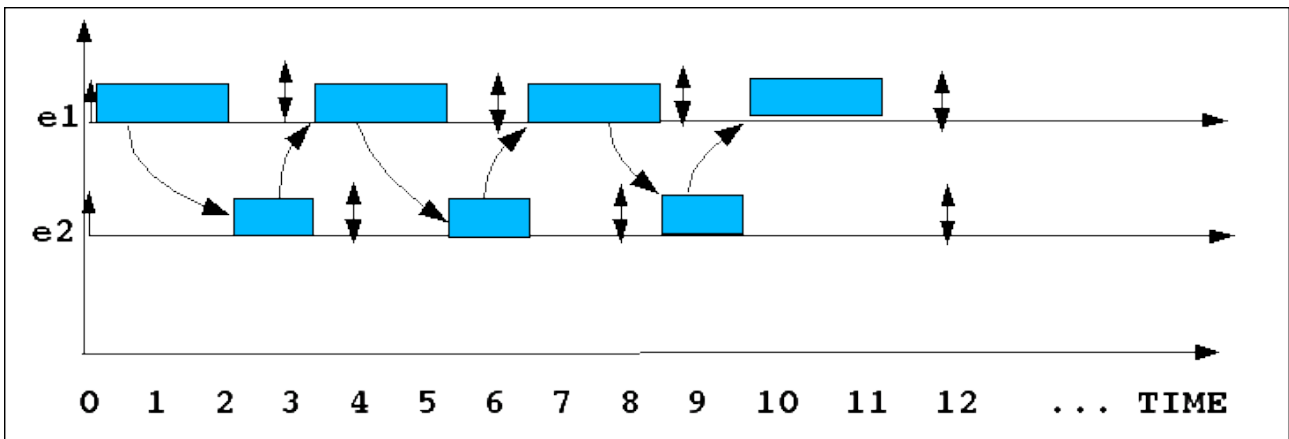
	Fachbereich 2 Studiengang: <i>Ing.Informatik</i>
	Prozessrechner – Übungsaufgabe Nr.2
	Letzte Änderung: 21.April 2004 <span style="float: right;">Seite 1//4</span>

## Das Verhalten eines Realzeitsystems planen


### Beschreibung

Im Rahmen der Vorlesung geht es jetzt zunehmend darum, Ereignisse, die als Input für Realzeitsysteme auftreten können, durch diese Systeme zu bearbeiten. Dazu ist es wichtig, vorab zu klären, ob diese Ereignisse bei den gegebenen Ressourcen so geplant werden können, dass Sie ihre zeitlichen Vorgaben ('Deadlines') einhalten können. Dazu muss ein Plan erstellt werden.



Obige Grafik zeigt ein Beispiel: Zwei *Ereignisse*  $e_1$  und  $e_2$  treten regelmässig (*periodisch*) auf. Jedes Ereignis hat eine eigene Zeitlinie. Alle Ereignisse beginnen bei der Zeitmarke 0. Ereignis  $e_1$  hat die *Periode*  $T=3$ , Ereignis  $e_2$  hat die *Periode*  $T=4$ ; dies bedeutet, dass Ereignis  $e_1$  alle 3 Zeiteinheiten (ab 0) auftritt und  $e_2$  alle 4 Zeiteinheiten (ab 0). Das *Auftreten* (*occurrence*) eines Ereignisses auf der Zeitachse wird durch einen nach oben zeigenden Pfeil angezeigt. In diesem Beispiel wird zusätzlich angenommen, dass bei beiden Ereignissen die *relative Deadline*  $D$  mit der Periode zusammenfällt, d.h.  $D = T$ . Dies bedeutet, die zu einem Ereignis gehörende *Task* muss spätestens bis zur *Deadline*  $D$  abgearbeitet sein, also, wegen  $D = T$ , bis spätestens zum Auftreten des nächsten Ereignisses. *Deadlines* werden angezeigt durch einen nach unten zeigenden Pfeil; fallen *Deadline* und *Periode* zusammen, dann benutzt man einen Doppelpfeil. Ereignisse treten 'punktuell' auf, zu einem bestimmten Zeitpunkt  $t$  auf der Zeitachse. Wenn ein Ereignis  $e_i$  als Input eines Systems  $S$  auftritt, dann ist normalerweise festgelegt, welche *Aufgabe* (*Task*)  $t$  aufgrund des Auftretens des Ereignisses  $e_i$  ausgeführt werden soll. Nehmen wir vorläufig an, jedem Ereignis wird ein eigener *Task* zugeordnet, also es soll gelten:  $task(e_i) = t_i$ . Ferner gilt, dass jeder *Task*  $t_i$  eine typische *Ausführungszeit* (*execution time*)  $C_i$  besitzt.

Setzt man nun voraus, dass man nur *ein einziges* System  $S$  hat und dieses System zur gleichen

	Fachbereich 2 Studiengang: <i>Ing.Informatik</i>
	Prozessrechner – Übungsaufgabe Nr.2
	Letzte Änderung: 21.April 2004 <span style="float: right;">Seite 2 / 4</span>

Zeit nur *einen einzigen Task* ausführen kann, dann gibt es ein Problem, sobald mehr als ein Ereignis gleichzeitig oder 'ah beieinander' auftreten. Die zugehörigen Tasks können nicht gleichzeitig ausgeführt werden.

Im obigen Beispiel treten die beiden Ereignisse e1 und e2 bei Zeitpunkt 0 gleichzeitig auf. Zu diesem Zeitpunkt gilt:

$t_1(0, 2.5, 3, 3)$

$t_2(0, 1.7, 4, 4)$


Dabei wird das Schema benutzt:

$t(r_0, C, D, T)$  mit  $0 \leq C \leq D \leq T$

(mit der Bedeutung  $r_0$  := erstmalige Auftretenszeit des Ereignisses, das Task  $t$  hervorruft;  $D$  := Ausführungszeit;  $D$  := Deadline;  $T$  := Periode).

Zum Zeitpunkt  $r=0$  muss also entschieden werden, welche der beiden Tasks soll nun zuerst ausgeführt werden? Wir nehmen für diese Übung an, dass zur gleichen Zeit nur ein Task ausgeführt werden kann und der andere warten muss, bis ein schon laufender Task beendet ist. Um die Frage zu entscheiden, benötigt man eine *Entscheidungsregel*. Hier ist ein Beispiel einer Entscheidungsregel in Form eines *Entscheidungsalgorithmus (Planungsalgorithmus, Scheduler)*:

<b>Scheduler SCH1</b>
<p>Gegeben:            WARTENDE TASKS <math>W^*</math></p> <p>Wenn zu einem Zeitpunkt <math>T</math> die Menge der NEUEREIGNISSE <math>E^*</math> auftritt (mit <math>E^* \neq \text{leer}</math>), dann wird nach Beendigung eines gerade laufenden Tasks <math>t</math> aus <math>W^*</math> die Menge <math>E^*</math> zur Menge <math>W^*</math> hinzugefügt und für jedes Element aus <math>W^*</math> wird neu berechnet, wie lange die verbleibende Zeit <math>K</math> bis zur relativen Deadline <math>D</math> ist. Alle Elemente werden entsprechend der Grösse <math>K</math> geordnet. Das Element <math>t</math> aus <math>W^*</math>, das das kleinste <math>K</math> ist, wird als nächstes ausgeführt.</p> <p>Gibt es mehr als ein Element mit einem bestimmten Wert für <math>K</math>, dann ordne diese Menge beliebig.</p> <p>Tritt keine Menge <math>E^*</math> hinzu, dann wird nach Beendigung eines Tasks <math>t</math> aus <math>W^*</math> der nächste Task <math>t'</math> aus <math>W^*</math> ausgeführt.</p>


	Fachbereich 2 Studiengang: <i>Ing.Informatik</i>
	Prozessrechner – Übungsaufgabe Nr.2
	Letzte Änderung: 21.April 2004 <span style="float: right;">Seite 3//4</span>

<b>ZEI T</b>	<b>E*</b>	<b>W*</b>	<b>Nach Berechnung</b>
0	{t1,t2}	{t1, t2}	<t1,t2>
3	{t1}	{t1}	<t1>
4	{t2}	{t2}	<t2>
6	{t1}	{t1}	<t1>
8	{t2}	{t2}	<t2>
9	{t1}	{t1}	<t1>
...	Wiederholung	...	...

Aus dem Beispiel kann man ersehen, dass sich die Tasks t1 und t2 planen lassen, ohne dass die deadlines irgendwo überschritten werden.

### Teilaufgaben

- (Max.Pkt: 2) Überlegen Sie sich eine realistische Anwendungssituation, in der mindestens drei verschiedene periodische Ereignisse {e1, e2, e3, ...} auftreten, auf die ein System mit unterschiedlichen Tasks {t1, t2, t3, ...} antworten muss. Nehmen Sie auch an, dass ihr System nur einen Task gleichzeitig ausführen kann, ein laufender Task nicht unterbrochen werden darf und die Deadline D gleich der Periode T ist. Beschreiben sie die Anwendungssituation.
- (Max.Pkt: 2) Zeichnen sie ein Zeitdiagramm wie im vorausgehenden Beispiel: pro Task eine Zeitlinie. Tragen Sie die Periode und die Deadlines für jedes Ereignis ein. Achten Sie darauf, dass ihr Zeitfenster lange genug ist. Ordnen sie die Tasks entsprechend dem Scheduler SCH1 an.
- (Max.Pkt: 4) Schreiben Sie ein kleines Programm (C oder C++), in dem Sie den Scheduler SCH1 simulieren können. Eingabe für den Algorithmus sind Ereignisse und zugehörige Tasks. Ausgabe ist eine Liste von Zeitpunkten und die zu jedem Zeitpunkt auszuführenden Tasks, geordnet nach K.

	Fachbereich 2 Studiengang: <i>Ing.Informatik</i>
	Prozessrechner – Übungsaufgabe Nr.2
	Letzte Änderung: 21.April 2004 <span style="float: right;">Seite 4 / 4</span>

## Abgabe

Die Übungsaufgabe gilt als erfüllt, wenn folgende Bedingungen erfüllt sind:

1. In der Übung liegen alle Dokumente ausgedruckt vor (vergewissern sie sich rechtzeitig, wie und wo Sie ausdrucken können)
2. Sie haben ihr Programm auf einem elektronischen Datenträger und können die Dateien von diesen Datenträgern auf ihren Zielrechner laden (Diskette oder CD-ROM oder per sftp) (Vergewissern Sie sich rechtzeitig, wie Sie Daten entsprechend präparieren)
3. Sie können ihren Beitrag auf dem Zielrechner und per Beamer vorführen (Bedenken sie, dass unterschiedliche Versionen von Compilern und Präsentationsprogrammen gibt. Vergewissern sie sich, dass sich ihr programm und ihr Beitrag auf dem Zielsystem so verhält, wie auf ihrem Entwicklungssystem).
4. Alle an der Erstellung der Übung Beteiligten sind anwesend und erläutern den Beitrag (Wer nicht anwesend ist bekommt keine Punkte).