürzerer Belegungszeit $J_1$  (niedrige Priorität)

- $t_0$  startet
- $t_1$  belegt  $R_1$

 $J_m$  (mittlere Priorität)

- $t_2$  startet, verdrängt  $J_1$
- $t_3$  fordert  $R_1$  an

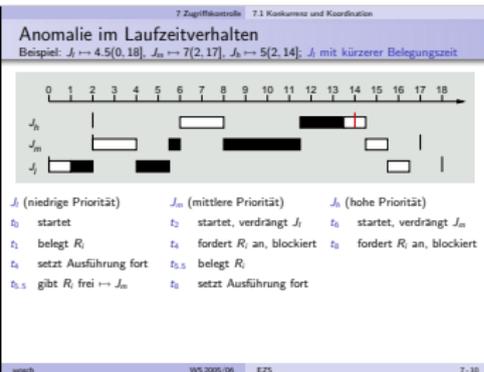
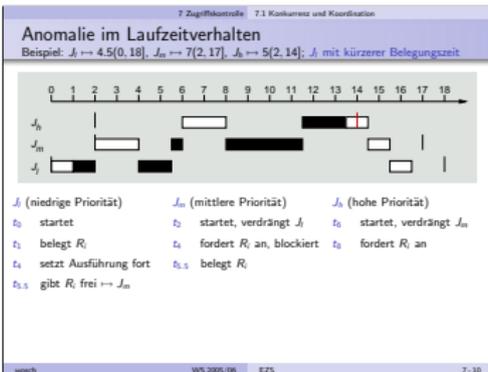
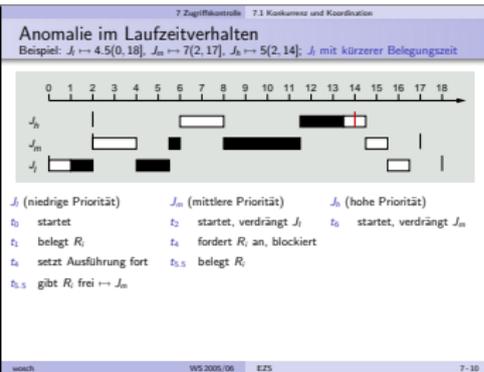
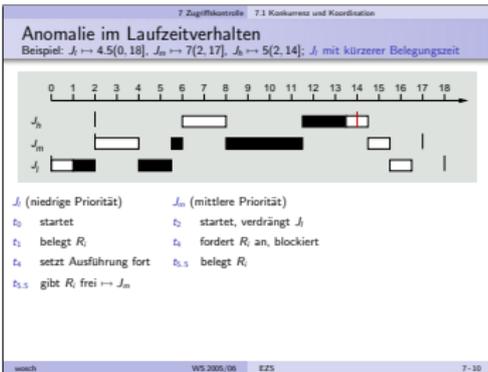
## Anomalie im Laufzeitverhalten

Beispiel:  $J_1 \mapsto 4.5(0,18)$ ,  $J_m \mapsto 7(2,17)$ ,  $J_h \mapsto 5(2,14)$ ;  $J_1$  mit kürzerer Belegungszeit $J_1$  (niedrige Priorität)

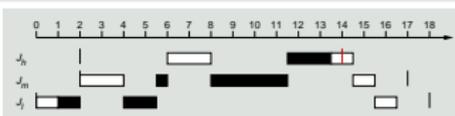
- $t_0$  startet
- $t_1$  belegt  $R_1$
- $t_4$  setzt Ausführung fort

 $J_m$  (mittlere Priorität)

- $t_2$  startet, verdrängt  $J_1$
- $t_3$  fordert  $R_1$  an, blockiert



## Anomalie im Laufzeitverhalten

Beispiel:  $J_i \mapsto 4.5(0,18)$ ,  $J_m \mapsto 7(2,17)$ ,  $J_h \mapsto 5(2,14)$ ;  $J_i$  mit kürzerer Belegungszeit

$J_i$ (niedrige Priorität)	$J_m$ (mittlere Priorität)	$J_h$ (hohe Priorität)
$t_0$ startet	$t_2$ startet, verdrängt $J_i$	$t_8$ startet, verdrängt $J_m$
$t_1$ belegt $R_i$	$t_4$ fordert $R_i$ an, blockiert	$t_9$ fordert $R_i$ an, blockiert
$t_4$ setzt Ausführung fort	$t_5$ belegt $R_i$	$t_{11.5}$ belegt $R_i$
$t_{5.5}$ gibt $R_i$ frei $\mapsto J_m$	$t_6$ setzt Ausführung fort	
	$t_{11.5}$ gibt $R_i$ frei $\mapsto J_h$	

## Anomalie im Laufzeitverhalten

Beispiel:  $J_i \mapsto 4.5(0,18)$ ,  $J_m \mapsto 7(2,17)$ ,  $J_h \mapsto 5(2,14)$ ;  $J_i$  mit kürzerer Belegungszeit

$J_i$ (niedrige Priorität)	$J_m$ (mittlere Priorität)	$J_h$ (hohe Priorität)
$t_0$ startet	$t_2$ startet, verdrängt $J_i$	$t_8$ startet, verdrängt $J_m$
$t_1$ belegt $R_i$	$t_4$ fordert $R_i$ an, blockiert	$t_9$ fordert $R_i$ an, blockiert
$t_4$ setzt Ausführung fort	$t_5$ belegt $R_i$	$t_{11.5}$ belegt $R_i$
$t_{5.5}$ gibt $R_i$ frei $\mapsto J_m$	$t_6$ setzt Ausführung fort	$t_{13.5}$ gibt $R_i$ frei
	$t_{11.5}$ gibt $R_i$ frei $\mapsto J_h$	

## Anomalie im Laufzeitverhalten

Beispiel:  $J_i \mapsto 4.5(0,18)$ ,  $J_m \mapsto 7(2,17)$ ,  $J_h \mapsto 5(2,14)$ ;  $J_i$  mit kürzerer Belegungszeit

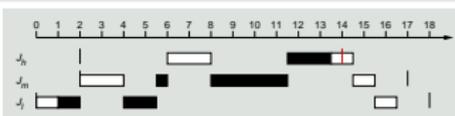
$J_i$ (niedrige Priorität)	$J_m$ (mittlere Priorität)	$J_h$ (hohe Priorität)
$t_0$ startet	$t_2$ startet, verdrängt $J_i$	$t_8$ startet, verdrängt $J_m$
$t_1$ belegt $R_i$	$t_4$ fordert $R_i$ an, blockiert	$t_9$ fordert $R_i$ an, blockiert
$t_4$ setzt Ausführung fort	$t_5$ belegt $R_i$	$t_{11.5}$ belegt $R_i$
$t_{5.5}$ gibt $R_i$ frei $\mapsto J_m$	$t_6$ setzt Ausführung fort	$t_{13.5}$ gibt $R_i$ frei
	$t_{11.5}$ gibt $R_i$ frei $\mapsto J_h$	$t_{14}$ verletzt seinen Termin

## Anomalie im Laufzeitverhalten

Beispiel:  $J_i \mapsto 4.5(0,18)$ ,  $J_m \mapsto 7(2,17)$ ,  $J_h \mapsto 5(2,14)$ ;  $J_i$  mit kürzerer Belegungszeit

$J_i$ (niedrige Priorität)	$J_m$ (mittlere Priorität)	$J_h$ (hohe Priorität)
$t_0$ startet	$t_2$ startet, verdrängt $J_i$	$t_8$ startet, verdrängt $J_m$
$t_1$ belegt $R_i$	$t_4$ fordert $R_i$ an, blockiert	$t_9$ fordert $R_i$ an, blockiert
$t_4$ setzt Ausführung fort	$t_5$ belegt $R_i$	$t_{11.5}$ belegt $R_i$
$t_{5.5}$ gibt $R_i$ frei $\mapsto J_m$	$t_6$ setzt Ausführung fort	$t_{13.5}$ gibt $R_i$ frei
	$t_{11.5}$ gibt $R_i$ frei $\mapsto J_h$	$t_{14}$ beendet die Ausführung
	$t_{14.5}$ setzt Ausführung fort	

## Anomalie im Laufzeitverhalten

Beispiel:  $J_i \mapsto 4.5(0,18)$ ,  $J_m \mapsto 7(2,17)$ ,  $J_h \mapsto 5(2,14)$ ;  $J_i$  mit kürzerer Belegungszeit $J_i$  (niedrige Priorität)

- $t_0$  startet
- $t_1$  belegt  $R_i$
- $t_4$  setzt Ausführung fort
- $t_{4.5}$  gibt  $R_i$  frei  $\mapsto J_m$
- $t_{15.5}$  setzt Ausführung fort

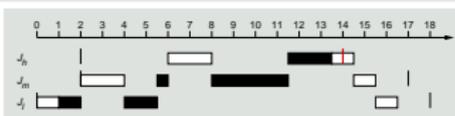
 $J_m$  (mittlere Priorität)

- $t_2$  startet, verdrängt  $J_i$
- $t_4$  fordert  $R_i$  an, blockiert
- $t_{5.5}$  belegt  $R_i$
- $t_7$  setzt Ausführung fort
- $t_{11.5}$  gibt  $R_i$  frei  $\mapsto J_h$
- $t_{14.5}$  setzt Ausführung fort
- $t_{15.5}$  beendet die Ausführung

 $J_h$  (hohe Priorität)

- $t_2$  startet, verdrängt  $J_m$
- $t_4$  fordert  $R_i$  an, blockiert
- $t_{11.5}$  belegt  $R_i$
- $t_{13.5}$  gibt  $R_i$  frei
- $t_{14}$  verletzt seinen Termin
- $t_{14.5}$  beendet die Ausführung

## Anomalie im Laufzeitverhalten

Beispiel:  $J_i \mapsto 4.5(0,18)$ ,  $J_m \mapsto 7(2,17)$ ,  $J_h \mapsto 5(2,14)$ ;  $J_i$  mit kürzerer Belegungszeit $J_i$  (niedrige Priorität)

- $t_0$  startet
- $t_1$  belegt  $R_i$
- $t_4$  setzt Ausführung fort
- $t_{4.5}$  gibt  $R_i$  frei  $\mapsto J_m$
- $t_{15.5}$  setzt Ausführung fort
- $t_{16.5}$  beendet die Ausführung

 $J_m$  (mittlere Priorität)

- $t_2$  startet, verdrängt  $J_i$
- $t_4$  fordert  $R_i$  an, blockiert
- $t_{5.5}$  belegt  $R_i$
- $t_7$  setzt Ausführung fort
- $t_{11.5}$  gibt  $R_i$  frei  $\mapsto J_h$
- $t_{14.5}$  setzt Ausführung fort
- $t_{15.5}$  beendet die Ausführung

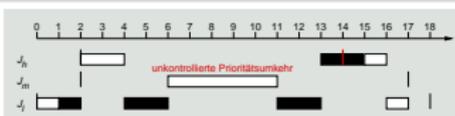
 $J_h$  (hohe Priorität)

- $t_2$  startet, verdrängt  $J_m$
- $t_4$  fordert  $R_i$  an, blockiert
- $t_{11.5}$  belegt  $R_i$
- $t_{13.5}$  gibt  $R_i$  frei
- $t_{14}$  verletzt seinen Termin
- $t_{14.5}$  beendet die Ausführung

## Anomalie im Laufzeitverhalten (Forts.)

Beispiel:  $J_i \mapsto 7(0,18)$ ,  $J_m \mapsto 5(2,17)$ ,  $J_h \mapsto 5(2,14)$ ;  $J_m$  fordert  $R_i$  nicht an

## Anomalie im Laufzeitverhalten (Forts.)

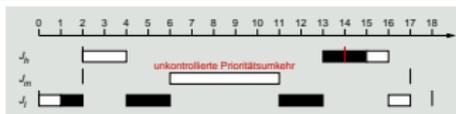
Beispiel:  $J_i \mapsto 7(0,18)$ ,  $J_m \mapsto 5(2,17)$ ,  $J_h \mapsto 5(2,14)$ ;  $J_m$  fordert  $R_i$  nicht an $J_i$  (niedrige Priorität)

- $t_0$  startet

## Anomalie im Laufzeitverhalten (Forts.)

Beispiel:  $J_i \mapsto 7(0, 18)$ ,  $J_m \mapsto 5(2, 17)$ ,  $J_h \mapsto 5(2, 14)$ ;  $J_m$  fordert  $R_i$  nicht an $J_i$  (niedrige Priorität) $t_0$  startet $t_1$  belegt  $R_i$ 

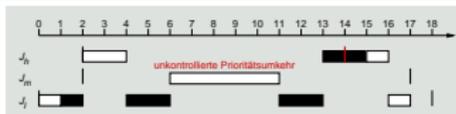
## Anomalie im Laufzeitverhalten (Forts.)

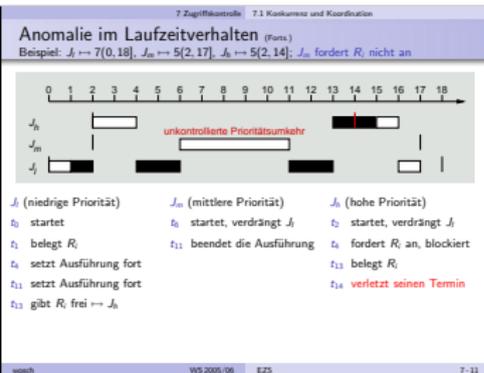
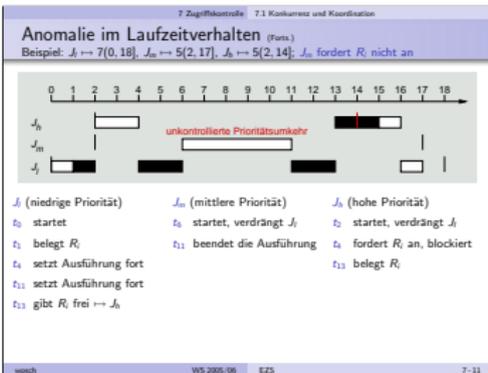
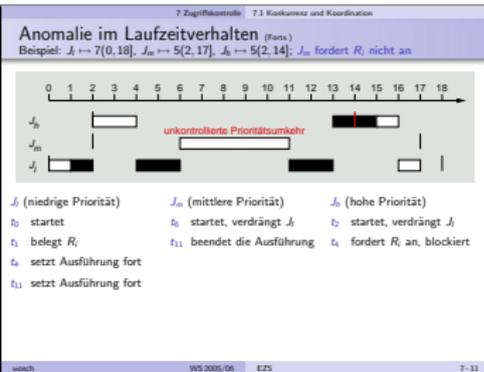
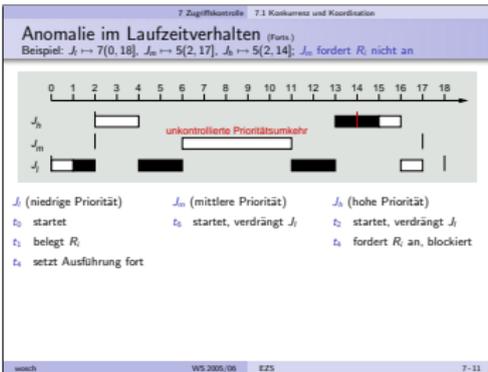
Beispiel:  $J_i \mapsto 7(0, 18)$ ,  $J_m \mapsto 5(2, 17)$ ,  $J_h \mapsto 5(2, 14)$ ;  $J_m$  fordert  $R_i$  nicht an $J_i$  (niedrige Priorität) $t_0$  startet $t_1$  belegt  $R_i$  $J_h$  (hohe Priorität) $t_2$  startet, verdrängt  $J_i$ 

## Anomalie im Laufzeitverhalten (Forts.)

Beispiel:  $J_i \mapsto 7(0, 18)$ ,  $J_m \mapsto 5(2, 17)$ ,  $J_h \mapsto 5(2, 14)$ ;  $J_m$  fordert  $R_i$  nicht an $J_i$  (niedrige Priorität) $t_0$  startet $t_1$  belegt  $R_i$  $J_h$  (hohe Priorität) $t_2$  startet, verdrängt  $J_i$  $t_3$  fordert  $R_i$  an

## Anomalie im Laufzeitverhalten (Forts.)

Beispiel:  $J_i \mapsto 7(0, 18)$ ,  $J_m \mapsto 5(2, 17)$ ,  $J_h \mapsto 5(2, 14)$ ;  $J_m$  fordert  $R_i$  nicht an $J_i$  (niedrige Priorität) $t_0$  startet $t_1$  belegt  $R_i$  $t_3$  setzt Ausführung fort $J_h$  (hohe Priorität) $t_2$  startet, verdrängt  $J_i$  $t_3$  fordert  $R_i$  an, blockiert



## Anomalie im Laufzeitverhalten (Purs.)

Beispiel:  $J_i \mapsto 7(0, 18)$ ,  $J_m \mapsto 5(2, 17)$ ,  $J_h \mapsto 5(2, 14)$ ;  $J_m$  fordert  $R_i$  nicht an $J_i$  (niedrige Priorität)

- $t_0$  startet
- $t_1$  belegt  $R_i$
- $t_2$  setzt Ausführung fort
- $t_{11}$  setzt Ausführung fort
- $t_{13}$  gibt  $R_i$  frei  $\mapsto J_h$

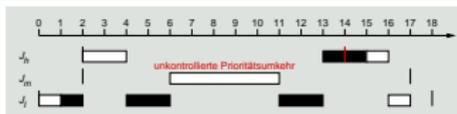
 $J_m$  (mittlere Priorität)

- $t_5$  startet, verdrängt  $J_i$
- $t_{11}$  beendet die Ausführung

 $J_h$  (hohe Priorität)

- $t_2$  startet, verdrängt  $J_i$
- $t_4$  fordert  $R_i$  an, blockiert
- $t_{13}$  belegt  $R_i$
- $t_{14}$  verletzt seinen Termin
- $t_{15}$  gibt  $R_i$  frei

## Anomalie im Laufzeitverhalten (Purs.)

Beispiel:  $J_i \mapsto 7(0, 18)$ ,  $J_m \mapsto 5(2, 17)$ ,  $J_h \mapsto 5(2, 14)$ ;  $J_m$  fordert  $R_i$  nicht an $J_i$  (niedrige Priorität)

- $t_0$  startet
- $t_1$  belegt  $R_i$
- $t_2$  setzt Ausführung fort
- $t_{11}$  setzt Ausführung fort
- $t_{13}$  gibt  $R_i$  frei  $\mapsto J_h$
- $t_{10}$  setzt Ausführung fort

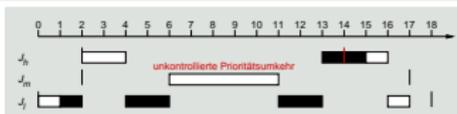
 $J_m$  (mittlere Priorität)

- $t_5$  startet, verdrängt  $J_i$
- $t_{11}$  beendet die Ausführung

 $J_h$  (hohe Priorität)

- $t_2$  startet, verdrängt  $J_i$
- $t_4$  fordert  $R_i$  an, blockiert
- $t_{13}$  belegt  $R_i$
- $t_{14}$  verletzt seinen Termin
- $t_{15}$  gibt  $R_i$  frei
- $t_{16}$  beendet die Ausführung

## Anomalie im Laufzeitverhalten (Purs.)

Beispiel:  $J_i \mapsto 7(0, 18)$ ,  $J_m \mapsto 5(2, 17)$ ,  $J_h \mapsto 5(2, 14)$ ;  $J_m$  fordert  $R_i$  nicht an $J_i$  (niedrige Priorität)

- $t_0$  startet
- $t_1$  belegt  $R_i$
- $t_2$  setzt Ausführung fort
- $t_{11}$  setzt Ausführung fort
- $t_{13}$  gibt  $R_i$  frei  $\mapsto J_h$
- $t_{16}$  setzt Ausführung fort
- $t_{17}$  beendet die Ausführung

 $J_m$  (mittlere Priorität)

- $t_5$  startet, verdrängt  $J_i$
- $t_{11}$  beendet die Ausführung

 $J_h$  (hohe Priorität)

- $t_2$  startet, verdrängt  $J_i$
- $t_4$  fordert  $R_i$  an, blockiert
- $t_{13}$  belegt  $R_i$
- $t_{14}$  verletzt seinen Termin
- $t_{15}$  gibt  $R_i$  frei
- $t_{16}$  beendet die Ausführung

Synchronisation *Considered Harmful*

Prioritätsorientierte Einplanung

Prioritätsumkehr (engl. *priority inversion*, S. 6-26)

<small>7 Zugriffskontrolle 7.1 Konkurrenz und Koordination</small> <b>Synchronisation <i>Considered Harmful</i></b> Prioritätsorientierte Einplanung	<small>7 Zugriffskontrolle 7.1 Konkurrenz und Koordination</small> <b>Synchronisation <i>Considered Harmful</i></b> Prioritätsorientierte Einplanung
<p><b>Prioritätsumkehr</b> (engl. <i>priority inversion</i>, S. 6-26), mögliches Phänomen abhängiger Tasks, vorausgesetzt zwei Bedingungen sind erfüllt:</p> <p><b>notwendig</b> ist, dass eine Task hoher Priorität auf eine Task niedriger Priorität wartet <math>\mapsto</math> <b>gegenseitiger Ausschluss</b></p> <p><b>hinreichend</b> ist, dass die Task niedriger Priorität von einer oder mehrerer Tasks mittlerer Priorität verdrängt wird</p> <ul style="list-style-type: none"> <li>▶ auch als unkontrollierte Prioritätsumkehr bezeichnet</li> </ul>	<p><b>Prioritätsumkehr</b> (engl. <i>priority inversion</i>, S. 6-26), mögliches Phänomen abhängiger Tasks, vorausgesetzt zwei Bedingungen sind erfüllt:</p> <p><b>notwendig</b> ist, dass eine Task hoher Priorität auf eine Task niedriger Priorität wartet <math>\mapsto</math> <b>gegenseitiger Ausschluss</b></p> <p><b>hinreichend</b> ist, dass die Task niedriger Priorität von einer oder mehrerer Tasks mittlerer Priorität verdrängt wird</p> <ul style="list-style-type: none"> <li>▶ auch als unkontrollierte Prioritätsumkehr bezeichnet</li> </ul>
<small>soech WS 2005/06 E2S 7-12</small>	<small>soech WS 2005/06 E2S 7-12</small>

<small>7 Zugriffskontrolle 7.1 Konkurrenz und Koordination</small> <b>Synchronisation <i>Considered Harmful</i></b> Prioritätsorientierte Einplanung	<small>7 Zugriffskontrolle 7.1 Konkurrenz und Koordination</small> <b>Synchronisation <i>Considered Harmful</i></b> Prioritätsorientierte Einplanung
<p><b>Prioritätsumkehr</b> (engl. <i>priority inversion</i>, S. 6-26), mögliches Phänomen abhängiger Tasks, vorausgesetzt zwei Bedingungen sind erfüllt:</p> <p><b>notwendig</b> ist, dass eine Task hoher Priorität auf eine Task niedriger Priorität wartet <math>\mapsto</math> <b>gegenseitiger Ausschluss</b></p> <p><b>hinreichend</b> ist, dass die Task niedriger Priorität von einer oder mehrerer Tasks mittlerer Priorität verdrängt wird</p> <ul style="list-style-type: none"> <li>▶ auch als unkontrollierte Prioritätsumkehr bezeichnet</li> </ul>	<p><b>Prioritätsumkehr</b> (engl. <i>priority inversion</i>, S. 6-26), mögliches Phänomen abhängiger Tasks, vorausgesetzt zwei Bedingungen sind erfüllt:</p> <p><b>notwendig</b> ist, dass eine Task hoher Priorität auf eine Task niedriger Priorität wartet <math>\mapsto</math> <b>gegenseitiger Ausschluss</b></p> <p><b>hinreichend</b> ist, dass die Task niedriger Priorität von einer oder mehrerer Tasks mittlerer Priorität verdrängt wird</p> <ul style="list-style-type: none"> <li>▶ auch als unkontrollierte Prioritätsumkehr bezeichnet</li> </ul> <p>Lösungsansätze sind, sofern blockierende Synchronisation in der jeweils gegebenen Situation nicht vermieden werden kann:</p> <ul style="list-style-type: none"> <li>▶ Verdrängungssteuerung <span style="float: right;">☞ S. 7-13</span></li> <li>▶ Prioritätsvererbung <span style="float: right;">☞ S. 7-18</span></li> <li>▶ Prioritätsobergrenzen <span style="float: right;">☞ S. 7-22</span></li> </ul>
<small>soech WS 2005/06 E2S 7-12</small>	<small>soech WS 2005/06 E2S 7-12</small>

7 Zugriffskontrolle 7.2 Verdrängungsteuerung		7 Zugriffskontrolle 7.2 Verdrängungsteuerung	
<b>Verdrängung zeitweise unterbinden</b> Verdrängungsfreie kritische Abschnitte (engl. <i>non-preemptive critical sections</i> , NPCS)		<b>Verdrängung zeitweise unterbinden</b> Verdrängungsfreie kritische Abschnitte (engl. <i>non-preemptive critical sections</i> , NPCS)	
Arbeitsaufträge werden für die <b>Gesamtzeit der Belegung</b> von (unteilbaren) Betriebsmitteln nicht von anderen Arbeitsaufträgen verdrängt <ul style="list-style-type: none"> <li>▶ die Benutzung der Betriebsmittel kontrolliert ein <b>Monitor</b> [25, 26]               <ul style="list-style-type: none"> <li>▶ <i>kernelized monitor</i> [27]</li> </ul> </li> </ul>		Arbeitsaufträge werden für die <b>Gesamtzeit der Belegung</b> von (unteilbaren) Betriebsmitteln nicht von anderen Arbeitsaufträgen verdrängt <ul style="list-style-type: none"> <li>▶ die Benutzung der Betriebsmittel kontrolliert ein <b>Monitor</b> [25, 26]               <ul style="list-style-type: none"> <li>▶ <i>kernelized monitor</i> [27]</li> </ul> </li> <li><b>Eintrittsprotokoll</b> ↔ Verdrängung abwehren               <ul style="list-style-type: none"> <li>▶ ausgelöste Jobs einplanen, aber nicht einlasten</li> </ul> </li> </ul>	
sooch	WS 2005/06 E25	sooch	WS 2005/06 E25
	7-12		7-13

7 Zugriffskontrolle 7.2 Verdrängungsteuerung		7 Zugriffskontrolle 7.2 Verdrängungsteuerung	
<b>Verdrängung zeitweise unterbinden</b> Verdrängungsfreie kritische Abschnitte (engl. <i>non-preemptive critical sections</i> , NPCS)		<b>Verdrängung zeitweise unterbinden</b> Verdrängungsfreie kritische Abschnitte (engl. <i>non-preemptive critical sections</i> , NPCS)	
Arbeitsaufträge werden für die <b>Gesamtzeit der Belegung</b> von (unteilbaren) Betriebsmitteln nicht von anderen Arbeitsaufträgen verdrängt <ul style="list-style-type: none"> <li>▶ die Benutzung der Betriebsmittel kontrolliert ein <b>Monitor</b> [25, 26]               <ul style="list-style-type: none"> <li>▶ <i>kernelized monitor</i> [27]</li> </ul> </li> <li><b>Eintrittsprotokoll</b> ↔ Verdrängung abwehren               <ul style="list-style-type: none"> <li>▶ ausgelöste Jobs einplanen, aber nicht einlasten</li> </ul> </li> <li><b>Austrittsprotokoll</b> ↔ Verdrängung wieder zulassen               <ul style="list-style-type: none"> <li>▶ höher priorisierte Jobs (nachträglich) einlasten</li> </ul> </li> </ul>		Arbeitsaufträge werden für die <b>Gesamtzeit der Belegung</b> von (unteilbaren) Betriebsmitteln nicht von anderen Arbeitsaufträgen verdrängt <ul style="list-style-type: none"> <li>▶ die Benutzung der Betriebsmittel kontrolliert ein <b>Monitor</b> [25, 26]               <ul style="list-style-type: none"> <li>▶ <i>kernelized monitor</i> [27]</li> </ul> </li> <li><b>Eintrittsprotokoll</b> ↔ Verdrängung abwehren               <ul style="list-style-type: none"> <li>▶ ausgelöste Jobs einplanen, aber nicht einlasten</li> </ul> </li> <li><b>Austrittsprotokoll</b> ↔ Verdrängung wieder zulassen               <ul style="list-style-type: none"> <li>▶ höher priorisierte Jobs (nachträglich) einlasten</li> </ul> </li> <li>▶ sobald ein Job ein Betriebsmittel belegt, läuft er unverdrängbar weiter               <ul style="list-style-type: none"> <li>▶ <i>verklemmungsfreies Verfahren</i> durch <i>Verklemmungsvorbeugung</i> <ul style="list-style-type: none"> <li>▶ engl. <i>deadlock prevention</i></li> </ul> </li> </ul> </li> </ul>	
sooch	WS 2005/06 E25	sooch	WS 2005/06 E25
	7-11		7-13

7 Zugriffskontrolle 7.2 Verdrängungssteuerung

### Verdrängung zeitweise unterbinden

Verdrängungsfreie kritische Abschnitte (engl. *non-preemptive critical sections*, NPCS)

Arbeitsaufträge werden für die **Gesamtzeit der Belegung** von (unteilbaren) Betriebsmitteln nicht von anderen Arbeitsaufträgen verdrängt

- ▶ die Benutzung der Betriebsmittel kontrolliert ein **Monitor** [25, 26]
  - ▶ **kernelized monitor** [27]
- Eintrittsprotokoll  $\mapsto$  Verdrängung abwehren
  - ▶ ausgeloste Jobs einplanen, aber nicht einlasten
- Austrittsprotokoll  $\mapsto$  Verdrängung wieder zulassen
  - ▶ höher priorisierte Jobs (nachträglich) einlasten
- ▶ sobald ein Job ein Betriebsmittel belegt, läuft er unverdrängbar weiter
  - ▶ **verklemmungsfreies Verfahren** durch **Verklemmungsvorbeugung**
    - ▶ engl. *deadlock prevention*

☞ Taktsteuerung unterstützt die unverdrängbare Ausführung rahmenweise

woach WS 2005/06 E2S 7-12

7 Zugriffskontrolle 7.2 Verdrängungssteuerung

### Kernelized Monitor

Beispiel (S. 7-11):  $J_i \mapsto 7(0, 18)$ ,  $J_m \mapsto 5(2, 17)$ ,  $J_h \mapsto 5(2, 14)$

woach WS 2005/06 E2S 7-14

7 Zugriffskontrolle 7.2 Verdrängungssteuerung

### Kernelized Monitor

Beispiel (S. 7-11):  $J_i \mapsto 7(0, 18)$ ,  $J_m \mapsto 5(2, 17)$ ,  $J_h \mapsto 5(2, 14)$

$J_i$  (niedrige Priorität)

- $t_0$  startet

woach WS 2005/06 E2S 7-14

7 Zugriffskontrolle 7.2 Verdrängungssteuerung

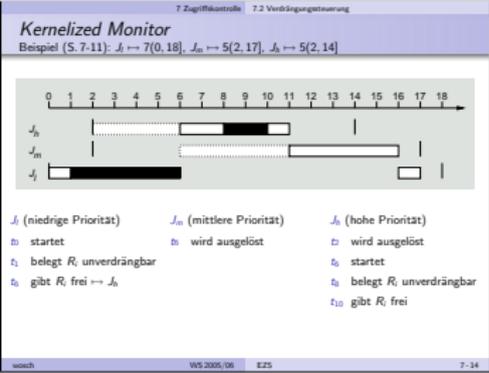
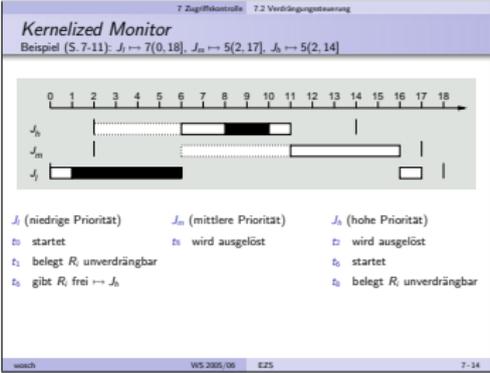
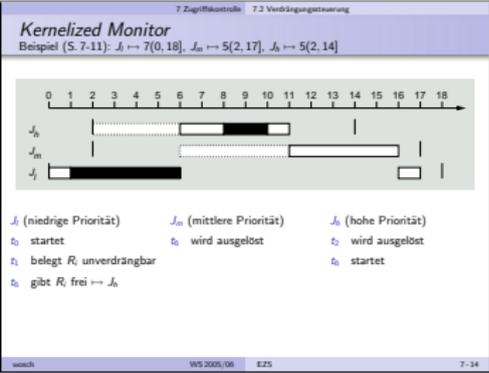
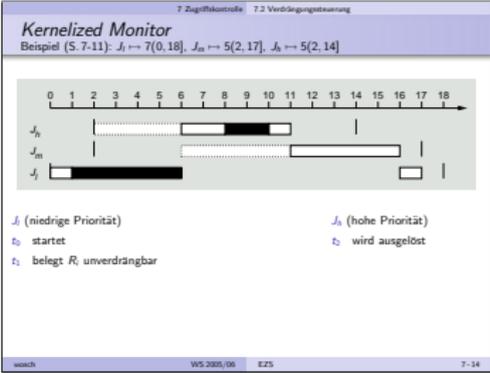
### Kernelized Monitor

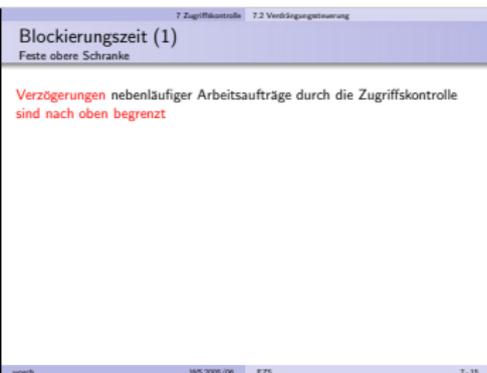
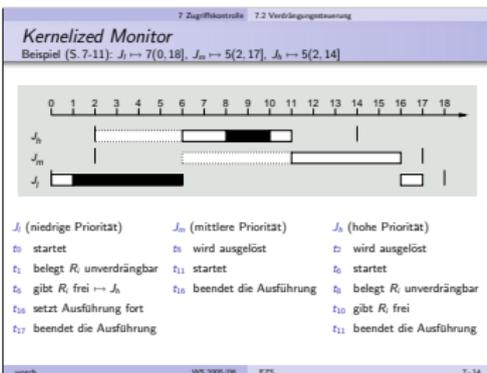
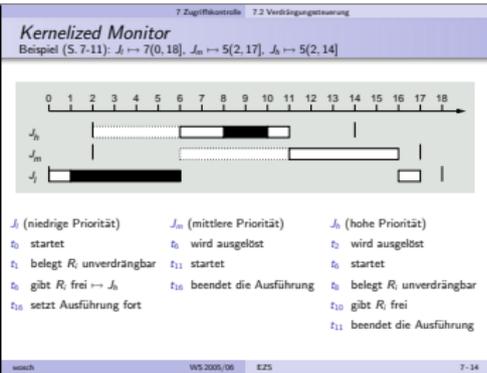
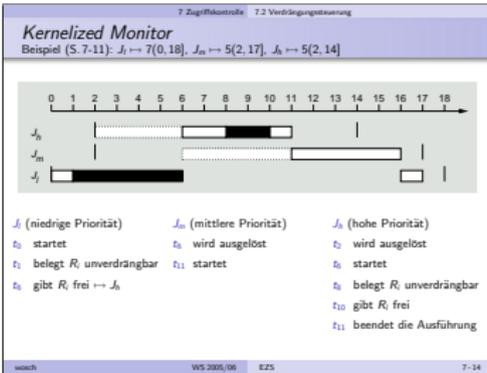
Beispiel (S. 7-11):  $J_i \mapsto 7(0, 18)$ ,  $J_m \mapsto 5(2, 17)$ ,  $J_h \mapsto 5(2, 14)$

$J_i$  (niedrige Priorität)

- $t_0$  startet
- $t_1$  belegt  $R_i$  unverdrängbar

woach WS 2005/06 E2S 7-14





<p style="text-align: center;">7 Zugriffskontrolle 7.2 Verdrängungssteuerung</p> <h3 style="text-align: center;">Blockierungszeit (1)</h3> <p style="text-align: center;">Feste obere Schranke</p> <p><b>Verzögerungen</b> nebenläufiger Arbeitsaufträge durch die Zugriffskontrolle <b>sind nach oben begrenzt</b></p> <ul style="list-style-type: none"> <li>▶ die feste obere Schranke <math>bt</math> bestimmt sich aus der größten WCET aller kritischen Abschnitte aller niedriger priorisierten Jobs: <math>\max(cs)</math></li> </ul>	<p style="text-align: center;">7 Zugriffskontrolle 7.2 Verdrängungssteuerung</p> <h3 style="text-align: center;">Blockierungszeit (1)</h3> <p style="text-align: center;">Feste obere Schranke</p> <p><b>Verzögerungen</b> nebenläufiger Arbeitsaufträge durch die Zugriffskontrolle <b>sind nach oben begrenzt</b></p> <ul style="list-style-type: none"> <li>▶ die feste obere Schranke <math>bt</math> bestimmt sich aus der größten WCET aller kritischen Abschnitte aller niedriger priorisierten Jobs: <math>\max(cs)</math></li> <li>▶ höher priorisierte Jobs werden schlimmstenfalls einmal durch einen niedriger priorisierten Job blockiert</li> </ul>
<p>soech WS 2005/06 E25 7-15</p>	<p>soech WS 2005/06 E25 7-15</p>

<p style="text-align: center;">7 Zugriffskontrolle 7.2 Verdrängungssteuerung</p> <h3 style="text-align: center;">Blockierungszeit (1)</h3> <p style="text-align: center;">Feste obere Schranke</p> <p><b>Verzögerungen</b> nebenläufiger Arbeitsaufträge durch die Zugriffskontrolle <b>sind nach oben begrenzt</b></p> <ul style="list-style-type: none"> <li>▶ die feste obere Schranke <math>bt</math> bestimmt sich aus der größten WCET aller kritischen Abschnitte aller niedriger priorisierten Jobs: <math>\max(cs_k)</math></li> <li>▶ höher priorisierte Jobs werden schlimmstenfalls einmal durch einen niedriger priorisierten Job blockiert</li> </ul> <p>NPCS verzögert eine periodische Task <math>T_i</math> von <math>n</math> periodischen Tasks im taktweisen Betrieb um <math>bt_i = \max(cs_k)</math>, für <math>i + 1 \leq k \leq n</math>:</p> <p><i>fixed-priority</i> bei Abarbeitung nach absteigender Priorität</p>	<p style="text-align: center;">7 Zugriffskontrolle 7.2 Verdrängungssteuerung</p> <h3 style="text-align: center;">Blockierungszeit (1)</h3> <p style="text-align: center;">Feste obere Schranke</p> <p><b>Verzögerungen</b> nebenläufiger Arbeitsaufträge durch die Zugriffskontrolle <b>sind nach oben begrenzt</b></p> <ul style="list-style-type: none"> <li>▶ die feste obere Schranke <math>bt</math> bestimmt sich aus der größten WCET aller kritischen Abschnitte aller niedriger priorisierten Jobs: <math>\max(cs_k)</math></li> <li>▶ höher priorisierte Jobs werden schlimmstenfalls einmal durch einen niedriger priorisierten Job blockiert</li> </ul> <p>NPCS verzögert eine periodische Task <math>T_i</math> von <math>n</math> periodischen Tasks im taktweisen Betrieb um <math>bt_i = \max(cs_k)</math>, für <math>i + 1 \leq k \leq n</math>:</p> <p><i>fixed-priority</i> bei Abarbeitung nach absteigender Priorität</p> <p><i>dynamic-priority</i> z.B. EDF: Jobs in <math>T_i</math> mit relativem Termin <math>D_i</math> können nur durch Jobs mit längeren relativen Terminen als <math>D_i</math> blockiert werden <math>\Leftrightarrow i &lt; j</math> wenn <math>D_i &lt; D_j</math></p>
<p>soech WS 2005/06 E25 7-15</p>	<p>soech WS 2005/06 E25 7-15</p>

7 Zugriffskontrolle 7.2 Verdrängungsteuerung		7 Zugriffskontrolle 7.2 Verdrängungsteuerung	
<b>Pragmatischer Ansatz</b> Effektiv, bei vergleichsweise geringem Aufwand		<b>Pragmatischer Ansatz</b> Effektiv, bei vergleichsweise geringem Aufwand	
<b>Vorteil:</b> erfordert kein <i>à priori</i> Wissen über Betriebsmittelanforderungen		<b>Vorteil:</b> erfordert kein <i>à priori</i> Wissen über Betriebsmittelanforderungen <ul style="list-style-type: none"> <li>▶ beugt unkontrollierter Prioritätsumkehr vor               <ul style="list-style-type: none"> <li>▶ <math>J_b</math> blockiert nur wenn bei Auslösung <math>J_i</math> bereits das Betriebsmittel hält</li> <li>▶ beendet <math>J_i</math> seinen kritischen Abschnitt, sind alle Betriebsmittel frei</li> <li>▶ Jobs niedrigerer Priorität als <math>J_b</math> können ihm diese nicht streitig machen</li> </ul> </li> </ul>	
soech	WS 2005/06 E25 7-16	soech	WS 2005/06 E25 7-16

7 Zugriffskontrolle 7.2 Verdrängungsteuerung		7 Zugriffskontrolle 7.2 Verdrängungsteuerung	
<b>Pragmatischer Ansatz</b> Effektiv, bei vergleichsweise geringem Aufwand		<b>Pragmatischer Ansatz</b> Effektiv, bei vergleichsweise geringem Aufwand	
<b>Vorteil:</b> erfordert kein <i>à priori</i> Wissen über Betriebsmittelanforderungen <ul style="list-style-type: none"> <li>▶ beugt unkontrollierter Prioritätsumkehr vor               <ul style="list-style-type: none"> <li>▶ <math>J_b</math> blockiert nur wenn bei Auslösung <math>J_i</math> bereits das Betriebsmittel hält</li> <li>▶ beendet <math>J_i</math> seinen kritischen Abschnitt, sind alle Betriebsmittel frei</li> <li>▶ Jobs niedrigerer Priorität als <math>J_b</math> können ihm diese nicht streitig machen</li> </ul> </li> <li>▶ beugt Verklemmung (engl. <i>deadlock</i>) vor, da Nachforderungen von Betriebsmitteln implizit unteilbar geschehen werden               <ul style="list-style-type: none"> <li>▶ eine notwendige Verklemmungsbedingung [3, VIII-60] wird entkräftet</li> <li>▶ genauer: der „<i>hold and wait</i>“ Fall kann nicht eintreten</li> </ul> </li> </ul>		<b>Vorteil:</b> erfordert kein <i>à priori</i> Wissen über Betriebsmittelanforderungen <ul style="list-style-type: none"> <li>▶ beugt unkontrollierter Prioritätsumkehr vor               <ul style="list-style-type: none"> <li>▶ <math>J_b</math> blockiert nur wenn bei Auslösung <math>J_i</math> bereits das Betriebsmittel hält</li> <li>▶ beendet <math>J_i</math> seinen kritischen Abschnitt, sind alle Betriebsmittel frei</li> <li>▶ Jobs niedrigerer Priorität als <math>J_b</math> können ihm diese nicht streitig machen</li> </ul> </li> <li>▶ beugt Verklemmung (engl. <i>deadlock</i>) vor, da Nachforderungen von Betriebsmitteln implizit unteilbar geschehen werden               <ul style="list-style-type: none"> <li>▶ eine notwendige Verklemmungsbedingung [3, VIII-60] wird entkräftet</li> <li>▶ genauer: der „<i>hold and wait</i>“ Fall kann nicht eintreten</li> </ul> </li> <li>▶ einfach zu implementieren; ein gutes Verfahren, wenn...               <ul style="list-style-type: none"> <li>▶ alle Belegungszeiten aller Betriebsmittel kurz sind</li> <li>▶ die meisten Jobs im Konflikt zueinander stehen</li> </ul> </li> </ul>	
soech	WS 2005/06 E25 7-16	soech	WS 2005/06 E25 7-16

<p>7 Zugriffskontrolle 7.2 Verdrängungsteuerung</p> <h3>Pragmatischer Ansatz</h3> <p>Effektiv, bei vergleichsweise geringem Aufwand</p>	<p>7 Zugriffskontrolle 7.2 Verdrängungsteuerung</p> <h3>Pragmatischer Ansatz</h3> <p>mit Schörrahmeflächen Alternativen, sofern bestimmte Voraussetzungen gegeben sind</p>
<p><b>Vorteil:</b> erfordert kein <i>a priori</i> Wissen über Betriebsmittelanforderungen</p> <ul style="list-style-type: none"> <li>▶ beugt unkontrollierter Prioritätsumkehr vor           <ul style="list-style-type: none"> <li>▶ <math>J_n</math> blockiert nur wenn bei Auslösung <math>J_l</math> bereits das Betriebsmittel hält</li> <li>▶ beendet <math>J_l</math> seinen kritischen Abschnitt, sind alle Betriebsmittel frei</li> <li>▶ Jobs niedrigerer Priorität als <math>J_n</math> können ihm diese nicht streitig machen</li> </ul> </li> <li>▶ beugt Verklemmung (engl. <i>deadlock</i>) vor, da Nachforderungen von Betriebsmitteln implizit unteilbar geschehen werden           <ul style="list-style-type: none"> <li>▶ eine notwendige Verklemmungsbedingung [3, VIII-60] wird entkräftet</li> <li>▶ genauer: der „<i>hold and wait</i>“ Fall kann nicht eintreten</li> </ul> </li> <li>▶ einfach zu implementieren; ein gutes Verfahren, wenn...           <ul style="list-style-type: none"> <li>▶ alle Belegungszeiten aller Betriebsmittel kurz sind</li> <li>▶ die meisten Jobs im Konflikt zueinander stehen</li> </ul> </li> </ul> <p>☞ eignet sich für Systeme mit fester und dynamischer Priorität</p>	<p><b>Nachteil:</b> höher priorisierte Jobs können durch niedriger priorisierte Jobs blockiert werden, obwohl zwischen ihnen kein Konflikt besteht</p> <ul style="list-style-type: none"> <li>▶ im Beispiel (S. 7-14) wird <math>J_m</math> durch <math>J_l</math> blockiert, obwohl beide Jobs nicht im gegenseitigen Ausschluss zueinander stehen</li> </ul>
<p>WS 2005/06 E25 7-17</p>	<p>WS 2005/06 E25 7-17</p>

<p>7 Zugriffskontrolle 7.2 Verdrängungsteuerung</p> <h3>Pragmatischer Ansatz</h3> <p>mit Schörrahmeflächen Alternativen, sofern bestimmte Voraussetzungen gegeben sind</p>	<p>7 Zugriffskontrolle 7.3 Prioritätsvererbung</p> <h3>Priorität zeitweise erhöhen</h3> <p>Wechsel zwischen zugewiesene und aktuelle (geerbte) Priorität</p>
<p><b>Nachteil:</b> höher priorisierte Jobs können durch niedriger priorisierte Jobs blockiert werden, obwohl zwischen ihnen kein Konflikt besteht</p> <ul style="list-style-type: none"> <li>▶ im Beispiel (S. 7-14) wird <math>J_m</math> durch <math>J_l</math> blockiert, obwohl beide Jobs nicht im gegenseitigen Ausschluss zueinander stehen</li> </ul> <p>Verbesserungsmöglichkeiten...</p> <ul style="list-style-type: none"> <li>▶ so Verklemmungen nicht auftreten können oder durch eine andere Technik vorgebeugt oder vermieden werden:           <ul style="list-style-type: none"> <li>▶ der ein Betriebsmittel haltende Job wird für die restliche Belegungszeit ggf. auf die Priorität des jeweils anfordernden Jobs hochgesetzt</li> <li>▶ er wird durch <b>Prioritätsvererbung</b> (S. 7-18) „beschleunigt“</li> </ul> </li> <li>▶ so Betriebsmittelanforderungen <i>a priori</i> bekannt sind:           <ul style="list-style-type: none"> <li>▶ der ein Betriebsmittel haltende Job läuft mit der höchsten Priorität aller Jobs, die das Betriebsmittel beanspruchen</li> <li>▶ das Betriebsmittel ist mit einer <b>Prioritätsobergrenze</b> (S. 7-22) assoziiert</li> </ul> </li> </ul>	<p>Arbeitsaufträge werden für die <b>Restzeit der Belegung</b> von (unteilbaren) Betriebsmitteln durch andere Arbeitsaufträge höher priorisiert</p>
<p>WS 2005/06 E25 7-17</p>	<p>WS 2005/06 E25 7-18</p>

7 Zugriffskontrolle 7.3 Prioritätsvererbung

### Priorität zeitweise erhöhen

Wechsel zwischen zugewiesene und aktuelle (geerbte) Priorität

Arbeitsaufträge werden für die **Restzeit der Belegung** von (unteilbaren) Betriebsmitteln durch andere Arbeitsaufträge höher priorisiert

- ▶ fordert ein Job ein gesperrtes Betriebsmittel an, vererbt er seine Priorität an den das Betriebsmittel haltenden Job
  - ▶ der anfordernde Job hat zu dem Zeitpunkt die höchste Priorität
    - ▶ er hat den das Betriebsmittel haltenden Job (indirekt) verdrängt
  - ▶ die Priorität des das Betriebsmittel haltenden Jobs wird erhöht

WS 2005/06 E25 7-18

7 Zugriffskontrolle 7.3 Prioritätsvererbung

### Priorität zeitweise erhöhen

Wechsel zwischen zugewiesene und aktuelle (geerbte) Priorität

Arbeitsaufträge werden für die **Restzeit der Belegung** von (unteilbaren) Betriebsmitteln durch andere Arbeitsaufträge höher priorisiert

- ▶ fordert ein Job ein gesperrtes Betriebsmittel an, vererbt er seine Priorität an den das Betriebsmittel haltenden Job
  - ▶ der anfordernde Job hat zu dem Zeitpunkt die höchste Priorität
    - ▶ er hat den das Betriebsmittel haltenden Job (indirekt) verdrängt
  - ▶ die Priorität des das Betriebsmittel haltenden Jobs wird erhöht
- ▶ gibt der Job das durch ihn gesperrte Betriebsmittel frei, nimmt er die ihm ursprünglich zugewiesene Priorität wieder an
  - ▶ der das Betriebsmittel freigebende Job wird ggf. sofort verdrängt
  - ▶ der auf die Freigabe wartende Job wird ggf. sofort eingelastet

WS 2005/06 E25 7-18

7 Zugriffskontrolle 7.3 Prioritätsvererbung

### Priorität zeitweise erhöhen

Wechsel zwischen zugewiesene und aktuelle (geerbte) Priorität

Arbeitsaufträge werden für die **Restzeit der Belegung** von (unteilbaren) Betriebsmitteln durch andere Arbeitsaufträge höher priorisiert

- ▶ fordert ein Job ein gesperrtes Betriebsmittel an, vererbt er seine Priorität an den das Betriebsmittel haltenden Job
  - ▶ der anfordernde Job hat zu dem Zeitpunkt die höchste Priorität
    - ▶ er hat den das Betriebsmittel haltenden Job (indirekt) verdrängt
  - ▶ die Priorität des das Betriebsmittel haltenden Jobs wird erhöht
- ▶ gibt der Job das durch ihn gesperrte Betriebsmittel frei, nimmt er die ihm ursprünglich zugewiesene Priorität wieder an
  - ▶ der das Betriebsmittel freigebende Job wird ggf. sofort verdrängt
  - ▶ der auf die Freigabe wartende Job wird ggf. sofort eingelastet

Prioritätsumkehr wird nicht wirklich vermieden, jedoch entschärft:

- ▶ wie bei NPCS behalten niedriger priorisierte Jobs die CPU zugeeilt, obwohl höher priorisierte Jobs auf Zuteilung der CPU warten

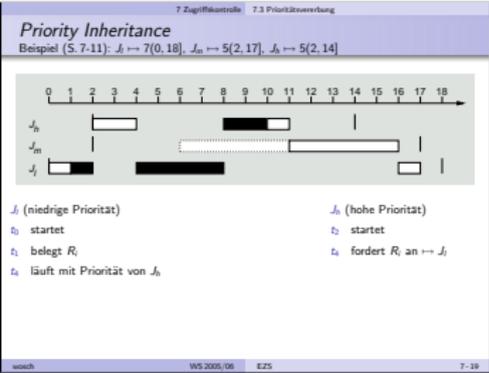
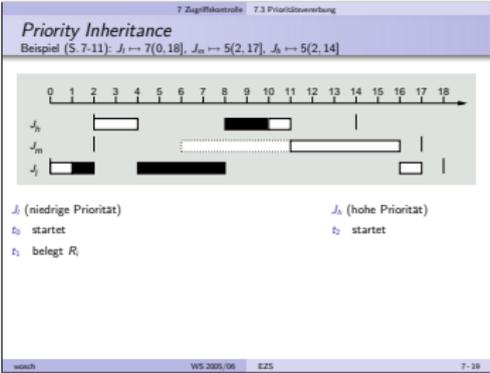
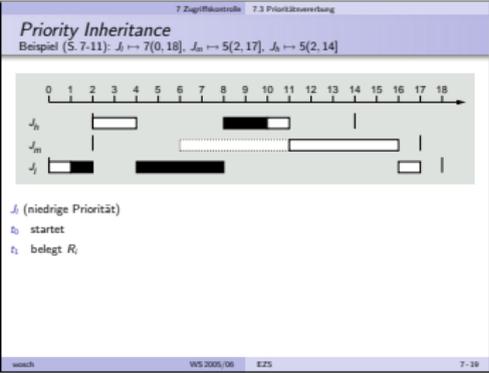
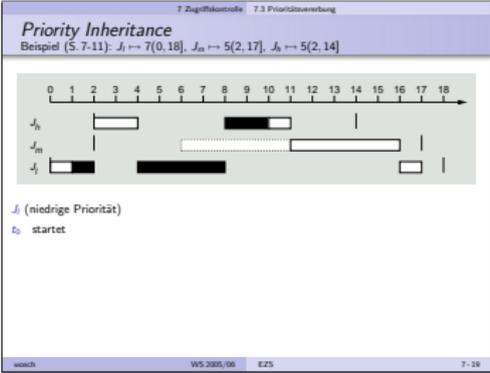
WS 2005/06 E25 7-18

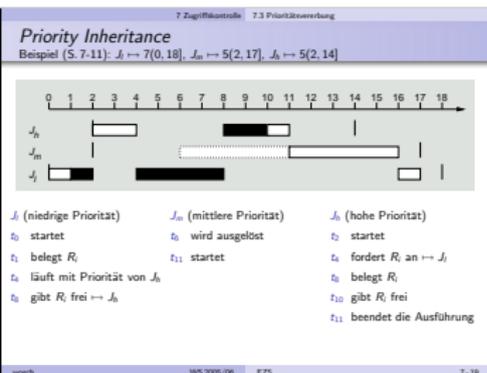
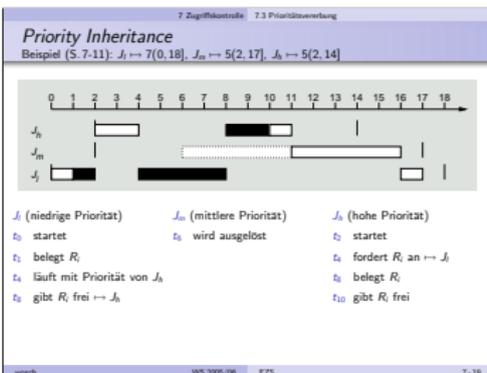
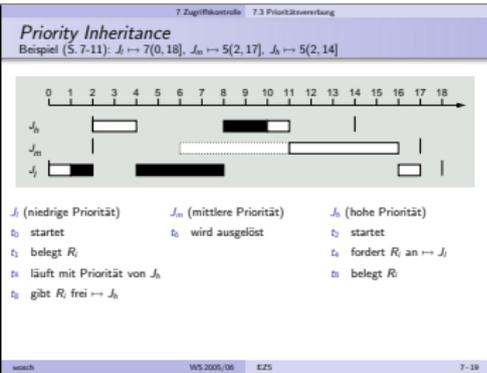
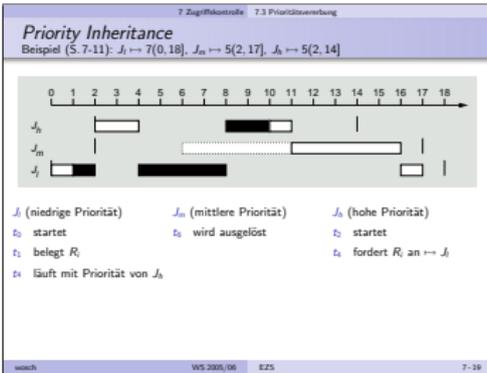
7 Zugriffskontrolle 7.3 Prioritätsvererbung

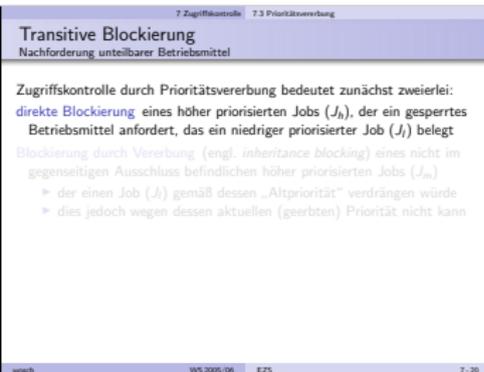
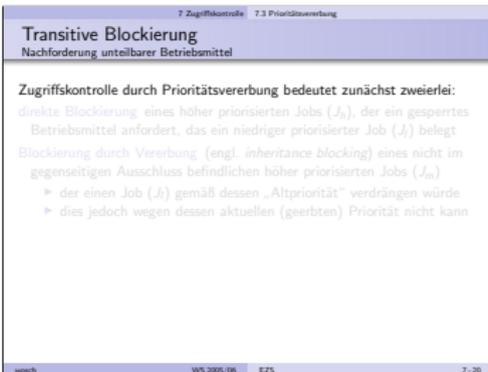
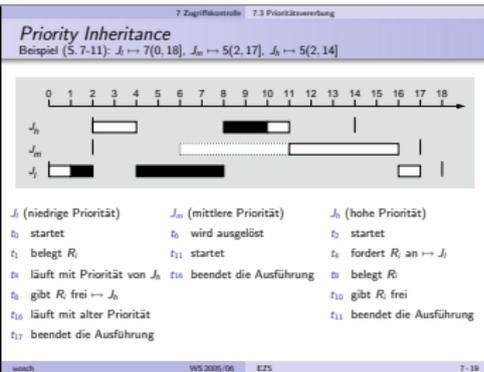
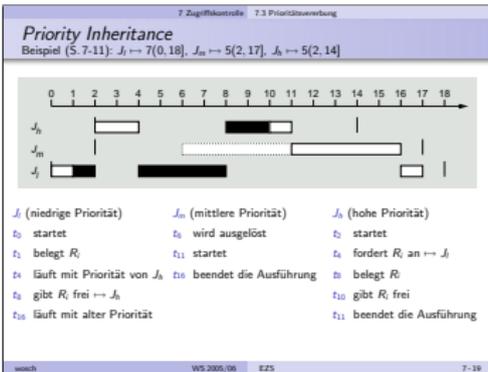
### Priority Inheritance

Beispiel (S. 7-11):  $J_i \mapsto 7(0, 18)$ ,  $J_m \mapsto 5(2, 17)$ ,  $J_b \mapsto 5(2, 14)$

WS 2005/06 E25 7-18







7 Zugriffskontrolle 7.3 Prioritätsvererbung Transitive Blockierung Nachforderung unteilbarer Betriebsmittel	7 Zugriffskontrolle 7.3 Prioritätsvererbung Transitive Blockierung Nachforderung unteilbarer Betriebsmittel
<p>Zugriffskontrolle durch Prioritätsvererbung bedeutet zunächst zweierlei:</p> <p><b>direkte Blockierung</b> eines höher priorisierten Jobs (<math>J_h</math>), der ein gesperrtes Betriebsmittel anfordert, das ein niedriger priorisierter Job (<math>J_l</math>) belegt</p> <p><b>Blockierung durch Vererbung</b> (engl. <i>inheritance blocking</i>) eines nicht im gegenseitigen Ausschluss befindlichen höher priorisierten Jobs (<math>J_m</math>)</p> <ul style="list-style-type: none"> <li>▶ der einen Job (<math>J_l</math>) gemäß dessen „Altpriorität“ verdrängen würde</li> <li>▶ dies jedoch wegen dessen aktuellen (geerbten) Priorität nicht kann</li> </ul>	<p>Zugriffskontrolle durch Prioritätsvererbung bedeutet zunächst zweierlei:</p> <p><b>direkte Blockierung</b> eines höher priorisierten Jobs (<math>J_h</math>), der ein gesperrtes Betriebsmittel anfordert, das ein niedriger priorisierter Job (<math>J_l</math>) belegt</p> <p><b>Blockierung durch Vererbung</b> (engl. <i>inheritance blocking</i>) eines nicht im gegenseitigen Ausschluss befindlichen höher priorisierten Jobs (<math>J_m</math>)</p> <ul style="list-style-type: none"> <li>▶ der einen Job (<math>J_l</math>) gemäß dessen „Altpriorität“ verdrängen würde</li> <li>▶ dies jedoch wegen dessen aktuellen (geerbten) Priorität nicht kann</li> </ul> <p>Blockierungen wirken ggf. transitiv: <b>geschachtelte kritische Abschnitte</b></p>
<small>soech WS 2005/06 E25 7-20</small>	<small>soech WS 2005/06 E25 7-20</small>

7 Zugriffskontrolle 7.3 Prioritätsvererbung Transitive Blockierung Nachforderung unteilbarer Betriebsmittel	7 Zugriffskontrolle 7.3 Prioritätsvererbung Transitive Blockierung Nachforderung unteilbarer Betriebsmittel
<p>Zugriffskontrolle durch Prioritätsvererbung bedeutet zunächst zweierlei:</p> <p><b>direkte Blockierung</b> eines höher priorisierten Jobs (<math>J_h</math>), der ein gesperrtes Betriebsmittel anfordert, das ein niedriger priorisierter Job (<math>J_l</math>) belegt</p> <p><b>Blockierung durch Vererbung</b> (engl. <i>inheritance blocking</i>) eines nicht im gegenseitigen Ausschluss befindlichen höher priorisierten Jobs (<math>J_m</math>)</p> <ul style="list-style-type: none"> <li>▶ der einen Job (<math>J_l</math>) gemäß dessen „Altpriorität“ verdrängen würde</li> <li>▶ dies jedoch wegen dessen aktuellen (geerbten) Priorität nicht kann</li> </ul> <p>Blockierungen wirken ggf. transitiv: <b>geschachtelte kritische Abschnitte</b></p> <ol style="list-style-type: none"> <li>1. <math>J_l</math> startet zuerst, belegt <math>R_1</math> und wird von <math>J_m</math> verdrängt</li> </ol>	<p>Zugriffskontrolle durch Prioritätsvererbung bedeutet zunächst zweierlei:</p> <p><b>direkte Blockierung</b> eines höher priorisierten Jobs (<math>J_h</math>), der ein gesperrtes Betriebsmittel anfordert, das ein niedriger priorisierter Job (<math>J_l</math>) belegt</p> <p><b>Blockierung durch Vererbung</b> (engl. <i>inheritance blocking</i>) eines nicht im gegenseitigen Ausschluss befindlichen höher priorisierten Jobs (<math>J_m</math>)</p> <ul style="list-style-type: none"> <li>▶ der einen Job (<math>J_l</math>) gemäß dessen „Altpriorität“ verdrängen würde</li> <li>▶ dies jedoch wegen dessen aktuellen (geerbten) Priorität nicht kann</li> </ul> <p>Blockierungen wirken ggf. transitiv: <b>geschachtelte kritische Abschnitte</b></p> <ol style="list-style-type: none"> <li>1. <math>J_l</math> startet zuerst, belegt <math>R_1</math> und wird von <math>J_m</math> verdrängt</li> <li>2. <math>J_m</math> belegt <math>R_2</math> und wird von <math>J_h</math> verdrängt</li> </ol>
<small>soech WS 2005/06 E25 7-20</small>	<small>soech WS 2005/06 E25 7-20</small>

7 Zugriffskontrolle 7.3 Prioritätsvererbung Transitive Blockierung Nachforderung unteilbarer Betriebsmittel	7 Zugriffskontrolle 7.3 Prioritätsvererbung Transitive Blockierung Nachforderung unteilbarer Betriebsmittel
<p>Zugriffskontrolle durch Prioritätsvererbung bedeutet zunächst zweierlei:</p> <p><b>direkte Blockierung</b> eines höher priorisierten Jobs (<math>J_h</math>), der ein gesperrtes Betriebsmittel anfordert, das ein niedriger priorisierter Job (<math>J_l</math>) belegt</p> <p><b>Blockierung durch Vererbung</b> (engl. <i>inheritance blocking</i>) eines nicht im gegenseitigen Ausschluss befindlichen höher priorisierten Jobs (<math>J_m</math>)</p> <ul style="list-style-type: none"> <li>▶ der einen Job (<math>J_l</math>) gemäß dessen „Altpriorität“ verdrängen würde</li> <li>▶ dies jedoch wegen dessen aktuellen (geerbten) Priorität nicht kann</li> </ul> <p>Blockierungen wirken ggf. transitiv: <b>geschachtelte kritische Abschnitte</b></p> <ol style="list-style-type: none"> <li>1. <math>J_l</math> startet zuerst, belegt <math>R_1</math> und wird von <math>J_m</math> verdrängt</li> <li>2. <math>J_m</math> belegt <math>R_2</math> und wird von <math>J_h</math> verdrängt</li> <li>3. <math>J_h</math> fordert <math>R_2</math> an und vererbt seine Priorität an <math>J_m</math></li> </ol>	<p>Zugriffskontrolle durch Prioritätsvererbung bedeutet zunächst zweierlei:</p> <p><b>direkte Blockierung</b> eines höher priorisierten Jobs (<math>J_h</math>), der ein gesperrtes Betriebsmittel anfordert, das ein niedriger priorisierter Job (<math>J_l</math>) belegt</p> <p><b>Blockierung durch Vererbung</b> (engl. <i>inheritance blocking</i>) eines nicht im gegenseitigen Ausschluss befindlichen höher priorisierten Jobs (<math>J_m</math>)</p> <ul style="list-style-type: none"> <li>▶ der einen Job (<math>J_l</math>) gemäß dessen „Altpriorität“ verdrängen würde</li> <li>▶ dies jedoch wegen dessen aktuellen (geerbten) Priorität nicht kann</li> </ul> <p>Blockierungen wirken ggf. transitiv: <b>geschachtelte kritische Abschnitte</b></p> <ol style="list-style-type: none"> <li>1. <math>J_l</math> startet zuerst, belegt <math>R_1</math> und wird von <math>J_m</math> verdrängt</li> <li>2. <math>J_m</math> belegt <math>R_2</math> und wird von <math>J_h</math> verdrängt</li> <li>3. <math>J_h</math> fordert <math>R_2</math> an und vererbt seine Priorität an <math>J_m</math></li> <li>4. <math>J_m</math> läuft weiter, fordert <math>R_1</math> an und vererbt „seine“ Priorität an <math>J_l</math></li> </ol>
soech WS 2005/06 E25 7-20	soech WS 2005/06 E25 7-20

7 Zugriffskontrolle 7.3 Prioritätsvererbung Blockierungszeit (2) Feste obere Schranken, die kaskadenartig zur Wirkung kommen können	7 Zugriffskontrolle 7.3 Prioritätsvererbung Blockierungszeit (2) Feste obere Schranken, die kaskadenartig zur Wirkung kommen können
<p>Arbeitsaufträge höherer Priorität werden nur einmal direkt blockiert, wenn die Nachforderung unteilbarer Betriebsmittel ausgeschlossen ist</p> <ul style="list-style-type: none"> <li>▶ für die Dauer der WCET des äußersten kritischen Abschnitts</li> </ul>	<p>Arbeitsaufträge höherer Priorität werden nur einmal direkt blockiert, wenn die Nachforderung unteilbarer Betriebsmittel ausgeschlossen ist</p> <ul style="list-style-type: none"> <li>▶ für die Dauer der WCET des äußersten kritischen Abschnitts</li> </ul> <p><b>Situation des schlimmsten Falls:</b> ein höher priorisierter Job benötigt <math>n &gt; 1</math> Betriebsmittel und steht mit <math>k &gt; 1</math> niedriger priorisierten Jobs im Konflikt</p> <ul style="list-style-type: none"> <li>▶ der höher priorisierte Job kann <math>\min(n, k)</math>-mal blockiert werden</li> <li>▶ jeweils für die Dauer der WCET des äußersten kritischen Abschnitts</li> </ul>
soech WS 2005/06 E25 7-21	soech WS 2005/06 E25 7-21

<p style="text-align: center;">7 Zugriffskontrolle 7.3 Prioritätsvererbung</p> <h3 style="text-align: center;">Blockierungszeit (2)</h3> <p style="text-align: center;">Feste obere Schranken, die kaskadenartig zur Wirkung kommen können</p>	<p style="text-align: center;">7 Zugriffskontrolle 7.3 Prioritätsvererbung</p> <h3 style="text-align: center;">Blockierungszeit (2)</h3> <p style="text-align: center;">Feste obere Schranken, die kaskadenartig zur Wirkung kommen können</p>
<p>Arbeitsaufträge höherer Priorität werden nur einmal direkt blockiert, wenn die Nachforderung unteilbarer Betriebsmittel ausgeschlossen ist</p> <ul style="list-style-type: none"> <li>▶ für die Dauer der WCET des äußersten kritischen Abschnitts</li> </ul> <p><b>Situation des schlimmsten Falls:</b> ein höher priorisierter Job benötigt <math>n &gt; 1</math> Betriebsmittel und steht mit <math>k &gt; 1</math> niedriger priorisierten Jobs im Konflikt</p> <ul style="list-style-type: none"> <li>▶ der höher priorisierte Job kann <math>\min(n, k)</math>-mal blockiert werden</li> <li>▶ jeweils für die Dauer der WCET des äußersten kritischen Abschnitts</li> </ul> <p>GAU: <math>n &gt; 1</math> Betriebsmittel; <math>k &gt; 1</math> Jobs; <math>J_i</math> hat Vorrang vor <math>J_j</math>, wenn <math>i &lt; j</math></p>	<p>Arbeitsaufträge höherer Priorität werden nur einmal direkt blockiert, wenn die Nachforderung unteilbarer Betriebsmittel ausgeschlossen ist</p> <ul style="list-style-type: none"> <li>▶ für die Dauer der WCET des äußersten kritischen Abschnitts</li> </ul> <p><b>Situation des schlimmsten Falls:</b> ein höher priorisierter Job benötigt <math>n &gt; 1</math> Betriebsmittel und steht mit <math>k &gt; 1</math> niedriger priorisierten Jobs im Konflikt</p> <ul style="list-style-type: none"> <li>▶ der höher priorisierte Job kann <math>\min(n, k)</math>-mal blockiert werden</li> <li>▶ jeweils für die Dauer der WCET des äußersten kritischen Abschnitts</li> </ul> <p>GAU: <math>n &gt; 1</math> Betriebsmittel; <math>k &gt; 1</math> Jobs; <math>J_i</math> hat Vorrang vor <math>J_j</math>, wenn <math>i &lt; j</math></p> <ol style="list-style-type: none"> <li>1. <math>J_k</math> startet zuerst, belegt <math>R_n</math></li> </ol>
<p style="text-align: center;">WS 2005/06 E25 7-21</p>	<p style="text-align: center;">WS 2005/06 E25 7-21</p>

<p style="text-align: center;">7 Zugriffskontrolle 7.3 Prioritätsvererbung</p> <h3 style="text-align: center;">Blockierungszeit (2)</h3> <p style="text-align: center;">Feste obere Schranken, die kaskadenartig zur Wirkung kommen können</p>	<p style="text-align: center;">7 Zugriffskontrolle 7.3 Prioritätsvererbung</p> <h3 style="text-align: center;">Blockierungszeit (2)</h3> <p style="text-align: center;">Feste obere Schranken, die kaskadenartig zur Wirkung kommen können</p>
<p>Arbeitsaufträge höherer Priorität werden nur einmal direkt blockiert, wenn die Nachforderung unteilbarer Betriebsmittel ausgeschlossen ist</p> <ul style="list-style-type: none"> <li>▶ für die Dauer der WCET des äußersten kritischen Abschnitts</li> </ul> <p><b>Situation des schlimmsten Falls:</b> ein höher priorisierter Job benötigt <math>n &gt; 1</math> Betriebsmittel und steht mit <math>k &gt; 1</math> niedriger priorisierten Jobs im Konflikt</p> <ul style="list-style-type: none"> <li>▶ der höher priorisierte Job kann <math>\min(n, k)</math>-mal blockiert werden</li> <li>▶ jeweils für die Dauer der WCET des äußersten kritischen Abschnitts</li> </ul> <p>GAU: <math>n &gt; 1</math> Betriebsmittel; <math>k &gt; 1</math> Jobs; <math>J_i</math> hat Vorrang vor <math>J_j</math>, wenn <math>i &lt; j</math></p> <ol style="list-style-type: none"> <li>1. <math>J_k</math> startet zuerst, belegt <math>R_n</math>; <math>J_{k-1}</math> verdrängt <math>J_k</math>, belegt <math>R_{n-1}</math></li> </ol>	<p>Arbeitsaufträge höherer Priorität werden nur einmal direkt blockiert, wenn die Nachforderung unteilbarer Betriebsmittel ausgeschlossen ist</p> <ul style="list-style-type: none"> <li>▶ für die Dauer der WCET des äußersten kritischen Abschnitts</li> </ul> <p><b>Situation des schlimmsten Falls:</b> ein höher priorisierter Job benötigt <math>n &gt; 1</math> Betriebsmittel und steht mit <math>k &gt; 1</math> niedriger priorisierten Jobs im Konflikt</p> <ul style="list-style-type: none"> <li>▶ der höher priorisierte Job kann <math>\min(n, k)</math>-mal blockiert werden</li> <li>▶ jeweils für die Dauer der WCET des äußersten kritischen Abschnitts</li> </ul> <p>GAU: <math>n &gt; 1</math> Betriebsmittel; <math>k &gt; 1</math> Jobs; <math>J_i</math> hat Vorrang vor <math>J_j</math>, wenn <math>i &lt; j</math></p> <ol style="list-style-type: none"> <li>1. <math>J_k</math> startet zuerst, belegt <math>R_n</math>; <math>J_{k-1}</math> verdrängt <math>J_k</math>, belegt <math>R_{n-1}</math>; ...</li> <li>2. <math>J_1</math> verdrängt <math>J_2</math>, belegt <math>R_1</math></li> </ol>
<p style="text-align: center;">WS 2005/06 E25 7-21</p>	<p style="text-align: center;">WS 2005/06 E25 7-21</p>

<p style="text-align: center; font-size: small;">7 Zugriffskontrolle   7.3 Prioritätsvererbung</p> <h3 style="text-align: center;">Blockierungszeit (2)</h3> <p style="text-align: center; font-size: small;">Feste obere Schranken, die kaskadenartig zur Wirkung kommen können</p>	<p style="text-align: center; font-size: small;">7 Zugriffskontrolle   7.4 Prioritätsbergrenzen</p> <h3 style="text-align: center;">Priorität zeitweise deckeln</h3> <p style="text-align: center; font-size: small;">Vorwissen über Arbeitsaufträge und Betriebsmittel</p>
<p>Arbeitsaufträge höherer Priorität werden nur einmal direkt blockiert, wenn die Nachforderung unteilbarer Betriebsmittel ausgeschlossen ist</p> <ul style="list-style-type: none"> <li>▶ für die Dauer der WCET des äußersten kritischen Abschnitts</li> </ul> <p>Situation des schlimmsten Falls: ein höher priorisierter Job benötigt <math>n &gt; 1</math> Betriebsmittel und steht mit <math>k &gt; 1</math> niedriger priorisierten Jobs im Konflikt</p> <ul style="list-style-type: none"> <li>▶ der höher priorisierte Job kann <math>\min(n, k)</math>-mal blockiert werden</li> <li>▶ jeweils für die Dauer der WCET des äußersten kritischen Abschnitts</li> </ul> <p>GAU: <math>n &gt; 1</math> Betriebsmittel; <math>k &gt; 1</math> Jobs; <math>J_i</math> hat Vorrang vor <math>J_j</math>, wenn <math>i &lt; j</math></p> <ol style="list-style-type: none"> <li>1. <math>J_k</math> startet zuerst, belegt <math>R_n</math>; <math>J_{k-1}</math> verdrängt <math>J_k</math>, belegt <math>R_{n-1}</math>; ...</li> <li>2. <math>J_1</math> verdrängt <math>J_2</math>, belegt <math>R_1</math></li> <li>3. <math>J_0</math> verdrängt <math>J_1</math>, fordert <math>R_i</math> an in der Reihenfolge <math>i = 1, 2, 3, \dots, n</math></li> </ol>	<p><b>Prioritätsbergrenze</b> (engl. <i>priority ceiling</i>) eines Betriebsmittels <math>R_i</math> ist die höchste Priorität aller Arbeitsaufträge, die <math>R_i</math> benötigen</p>
<p style="font-size: x-small;">soech   WS 2005/06   E25   7-22</p>	<p style="font-size: x-small;">soech   WS 2005/06   E25   7-22</p>

<p style="text-align: center; font-size: small;">7 Zugriffskontrolle   7.4 Prioritätsbergrenzen</p> <h3 style="text-align: center;">Priorität zeitweise deckeln</h3> <p style="text-align: center; font-size: small;">Vorwissen über Arbeitsaufträge und Betriebsmittel</p>	<p style="text-align: center; font-size: small;">7 Zugriffskontrolle   7.4 Prioritätsbergrenzen</p> <h3 style="text-align: center;">Priorität zeitweise deckeln</h3> <p style="text-align: center; font-size: small;">Vorwissen über Arbeitsaufträge und Betriebsmittel</p>
<p><b>Prioritätsbergrenze</b> (engl. <i>priority ceiling</i>) eines Betriebsmittels <math>R_i</math> ist die höchste Priorität aller Arbeitsaufträge, die <math>R_i</math> benötigen</p> <ul style="list-style-type: none"> <li>▶ die <b>aktuelle Prioritätsbergrenze</b> des Systems gleicht der höchsten Prioritätsbergrenze der zur Zeit belegten Betriebsmittel <ul style="list-style-type: none"> <li>▶ <math>\hat{n}(t)</math>, in Abhängigkeit vom betrachteten Zeitpunkt <math>t</math></li> </ul> </li> </ul>	<p><b>Prioritätsbergrenze</b> (engl. <i>priority ceiling</i>) eines Betriebsmittels <math>R_i</math> ist die höchste Priorität aller Arbeitsaufträge, die <math>R_i</math> benötigen</p> <ul style="list-style-type: none"> <li>▶ die <b>aktuelle Prioritätsbergrenze</b> des Systems gleicht der höchsten Prioritätsbergrenze der zur Zeit belegten Betriebsmittel <ul style="list-style-type: none"> <li>▶ <math>\hat{n}(t)</math>, in Abhängigkeit vom betrachteten Zeitpunkt <math>t</math></li> </ul> </li> <li>▶ sie ist kein unteilbares Betriebsmittel beachtet, existiert die <b>aktuelle Prioritätsbergrenze</b> (theoretisch) nicht <ul style="list-style-type: none"> <li>▶ sie ist dann niedriger als die niedrigste Priorität aller Jobs</li> </ul> </li> </ul>
<p style="font-size: x-small;">soech   WS 2005/06   E25   7-22</p>	<p style="font-size: x-small;">soech   WS 2005/06   E25   7-22</p>

<p style="text-align: right;">7 Zugriffskontrolle 7.4 Prioritätsbergangen</p> <h3 style="text-align: center;">Priorität zeitweise deckeln</h3> <p style="text-align: center;">Vorwissen über Arbeitsaufträge und Betriebsmittel</p> <p><b>Prioritätsobergrenze</b> (engl. <i>priority ceiling</i>) eines Betriebsmittels <math>R_i</math> ist die höchste Priorität aller Arbeitsaufträge, die <math>R_i</math> benötigen</p> <ul style="list-style-type: none"> <li>▶ die <b>aktuelle Prioritätsobergrenze</b> des Systems gleicht der höchsten Prioritätsobergrenze der zur Zeit belegten Betriebsmittel <ul style="list-style-type: none"> <li>▶ <math>\hat{\Pi}(t)</math>, in Abhängigkeit vom betrachteten Zeitpunkt <math>t</math></li> </ul> </li> <li>▶ ist kein unteilbares Betriebsmittel belegt, existiert die <b>aktuelle Prioritätsobergrenze</b> (theoretisch) nicht <ul style="list-style-type: none"> <li>▶ sie ist dann niedriger als die niedrigste Priorität aller Jobs</li> </ul> </li> <li>▶ die jeweiligen Werte sind für alle Betriebsmittel im Voraus bekannt</li> </ul> <p style="text-align: center;">hoch WS 2005/06 E25 7-22</p>	<p style="text-align: right;">7 Zugriffskontrolle 7.4 Prioritätsbergangen</p> <h3 style="text-align: center;">Priorität zeitweise deckeln</h3> <p style="text-align: center;">Vorwissen über Arbeitsaufträge und Betriebsmittel</p> <p><b>Prioritätsobergrenze</b> (engl. <i>priority ceiling</i>) eines Betriebsmittels <math>R_i</math> ist die höchste Priorität aller Arbeitsaufträge, die <math>R_i</math> benötigen</p> <ul style="list-style-type: none"> <li>▶ die <b>aktuelle Prioritätsobergrenze</b> des Systems gleicht der höchsten Prioritätsobergrenze der zur Zeit belegten Betriebsmittel <ul style="list-style-type: none"> <li>▶ <math>\hat{\Pi}(t)</math>, in Abhängigkeit vom betrachteten Zeitpunkt <math>t</math></li> </ul> </li> <li>▶ ist kein unteilbares Betriebsmittel belegt, existiert die <b>aktuelle Prioritätsobergrenze</b> (theoretisch) nicht <ul style="list-style-type: none"> <li>▶ sie ist dann niedriger als die niedrigste Priorität aller Jobs</li> </ul> </li> <li>▶ die jeweiligen Werte sind für alle Betriebsmittel im Voraus bekannt</li> </ul> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p>Arbeitsaufträge können während ihrer Ausführung (d.h., bei Anforderung eines gesperrten Betriebsmittels) eine durch <math>x</math> begrenzte Priorität erben</p> <ul style="list-style-type: none"> <li>▶ wenn sie ein Betriebsmittel mit Prioritätsobergrenze <math>x</math> benötigen</li> </ul> </div> <p style="text-align: center;">hoch WS 2005/06 E25 7-22</p>
--	--

<p style="text-align: right;">7 Zugriffskontrolle 7.4 Prioritätsbergangen</p> <h3 style="text-align: center;">Priorität zeitweise deckeln</h3> <p style="text-align: center;">Vorwissen über Arbeitsaufträge und Betriebsmittel</p> <p><b>Prioritätsobergrenze</b> (engl. <i>priority ceiling</i>) eines Betriebsmittels <math>R_i</math> ist die höchste Priorität aller Arbeitsaufträge, die <math>R_i</math> benötigen</p> <ul style="list-style-type: none"> <li>▶ die <b>aktuelle Prioritätsobergrenze</b> des Systems gleicht der höchsten Prioritätsobergrenze der zur Zeit belegten Betriebsmittel <ul style="list-style-type: none"> <li>▶ <math>\hat{\Pi}(t)</math>, in Abhängigkeit vom betrachteten Zeitpunkt <math>t</math></li> </ul> </li> <li>▶ ist kein unteilbares Betriebsmittel belegt, existiert die <b>aktuelle Prioritätsobergrenze</b> (theoretisch) nicht <ul style="list-style-type: none"> <li>▶ sie ist dann niedriger als die niedrigste Priorität aller Jobs</li> </ul> </li> <li>▶ die jeweiligen Werte sind für alle Betriebsmittel im Voraus bekannt</li> </ul> <div style="border: 1px solid black; padding: 5px; margin-top: 10px;"> <p>Arbeitsaufträge können während ihrer Ausführung (d.h., bei Anforderung eines gesperrten Betriebsmittels) eine durch <math>x</math> begrenzte Priorität erben</p> <ul style="list-style-type: none"> <li>▶ wenn sie ein Betriebsmittel mit Prioritätsobergrenze <math>x</math> benötigen</li> </ul> </div> <p>☞ Prioritätsobergrenzen sind eine Variante von Prioritätsvererbung</p> <p style="text-align: center;">hoch WS 2005/06 E25 7-22</p>	<p style="text-align: right;">7 Zugriffskontrolle 7.4 Prioritätsbergangen</p> <h3 style="text-align: center;">Betriebsmittelvergabe und Prioritätsvererbung</h3> <p style="text-align: center;">Vorwissen über Arbeitsaufträge und Betriebsmittel</p> <p>Vergabe von Betriebsmittel <math>R</math> zum Zeitpunkt <math>t</math> an Arbeitsauftrag <math>J</math> hängt ab vom Zustand von <math>R</math> und von der aktuellen Priorität <math>\pi(t)</math> von <math>J</math>:</p> <p style="text-align: center;">hoch WS 2005/06 E25 7-23</p>
--	--

7 Zugriffskontrolle 7.4 Prioritätsbegrenzen		7 Zugriffskontrolle 7.4 Prioritätsbegrenzen	
<b>Betriebsmittelvergabe und Prioritätsvererbung</b> Vorwissen über Arbeitsaufträge und Betriebsmittel		<b>Betriebsmittelvergabe und Prioritätsvererbung</b> Vorwissen über Arbeitsaufträge und Betriebsmittel	
Vergabe von Betriebsmittel $R$ zum Zeitpunkt $t$ an Arbeitsauftrag $J$ hängt ab vom Zustand von $R$ und von der aktuellen Priorität $\pi(t)$ von $J$ : <b>belegt</b> $\mapsto R$ ist gesperrt, $J$ blockiert		Vergabe von Betriebsmittel $R$ zum Zeitpunkt $t$ an Arbeitsauftrag $J$ hängt ab vom Zustand von $R$ und von der aktuellen Priorität $\pi(t)$ von $J$ : <b>belegt</b> $\mapsto R$ ist gesperrt, $J$ blockiert <b>frei</b> $\mapsto R$ wird $J$ zugeteilt und gesperrt, falls...	
soech	WS 2005/06 E25	soech	WS 2005/06 E25
	7-22		7-23

7 Zugriffskontrolle 7.4 Prioritätsbegrenzen		7 Zugriffskontrolle 7.4 Prioritätsbegrenzen	
<b>Betriebsmittelvergabe und Prioritätsvererbung</b> Vorwissen über Arbeitsaufträge und Betriebsmittel		<b>Betriebsmittelvergabe und Prioritätsvererbung</b> Vorwissen über Arbeitsaufträge und Betriebsmittel	
Vergabe von Betriebsmittel $R$ zum Zeitpunkt $t$ an Arbeitsauftrag $J$ hängt ab vom Zustand von $R$ und von der aktuellen Priorität $\pi(t)$ von $J$ : <b>belegt</b> $\mapsto R$ ist gesperrt, $J$ blockiert <b>frei</b> $\mapsto R$ wird $J$ zugeteilt und gesperrt, falls... <ol style="list-style-type: none"> <li><math>\pi(t) &gt; \hat{\Pi}(t)</math>: <math>J</math> hat die aktuell höchste Priorität</li> </ol>		Vergabe von Betriebsmittel $R$ zum Zeitpunkt $t$ an Arbeitsauftrag $J$ hängt ab vom Zustand von $R$ und von der aktuellen Priorität $\pi(t)$ von $J$ : <b>belegt</b> $\mapsto R$ ist gesperrt, $J$ blockiert <b>frei</b> $\mapsto R$ wird $J$ zugeteilt und gesperrt, falls... <ol style="list-style-type: none"> <li><math>\pi(t) &gt; \hat{\Pi}(t)</math>: <math>J</math> hat die aktuell höchste Priorität</li> <li><math>\pi(t) \leq \hat{\Pi}(t)</math>: <math>J</math> ist ein Job, der zum Zeitpunkt <math>t</math> mindestens ein Betriebsmittel mit Prioritätsobergrenze <math>\hat{\Pi}(t)</math> hält               <ul style="list-style-type: none"> <li>anderefalls bleibt <math>R</math> frei und <math>J</math> blockiert (S. 7-24)</li> </ul> </li> </ol>	
soech	WS 2005/06 E25	soech	WS 2005/06 E25
	7-21		7-23

7 Zugriffskontrolle 7.4 Prioritätsbergangen	7 Zugriffskontrolle 7.4 Prioritätsbergangen
<h3 style="text-align: center;">Betriebsmittelvergabe und Prioritätsvererbung</h3> <p style="text-align: center;">Vorwissen über Arbeitsaufträge und Betriebsmittel</p> <p>Vergabe von Betriebsmittel <math>R</math> zum Zeitpunkt <math>t</math> an Arbeitsauftrag <math>J</math> hängt ab vom Zustand von <math>R</math> und von der aktuellen Priorität <math>\pi(t)</math> von <math>J</math>:</p> <p><b>belegt</b> <math>\mapsto R</math> ist gesperrt, <math>J</math> blockiert</p> <p><b>frei</b> <math>\mapsto R</math> wird <math>J</math> zugeteilt und gesperrt, falls...</p> <ol style="list-style-type: none"> <li><math>\pi(t) &gt; \hat{\Pi}(t)</math>: <math>J</math> hat die aktuell höchste Priorität</li> <li><math>\pi(t) \leq \hat{\Pi}(t)</math>: <math>J</math> ist ein Job, der zum Zeitpunkt <math>t</math> mindestens ein Betriebsmittel mit Prioritätsobergrenze <math>\hat{\Pi}(t)</math> hält <ul style="list-style-type: none"> <li>anderenfalls bleibt <math>R</math> frei und <math>J</math> blockiert (S. 7-24)</li> </ul> </li> </ol> <p>Prioritätsvererbung findet (auch hier) nur statt, wenn <math>J</math> suspendiert wird:</p> <ul style="list-style-type: none"> <li><math>J_i</math>, der <math>J</math> blockiert, erbt die aktuelle Priorität <math>\pi(t)</math> von <math>J</math></li> <li><math>J_j</math> behält diese Priorität, bis er alle Betriebsmittel freigibt, deren Prioritätsobergrenze größer oder gleich <math>\pi(t)</math> ist <ul style="list-style-type: none"> <li>er nimmt dann Priorität <math>\pi(t')</math> an, die er zum Zeitpunkt <math>t' &lt; t</math> hatte</li> <li>d.h., als ihm die freigegebenen Betriebsmittel zugeteilt wurden</li> </ul> </li> </ul>	<h3 style="text-align: center;">Verklemmungsvorbeugung</h3> <p style="text-align: center;">Enkräftung der hinreichenden Bedingung [3, VIII-60]: zirkulares Warten</p> <p>Betriebsmittelvergabe kontrolliert durch Prioritätsbergangen ist weniger „gefällig“ (engl. <i>greedy</i>), als bloße Prioritätsvererbung</p> <ul style="list-style-type: none"> <li>die Anforderung von <math>J</math> kann zurückgewiesen werden, obwohl das angeforderte Betriebsmittel <math>R</math> frei ist</li> </ul>
sooch WS 2005/06 E25 7-22	sooch WS 2005/06 E25 7-24

7 Zugriffskontrolle 7.4 Prioritätsbergangen	7 Zugriffskontrolle 7.4 Prioritätsbergangen
<h3 style="text-align: center;">Verklemmungsvorbeugung</h3> <p style="text-align: center;">Enkräftung der hinreichenden Bedingung [3, VIII-60]: zirkulares Warten</p> <p>Betriebsmittelvergabe kontrolliert durch Prioritätsbergangen ist weniger „gefällig“ (engl. <i>greedy</i>), als bloße Prioritätsvererbung</p> <ul style="list-style-type: none"> <li>die Anforderung von <math>J</math> kann zurückgewiesen werden, obwohl das angeforderte Betriebsmittel <math>R</math> frei ist</li> <li>dies ist der Fall, wenn die durch die Menge von Prioritätsbergangen definierte (ansteigende) <b>lineare Ordnung</b> verletzt werden sollte <ul style="list-style-type: none"> <li><math>\pi(t) \leq \hat{\Pi}(t)</math> trifft zu und <math>J</math> hält kein Betriebsmittel mit <math>\hat{\Pi}(t)</math></li> <li>d.h., die direkte/indirekte Priorität von <math>J</math> durchbricht die Ordnung</li> </ul> </li> <li>alle Betriebsmittel des Systems sind linear geordnet aufgestellt</li> </ul>	<h3 style="text-align: center;">Verklemmungsvorbeugung</h3> <p style="text-align: center;">Enkräftung der hinreichenden Bedingung [3, VIII-60]: zirkulares Warten</p> <p>Betriebsmittelvergabe kontrolliert durch Prioritätsbergangen ist weniger „gefällig“ (engl. <i>greedy</i>), als bloße Prioritätsvererbung</p> <ul style="list-style-type: none"> <li>die Anforderung von <math>J</math> kann zurückgewiesen werden, obwohl das angeforderte Betriebsmittel <math>R</math> frei ist</li> <li>dies ist der Fall, wenn die durch die Menge von Prioritätsbergangen definierte (ansteigende) <b>lineare Ordnung</b> verletzt werden sollte <ul style="list-style-type: none"> <li><math>\pi(t) \leq \hat{\Pi}(t)</math> trifft zu und <math>J</math> hält kein Betriebsmittel mit <math>\hat{\Pi}(t)</math></li> <li>d.h., die direkte/indirekte Priorität von <math>J</math> durchbricht die Ordnung</li> </ul> </li> <li>alle Betriebsmittel des Systems sind linear geordnet aufgestellt</li> </ul> <p>Blockierung durch Prioritätsbergangen wird auch als <b>Aufhebungssperre</b> (engl. <i>avoidance blocking</i>) bezeichnet</p> <ul style="list-style-type: none"> <li>da implizit die Kosten anfallen, um auch Verklemmungen vorzubeugen</li> </ul>
sooch WS 2005/06 E25 7-24	sooch WS 2005/06 E25 7-24

7 Zugriffskontrolle 7.4 Prioritätsbegrenzen	7 Zugriffskontrolle 7.4 Prioritätsbegrenzen
<b>Blockierungszeit (3)</b> Feste obere Schranke	<b>Blockierungszeit (3)</b> Feste obere Schranke
Zugriffskontrolle durch Prioritätsbegrenzen impliziert drei Arten der Blockierung nebenläufiger Arbeitsaufträge: <ol style="list-style-type: none"> <li>1. direkte Blockierung,</li> <li>2. Blockierung durch Vererbung,</li> <li>3. Blockierung in Form einer Aufhebungssperre</li> </ol>	Zugriffskontrolle durch Prioritätsbegrenzen impliziert drei Arten der Blockierung nebenläufiger Arbeitsaufträge: <ol style="list-style-type: none"> <li>1. direkte Blockierung,</li> <li>2. Blockierung durch Vererbung,</li> <li>3. Blockierung in Form einer Aufhebungssperre</li> </ol> <div style="text-align: right; color: red;">} <b>Prioritätsvererbung</b></div>
<small>soech WS 2005/06 E25 7-25</small>	<small>soech WS 2005/06 E25 7-25</small>

7 Zugriffskontrolle 7.4 Prioritätsbegrenzen	7 Zugriffskontrolle 7.4 Prioritätsbegrenzen
<b>Blockierungszeit (3)</b> Feste obere Schranke	<b>Blockierungszeit (3)</b> Feste obere Schranke
Zugriffskontrolle durch Prioritätsbegrenzen impliziert drei Arten der Blockierung nebenläufiger Arbeitsaufträge: <ol style="list-style-type: none"> <li>1. direkte Blockierung,</li> <li>2. Blockierung durch Vererbung,</li> <li>3. Blockierung in Form einer Aufhebungssperre</li> </ol> <div style="text-align: right; color: red;">} <b>Prioritätsvererbung</b></div> <p>Effekt von 3. ist, dass jeder Arbeitsauftrag höchstens einmal blockiert und dass eine Blockierung nicht transitiv ist [28]</p>	Zugriffskontrolle durch Prioritätsbegrenzen impliziert drei Arten der Blockierung nebenläufiger Arbeitsaufträge: <ol style="list-style-type: none"> <li>1. direkte Blockierung,</li> <li>2. Blockierung durch Vererbung,</li> <li>3. Blockierung in Form einer Aufhebungssperre</li> </ol> <div style="text-align: right; color: red;">} <b>Prioritätsvererbung</b></div> <p>Effekt von 3. ist, dass jeder Arbeitsauftrag höchstens einmal blockiert und dass eine Blockierung nicht transitiv ist [28]</p> <ul style="list-style-type: none"> <li>▶ die Blockierungszeit ist begrenzt durch die größte WCET aller kritischen Abschnitte aller niedriger priorisierten Jobs</li> </ul>
<small>soech WS 2005/06 E25 7-25</small>	<small>soech WS 2005/06 E25 7-25</small>

7 Zugriffskontrolle | 7.4 Prioritätsbergewesen

## Blockierungszeit (3)

Feste obere Schranke

Zugriffskontrolle durch Prioritätsbergrenzen impliziert drei Arten der Blockierung nebenläufiger Arbeitsaufträge:

1. direkte Blockierung,
2. Blockierung durch Vererbung, } **Prioritätsvererbung**
3. Blockierung in Form einer Aufhebungssperre

Effekt von 3. ist, dass jeder Arbeitsauftrag höchstens einmal blockiert und dass eine Blockierung nicht transitiv ist [28]

- ▶ die Blockierungszeit ist begrenzt durch die größte WCET aller kritischen Abschnitte aller niedriger priorisierten Jobs
- ▶ unabhängig von der Anzahl der im Konflikt stehenden Jobs
  - (a) wenn ein Job blockiert, dann nur durch höchstens einen Job
  - (b) ein Job, der einen anderen blockiert, wird von keinem anderen blockiert

swoch      WS 2005/06      E25      7-26

7 Zugriffskontrolle | 7.4 Prioritätsbergewesen

## Priority Ceiling

Jobs blockieren nicht transitiv und höchstens einmal im Konfliktfall

swoch      WS 2005/06      E25      7-26

7 Zugriffskontrolle | 7.4 Prioritätsbergewesen

## Priority Ceiling

Jobs blockieren nicht transitiv und höchstens einmal im Konfliktfall

$J_i$  (niedrige Priorität)  
 $t_0$  startet

swoch      WS 2005/06      E25      7-26

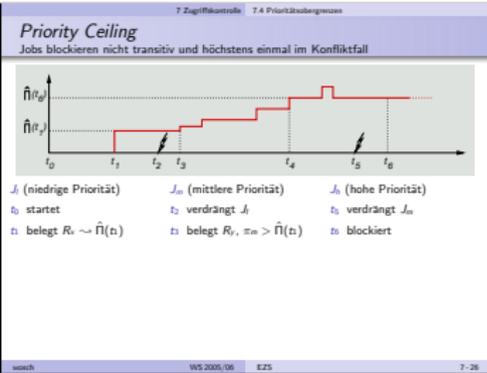
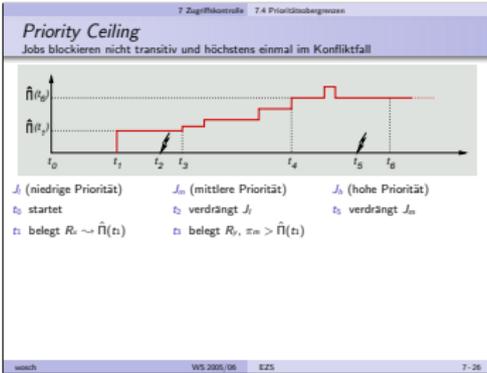
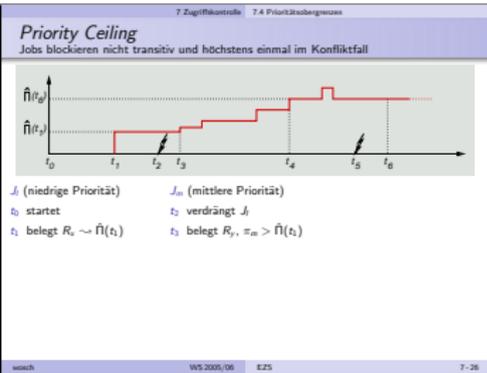
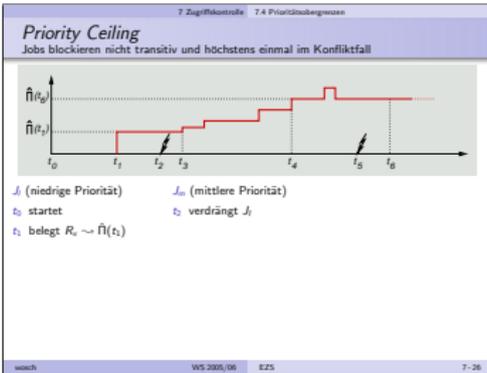
7 Zugriffskontrolle | 7.4 Prioritätsbergewesen

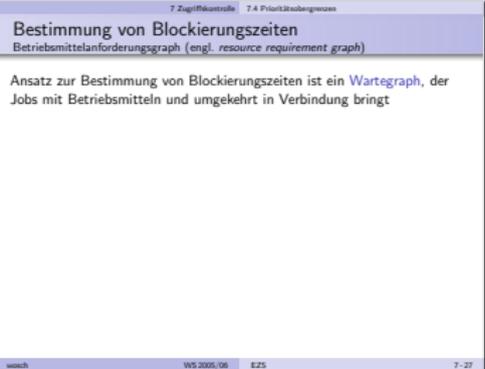
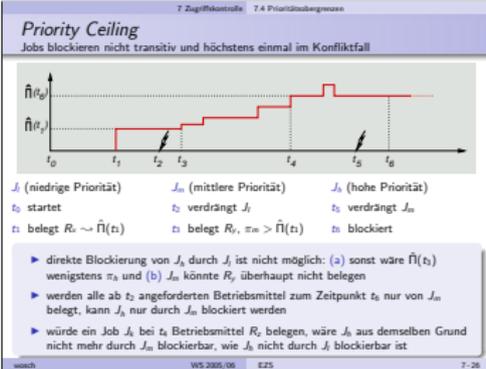
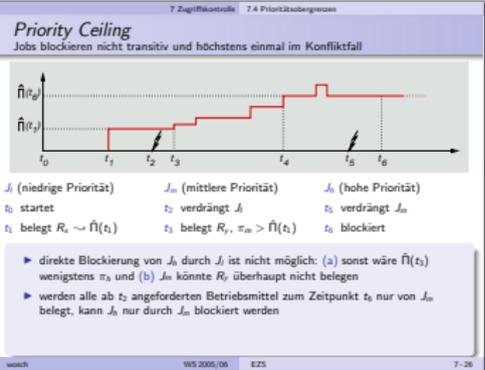
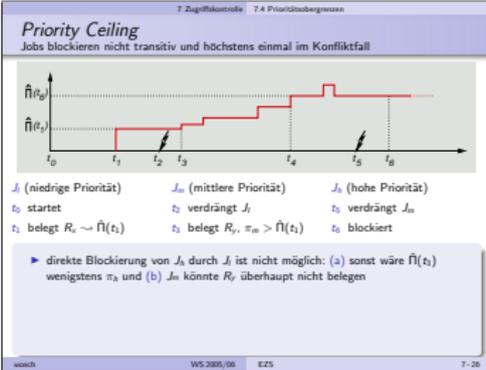
## Priority Ceiling

Jobs blockieren nicht transitiv und höchstens einmal im Konfliktfall

$J_i$  (niedrige Priorität)  
 $t_0$  startet  
 $t_1$  belegt  $R_i \rightsquigarrow \hat{n}(t_1)$

swoch      WS 2005/06      E25      7-26





7 Zugriffskontrolle 7.4 Prioritätsbergbau		7 Zugriffskontrolle 7.4 Prioritätsbergbau	
<b>Bestimmung von Blockierungszeiten</b> Betriebsmittelanforderungsgraph (engl. <i>resource requirement graph</i> )		<b>Bestimmung von Blockierungszeiten</b> Betriebsmittelanforderungsgraph (engl. <i>resource requirement graph</i> )	
Ansatz zur Bestimmung von Blockierungszeiten ist ein <b>Wartegraph</b> , der Jobs mit Betriebsmitteln und umgekehrt in Verbindung bringt <b>Knoten</b> repräsentieren Jobs bzw. Betriebsmittel, annotiert mit einem Namen und der Betriebsmittelanzahl (nur Betriebsmittelknoten)		Ansatz zur Bestimmung von Blockierungszeiten ist ein <b>Wartegraph</b> , der Jobs mit Betriebsmitteln und umgekehrt in Verbindung bringt <b>Knoten</b> repräsentieren Jobs bzw. Betriebsmittel, annotiert mit einem Namen und der Betriebsmittelanzahl (nur Betriebsmittelknoten) <b>Kanten</b> drücken „wartet auf“- bzw. „belegt von“-Beziehungen aus und sind mit einem 2-Tupel $(n, t)$ annotiert ▶ Betriebsmittelanzahl $n$ , Belegungszeit $t$	
soech	WS 2005/06 E25	soech	WS 2005/06 E25
	7-27		7-27

7 Zugriffskontrolle 7.4 Prioritätsbergbau		7 Zugriffskontrolle 7.4 Prioritätsbergbau	
<b>Bestimmung von Blockierungszeiten</b> Betriebsmittelanforderungsgraph (engl. <i>resource requirement graph</i> )		<b>Bestimmung von Blockierungszeiten</b> Betriebsmittelanforderungsgraph (engl. <i>resource requirement graph</i> )	
Ansatz zur Bestimmung von Blockierungszeiten ist ein <b>Wartegraph</b> , der Jobs mit Betriebsmitteln und umgekehrt in Verbindung bringt <b>Knoten</b> repräsentieren Jobs bzw. Betriebsmittel, annotiert mit einem Namen und der Betriebsmittelanzahl (nur Betriebsmittelknoten) <b>Kanten</b> drücken „wartet auf“- bzw. „belegt von“-Beziehungen aus und sind mit einem 2-Tupel $(n, t)$ annotiert ▶ Betriebsmittelanzahl $n$ , Belegungszeit $t$ ▶ Pfade zwischen Jobknoten führen immer über Betriebsmittelknoten ▶ $J_i$ belegt $n$ Einheiten $R_i$ für $t$ Zeiten, worauf $J_j$ wartet ▶ führt ein Pfad von $J_k$ zu $J_l$ , ist $J_k$ durch $J_l$ direkt blockiert		Ansatz zur Bestimmung von Blockierungszeiten ist ein <b>Wartegraph</b> , der Jobs mit Betriebsmitteln und umgekehrt in Verbindung bringt <b>Knoten</b> repräsentieren Jobs bzw. Betriebsmittel, annotiert mit einem Namen und der Betriebsmittelanzahl (nur Betriebsmittelknoten) <b>Kanten</b> drücken „wartet auf“- bzw. „belegt von“-Beziehungen aus und sind mit einem 2-Tupel $(n, t)$ annotiert ▶ Betriebsmittelanzahl $n$ , Belegungszeit $t$ ▶ Pfade zwischen Jobknoten führen immer über Betriebsmittelknoten ▶ $J_i$ belegt $n$ Einheiten $R_i$ für $t$ Zeiten, worauf $J_j$ wartet ▶ führt ein Pfad von $J_k$ zu $J_l$ , ist $J_k$ durch $J_l$ direkt blockiert ▶ Zyklen im Wartegraph zeigen <b>Verklümmungen</b> an $\leadsto$ „Abfallprodukt“ <b>Verklümmungserkennung</b> (engl. <i>deadlock detection</i> ) beim Entwurf ▶ Verklümmungsvorbereitung zur Laufzeit als Zusicherung (engl. <i>assertion</i> )	
soech	WS 2005/06 E25	soech	WS 2005/06 E25
	7-27		7-27

7 Zugriffskontrolle 7.4 Prioritätsübergewinn		7 Zugriffskontrolle 7.4 Prioritätsübergewinn	
<b>Vereinfachung durch Stapelorientierung</b> Stapelbezogene Einplanung (engl. <i>stack-based scheduling</i> )		<b>Vereinfachung durch Stapelorientierung</b> Stapelbezogene Einplanung (engl. <i>stack-based scheduling</i> )	
Ausgangspunkt ist die <b>stapelorientierte Einplanung von Prozessen</b> , so dass $N$ Prozesse einen Stapel gemeinsam nutzen können [29, 30]		Ausgangspunkt ist die <b>stapelorientierte Einplanung von Prozessen</b> , so dass $N$ Prozesse einen Stapel gemeinsam nutzen können [29, 30] <ul style="list-style-type: none"> <li>▶ nicht immer ist es möglich, jeden Job durch einen eigenen Faden zu repräsentieren bzw. mit einem eigenen Stapel zu versehen               <ul style="list-style-type: none"> <li>▪ wenn die Jobanzahl zu hoch und/oder der Speicherplatz zu gering ist</li> </ul> </li> </ul>	
sooch	WS 2005/06 E25	sooch	WS 2005/06 E25

7 Zugriffskontrolle 7.4 Prioritätsübergewinn		7 Zugriffskontrolle 7.4 Prioritätsübergewinn	
<b>Vereinfachung durch Stapelorientierung</b> Stapelbezogene Einplanung (engl. <i>stack-based scheduling</i> )		<b>Vereinfachung durch Stapelorientierung</b> Stapelbezogene Einplanung (engl. <i>stack-based scheduling</i> )	
Ausgangspunkt ist die <b>stapelorientierte Einplanung von Prozessen</b> , so dass $N$ Prozesse einen Stapel gemeinsam nutzen können [29, 30] <ul style="list-style-type: none"> <li>▶ nicht immer ist es möglich, jeden Job durch einen eigenen Faden zu repräsentieren bzw. mit einem eigenen Stapel zu versehen               <ul style="list-style-type: none"> <li>▪ wenn die Jobanzahl zu hoch und/oder der Speicherplatz zu gering ist</li> </ul> </li> <li>▶ gemeinsame Nutzung desselben Stapels setzt voraus, dass kein Job bei Anforderung eines gemeinsamen Betriebsmittels blockiert               <ul style="list-style-type: none"> <li>▪ sonst droht die Überschreibung der Stapelbereiche anderer Jobs ☹</li> </ul> </li> </ul>		Ausgangspunkt ist die <b>stapelorientierte Einplanung von Prozessen</b> , so dass $N$ Prozesse einen Stapel gemeinsam nutzen können [29, 30] <ul style="list-style-type: none"> <li>▶ nicht immer ist es möglich, jeden Job durch einen eigenen Faden zu repräsentieren bzw. mit einem eigenen Stapel zu versehen               <ul style="list-style-type: none"> <li>▪ wenn die Jobanzahl zu hoch und/oder der Speicherplatz zu gering ist</li> </ul> </li> <li>▶ gemeinsame Nutzung desselben Stapels setzt voraus, dass kein Job bei Anforderung eines gemeinsamen Betriebsmittels blockiert               <ul style="list-style-type: none"> <li>▪ sonst droht die Überschreibung der Stapelbereiche anderer Jobs ☹</li> </ul> </li> <li>▶ die <b>Jobs dürfen ihre Ausführung niemals selbst aussetzen</b>, sie dürfen jedoch von höher priorisierten Jobs verdrängt werden               <ul style="list-style-type: none"> <li>▪ oben auf dem Stapel läuft immer der Job mit der höchsten Priorität</li> <li>▪ die logische Konsequenz, wenn Selbstaussetzung ausgeschlossen ist</li> </ul> </li> </ul>	
sooch	WS 2005/06 E25	sooch	WS 2005/06 E25

<small>7 Zugriffskontrolle 7.4 Prioritätsbergrenzen</small> <b>Vereinfachung durch Stapelorientierung</b> Stapelbezogene Einplanung (engl. <i>stack-based scheduling</i> )	<small>7 Zugriffskontrolle 7.4 Prioritätsbergrenzen</small> <b>Stapelbezogene Grenzprioritäten</b> Regeln
<p>Ausgangspunkt ist die <b>stapelorientierte Einplanung von Prozessen</b>, so dass <math>N</math> Prozesse einen Stapel gemeinsam nutzen können [29, 30]</p> <ul style="list-style-type: none"> <li>▶ nicht immer ist es möglich, jeden Job durch einen eigenen Faden zu repräsentieren bzw. mit einem eigenen Stapel zu versehen             <ul style="list-style-type: none"> <li>▸ wenn die Jobanzahl zu hoch und/oder der Speicherplatz zu gering ist</li> </ul> </li> <li>▶ gemeinsame Nutzung desselben Stapels setzt voraus, dass kein Job bei Anforderung eines gemeinsamen Betriebsmittels blockiert             <ul style="list-style-type: none"> <li>▸ sonst droht die Überschreitung der Stapelbereiche anderer Jobs ⚠</li> </ul> </li> <li>▶ die <b>Jobs dürfen ihre Ausführung niemals selbst aussetzen</b>, sie dürfen jedoch von höher priorisierten Jobs verdrängt werden             <ul style="list-style-type: none"> <li>▸ oben auf dem Stapel läuft immer der Job mit der höchsten Priorität</li> <li>▸ die logische Konsequenz, wenn Selbstausschaltung ausgeschlossen ist</li> </ul> </li> </ul> <p>☞ <b>stapelbezogene Prioritätsbergrenzen</b> (engl. <i>stack-based priority ceiling</i>)</p>	
<small>soech WS 2005/06 E25 7-28</small>	<small>soech WS 2005/06 E25 7-28</small>

<small>7 Zugriffskontrolle 7.4 Prioritätsbergrenzen</small> <b>Stapelbezogene Grenzprioritäten</b> Regeln	<small>7 Zugriffskontrolle 7.4 Prioritätsbergrenzen</small> <b>Stapelbezogene Grenzprioritäten</b> Regeln
<ol style="list-style-type: none"> <li>1. Aktualisierung der <b>Grenzpriorität</b> (engl. <i>ceiling priority</i>, S. 7-22) <math>\hat{\pi}(t)</math> <ul style="list-style-type: none"> <li>▶ erfolgt mit jeder Vergabe/Freigabe von Betriebsmitteln</li> <li>▶ sind alle Betriebsmittel frei, gibt es keine Grenzpriorität</li> </ul> </li> </ol>	<ol style="list-style-type: none"> <li>1. Aktualisierung der <b>Grenzpriorität</b> (engl. <i>ceiling priority</i>, S. 7-22) <math>\hat{\pi}(t)</math> <ul style="list-style-type: none"> <li>▶ erfolgt mit jeder Vergabe/Freigabe von Betriebsmitteln</li> <li>▶ sind alle Betriebsmittel frei, gibt es keine Grenzpriorität</li> </ul> </li> <li>2. Einplanung und Einlastung von Jobs             <ul style="list-style-type: none"> <li>▶ nach erfolgter Auslösung muss ein Job ggf. solange warten, bis die ihm zugewiesene Priorität die Grenzpriorität übersteigt</li> <li>▶ Jobs werden jeweils entsprechend ihrer zugewiesenen Priorität und verdrängend ausgeführt</li> </ul> </li> </ol>
<small>soech WS 2005/06 E25 7-29</small>	<small>soech WS 2005/06 E25 7-28</small>

7 Zugriffskontrolle 7.4 Prioritätsbegrenzung Stapelbezogene Grenzprioritäten Regeln	7 Zugriffskontrolle 7.4 Prioritätsbegrenzung Stapelbezogene Grenzprioritäten Regeln
<ol style="list-style-type: none"> <li>1. Aktualisierung der <b>Grenzpriorität</b> (engl. <i>ceiling priority</i>, S. 7-22) <math>\hat{n}(t)</math> <ul style="list-style-type: none"> <li>▶ erfolgt mit jeder Vergabe/Freigabe von Betriebsmitteln</li> <li>▶ sind alle Betriebsmittel frei, gibt es keine Grenzpriorität</li> </ul> </li> <li>2. Einplanung und Einlastung von Jobs           <ul style="list-style-type: none"> <li>▶ nach erfolgter Auslösung muss ein Job ggf. solange warten, bis die ihm zugewiesene Priorität die Grenzpriorität übersteigt</li> <li>▶ Jobs werden jeweils entsprechend ihrer zugewiesenen Priorität und verdrängend ausgeführt</li> </ul> </li> <li>3. Zuteilung eines Betriebsmittels           <ul style="list-style-type: none"> <li>▶ erfolgt sofort, wann immer ein Job das Betriebsmittel anfordert</li> </ul> </li> </ol>	<ol style="list-style-type: none"> <li>1. Aktualisierung der <b>Grenzpriorität</b> (engl. <i>ceiling priority</i>, S. 7-22) <math>\hat{n}(t)</math> <ul style="list-style-type: none"> <li>▶ erfolgt mit jeder Vergabe/Freigabe von Betriebsmitteln</li> <li>▶ sind alle Betriebsmittel frei, gibt es keine Grenzpriorität</li> </ul> </li> <li>2. Einplanung und Einlastung von Jobs           <ul style="list-style-type: none"> <li>▶ nach erfolgter Auslösung muss ein Job ggf. solange warten, bis die ihm zugewiesene Priorität die Grenzpriorität übersteigt</li> <li>▶ Jobs werden jeweils entsprechend ihrer zugewiesenen Priorität und verdrängend ausgeführt</li> </ul> </li> <li>3. Zuteilung eines Betriebsmittels           <ul style="list-style-type: none"> <li>▶ erfolgt sofort, wann immer ein Job das Betriebsmittel anfordert</li> </ul> </li> </ol> <div style="border: 1px solid gray; padding: 5px; margin-top: 10px;"> <p>Jobs blockieren nicht, indem ihnen die Betriebsmittelzuteilung nach Ausführungsbeginn verweigert werden würde</p> <ul style="list-style-type: none"> <li>▶ im Gegensatz zum Grundmodell von Prioritätsbegrenzen (S. 7-23)</li> </ul> </div>
hoch WS 2005/06 E25 7-28	hoch WS 2005/06 E25 7-28

7 Zugriffskontrolle 7.4 Prioritätsbegrenzung Stapelbezogene Grenzprioritäten (Forts.) Implikationen	7 Zugriffskontrolle 7.4 Prioritätsbegrenzung Stapelbezogene Grenzprioritäten (Forts.) Implikationen
<p>Verklammungen nebenläufig ausgeführter Arbeitsaufträge sind durch eine <b>indirekte Methode zur Verklammungsvorbeugung</b><sup>18</sup> ausgeschlossen</p>	<p>Verklammungen nebenläufig ausgeführter Arbeitsaufträge sind durch eine <b>indirekte Methode zur Verklammungsvorbeugung</b><sup>18</sup> ausgeschlossen</p> <p>(a) beginnt ein Job seine Ausführung, sind alle von ihm im weiteren Verlauf benötigten Betriebsmittel frei</p> <ul style="list-style-type: none"> <li>▶ sonst wäre die Grenzpriorität größer oder gleich seiner Priorität</li> <li>▶ in dem Fall wäre jedoch die Einlastung des Jobs verzögert worden</li> </ul>
<p><sup>18</sup>zirkulares Warten wird vorgebeugt (siehe auch S. 7-24)</p> <p>hoch WS 2005/06 E25 7-30</p>	<p><sup>18</sup>zirkulares Warten wird vorgebeugt (siehe auch S. 7-24)</p> <p>hoch WS 2005/06 E25 7-30</p>

7 Zugriffskontrolle   7.4 Prioritätsbergemann Stapelbezogene Grenzprioritäten (Fortz.) Implikationen	7 Zugriffskontrolle   7.4 Prioritätsbergemann Stapelbezogene Grenzprioritäten (Fortz.) Implikationen
<p>Verklemmungen nebenläufig ausgeführter Arbeitsaufträge sind durch eine <b>indirekte Methode zur Verklemmungsvorbeugung</b><sup>18</sup> ausgeschlossen</p> <p>(a) beginnt ein Job seine Ausführung, sind alle von ihm im weiteren Verlauf benötigten Betriebsmittel frei</p> <ul style="list-style-type: none"> <li>▶ sonst wäre die Grenzpriorität größer oder gleich seiner Priorität</li> <li>▶ in dem Fall wäre jedoch die Einlastung des Jobs verzögert worden</li> </ul> <p>(b) bei Verdrängung eines Jobs sind alle von ihm benötigten Betriebsmittel (noch oder bereits wieder) frei</p> <ul style="list-style-type: none"> <li>▶ sonst hätte die Grenzpriorität eine Verdrängung unterbunden</li> <li>▶ der verdrängende Job wird also immer komplett durchlaufen können</li> </ul> <hr/> <p><sup>18</sup>zirkulares Warten wird vorgebeugt (siehe auch S. 7-24)</p>	<p>Verklemmungen nebenläufig ausgeführter Arbeitsaufträge sind durch eine <b>indirekte Methode zur Verklemmungsvorbeugung</b><sup>18</sup> ausgeschlossen</p> <p>(a) beginnt ein Job seine Ausführung, sind alle von ihm im weiteren Verlauf benötigten Betriebsmittel frei</p> <ul style="list-style-type: none"> <li>▶ sonst wäre die Grenzpriorität größer oder gleich seiner Priorität</li> <li>▶ in dem Fall wäre jedoch die Einlastung des Jobs verzögert worden</li> </ul> <p>(b) bei Verdrängung eines Jobs sind alle von ihm benötigten Betriebsmittel (noch oder bereits wieder) frei</p> <ul style="list-style-type: none"> <li>▶ sonst hätte die Grenzpriorität eine Verdrängung unterbunden</li> <li>▶ der verdrängende Job wird also immer komplett durchlaufen können</li> </ul> <p>(c) auf ein benötigtes Betriebsmittel kann direkt zugegriffen werden</p> <hr/> <p><sup>18</sup>zirkulares Warten wird vorgebeugt (siehe auch S. 7-24)</p>
<p>soech WS 2005/06 E25 7-30</p>	<p>soech WS 2005/06 E25 7-30</p>

7 Zugriffskontrolle   7.4 Prioritätsbergemann Grundmodell und Stapelorientierung Feste vs. dynamische Priorität (vergleiche S. 6-6)	7 Zugriffskontrolle   7.4 Prioritätsbergemann Grundmodell und Stapelorientierung Feste vs. dynamische Priorität (vergleiche S. 6-6)
	<p>feste Priorität <math>\mapsto</math> einfach, wegen à priori Wissen</p>
<p>soech WS 2005/06 E25 7-31</p>	<p>soech WS 2005/06 E25 7-31</p>

7 Zugriffskontrolle 7.4 Prioritätsübergewinn		7 Zugriffskontrolle 7.4 Prioritätsübergewinn			
<b>Grundmodell und Stapelorientierung</b> Feste vs. dynamische Priorität (vergleiche S. 6-6)		<b>Grundmodell und Stapelorientierung</b> Feste vs. dynamische Priorität (vergleiche S. 6-6)			
<p>feste Priorität <math>\mapsto</math> einfach, wegen à priori Wissen</p> <p>dynamische Priorität <math>\mapsto</math> Prioritäten periodischer Aufgaben ändern sich, während die von ihnen beanspruchten Betriebsmittel konstant bleiben:</p> <ul style="list-style-type: none"> <li>☞ Grenzprioritäten der Betriebsmittel ändern sich mit der Zeit</li> </ul>		<p>feste Priorität <math>\mapsto</math> einfach, wegen à priori Wissen</p> <p>dynamische Priorität <math>\mapsto</math> Prioritäten periodischer Aufgaben ändern sich, während die von ihnen beanspruchten Betriebsmittel konstant bleiben:</p> <ul style="list-style-type: none"> <li>☞ Grenzprioritäten der Betriebsmittel ändern sich mit der Zeit</li> <li>▶ Aktualisierung der Grenzprioritäten bei jeder Auslösung eines Jobs</li> </ul>			
soech	WS 2005/06 E25	7-31	soech	WS 2005/06 E25	7-31

7 Zugriffskontrolle 7.4 Prioritätsübergewinn		7 Zugriffskontrolle 7.4 Prioritätsübergewinn			
<b>Grundmodell und Stapelorientierung</b> Feste vs. dynamische Priorität (vergleiche S. 6-6)		<b>Grundmodell und Stapelorientierung</b> Feste vs. dynamische Priorität (vergleiche S. 6-6)			
<p>feste Priorität <math>\mapsto</math> einfach, wegen à priori Wissen</p> <p>dynamische Priorität <math>\mapsto</math> Prioritäten periodischer Aufgaben ändern sich, während die von ihnen beanspruchten Betriebsmittel konstant bleiben:</p> <ul style="list-style-type: none"> <li>☞ Grenzprioritäten der Betriebsmittel ändern sich mit der Zeit</li> <li>▶ Aktualisierung der Grenzprioritäten bei jeder Auslösung eines Jobs               <ol style="list-style-type: none"> <li>1. dem ausgelösten Job zum Ereigniszeitpunkt eine Priorität zuweisen                   <ul style="list-style-type: none"> <li>▶ relativ zu den anderen bereits eingeplanten/laufbereiten Jobs</li> <li>▶ (dynamische) prioritätsorientierte Einplanung je nach Verfahren</li> </ul> </li> </ol> </li> </ul>		<p>feste Priorität <math>\mapsto</math> einfach, wegen à priori Wissen</p> <p>dynamische Priorität <math>\mapsto</math> Prioritäten periodischer Aufgaben ändern sich, während die von ihnen beanspruchten Betriebsmittel konstant bleiben:</p> <ul style="list-style-type: none"> <li>☞ Grenzprioritäten der Betriebsmittel ändern sich mit der Zeit</li> <li>▶ Aktualisierung der Grenzprioritäten bei jeder Auslösung eines Jobs               <ol style="list-style-type: none"> <li>1. dem ausgelösten Job zum Ereigniszeitpunkt eine Priorität zuweisen                   <ul style="list-style-type: none"> <li>▶ relativ zu den anderen bereits eingeplanten/laufbereiten Jobs</li> <li>▶ (dynamische) prioritätsorientierte Einplanung je nach Verfahren</li> </ul> </li> <li>2. Grenzprioritäten aller Betriebsmittel aktualisieren                   <ul style="list-style-type: none"> <li>▶ auf Basis der neuen Taskprioritäten</li> </ul> </li> </ol> </li> </ul>			
soech	WS 2005/06 E25	7-31	soech	WS 2005/06 E25	7-31

7 Zugriffskontrolle 7.4 Prioritätsübergewinn

## Grundmodell und Stapelorientierung

Feste vs. dynamische Priorität (vergleiche S. 6-6)

**feste Priorität**  $\mapsto$  einfach, wegen à priori Wissen

**dynamische Priorität**  $\mapsto$  Prioritäten periodischer Aufgaben ändern sich, während die von ihnen beanspruchten Betriebsmittel konstant bleiben:

- ☞ **Grenzprioritäten der Betriebsmittel ändern sich mit der Zeit**
  - ▶ Aktualisierung der Grenzprioritäten bei jeder Auslösung eines Jobs
    1. dem ausgelösten Job zum Ereigniszeitpunkt eine Priorität zuweisen
      - ▶ relativ zu den anderen bereits eingeplanten/laufbereiten Jobs
      - ▶ (dynamische) prioritätsorientierte Einplanung je nach Verfahren
    2. Grenzprioritäten aller Betriebsmittel aktualisieren
      - ▶ auf Basis der neuen Taskprioritäten
    3. Grenzpriorität des Systems Betriebsmittel aktualisieren
      - ▶ auf Basis der neuen Grenzprioritäten der Betriebsmittel

soech WS 2005/06 E25 7-31

7 Zugriffskontrolle 7.4 Prioritätsübergewinn

## Grundmodell und Stapelorientierung

Feste vs. dynamische Priorität (vergleiche S. 6-6)

**feste Priorität**  $\mapsto$  einfach, wegen à priori Wissen

**dynamische Priorität**  $\mapsto$  Prioritäten periodischer Aufgaben ändern sich, während die von ihnen beanspruchten Betriebsmittel konstant bleiben:

- ☞ **Grenzprioritäten der Betriebsmittel ändern sich mit der Zeit**
  - ▶ Aktualisierung der Grenzprioritäten bei jeder Auslösung eines Jobs
    1. dem ausgelösten Job zum Ereigniszeitpunkt eine Priorität zuweisen
      - ▶ relativ zu den anderen bereits eingeplanten/laufbereiten Jobs
      - ▶ (dynamische) prioritätsorientierte Einplanung je nach Verfahren
    2. Grenzprioritäten aller Betriebsmittel aktualisieren
      - ▶ auf Basis der neuen Taskprioritäten
    3. Grenzpriorität des Systems Betriebsmittel aktualisieren
      - ▶ auf Basis der neuen Grenzprioritäten der Betriebsmittel
  - ▶ bei auf Jobebene statische wie auch dynamische Prioritäten (S. 6-7)

soech WS 2005/06 E25 7-31

7 Zugriffskontrolle 7.5 Zusammenfassung

## Resümee

**Konkurrenz und Koordination nebenläufiger Aktivitäten**

- ▶ Nebenläufigkeit, Kausalität, Kausalordnung
- ▶ Konfliktsituationen  $\leadsto$  **synchronisieren ohne Prioritätsumkehr**

soech WS 2005/06 E25 7-32

7 Zugriffskontrolle 7.5 Zusammenfassung

## Resümee

**Konkurrenz und Koordination nebenläufiger Aktivitäten**

- ▶ Nebenläufigkeit, Kausalität, Kausalordnung
- ▶ Konfliktsituationen  $\leadsto$  **synchronisieren ohne Prioritätsumkehr**

**Verdrängungssteuerung**  $\mapsto$  verdrängungsfreie kritische Abschnitte

- ▶ benötigt kein à priori Wissen; Verklemmungsvorbeugung
- ▶ pragmatisch/effektiv, beeinträchtigt unabhängige Jobs

soech WS 2005/06 E25 7-32

## Resümee

## Konkurrenz und Koordination nebenläufiger Aktivitäten

- ▶ Nebenläufigkeit, Kausalität, Kausalordnung
- ▶ Konfliktsituationen  $\leadsto$  **synchronisieren ohne Prioritätsumkehr**

Verdrängungssteuerung  $\mapsto$  verdrängungsfreie kritische Abschnitte

- ▶ benötigt kein *a priori* Wissen; Verklemmungsvorbeugung
- ▶ pragmatisch/effektiv, beeinträchtigt unabhängige Jobs

Prioritätsvererbung  $\mapsto$  Priorität zeitweise erhöhen

- ▶ benötigt kein *a priori* Wissen
- ▶ direkte Blockierung, Blockierung durch Vererbung; transitiv

## Resümee

## Konkurrenz und Koordination nebenläufiger Aktivitäten

- ▶ Nebenläufigkeit, Kausalität, Kausalordnung
- ▶ Konfliktsituationen  $\leadsto$  **synchronisieren ohne Prioritätsumkehr**

Verdrängungssteuerung  $\mapsto$  verdrängungsfreie kritische Abschnitte

- ▶ benötigt kein *a priori* Wissen; Verklemmungsvorbeugung
- ▶ pragmatisch/effektiv, beeinträchtigt unabhängige Jobs

Prioritätsvererbung  $\mapsto$  Priorität zeitweise erhöhen

- ▶ benötigt kein *a priori* Wissen
- ▶ direkte Blockierung, Blockierung durch Vererbung; transitiv

Prioritätsbergrenzen  $\mapsto$  Priorität zeitweise deckeln

- ▶ benötigt *a priori* Wissen; Verklemmungsvorbeugung
- ▶ Grundmodell vs. (einfachere) stapelorientierte Variante