Klausursammlung Realzeitsysteme

Jürgen Quade

Klausursammlung Realzeitsysteme
von Jürgen Quade

V7.0, 30. Oktober 2013

Versionsgeschichte
Version $Revision: 1.2 $ $Date: 2013/10/30 09:56:41 $ Geändert durch: $Author: quade $

Inhaltsverzeichnis

Kapitel 1. Klausur Realzeitsysteme SS2013

1.1. Hinweise zum Prüfungsablauf

Tag der Prüfung	17.09.2013
Beginn	8:30 Uhr
Dauer	120 Minuten
Ort	Audimax

Die Prüfung besteht aus einem Fragenteil ohne Unterlagen und einem Aufgabenteil mit Unterlagen.

Die Bearbeitungszeit für den Fragenteil, zu dem keinerlei Hilfsmittel und Unterlagen zugelassen sind, beträgt 30 Minuten. Beantworten Sie die Fragen auf einem eigenen Prüfungsbogen. Sobald Sie mit der Bearbeitung des ersten Prüfungsteiles fertig sind, können Sie mit dem zweiten Teil fortfahren. Unterlagen dürfen allerdings erst verwendet werden, nachdem sämtliche Lösungsblätter des ersten Teiles eingesammelt worden sind.

Der zweite Teil der Prüfung besteht aus insgesamt 3 Aufgaben. Die einzelnen Aufgaben und größtenteils auch die einzelnen Teilaufgaben sind unabhängig voneinander lösbar. Verwenden Sie bitte für jede Aufgabe ein eigenes Lösungsblatt. Die für diesen Teil zur Verfügung stehende Bearbeitungszeit beträgt ingesamt 90 Minuten.

Bitte legen Sie am Ende der Prüfungszeit alle Lösungblätter ineinander.

Die über den Aufgaben stehende Punktzahl ist vorläufig und unverbindlich.

Achtung

Versehen Sie unbedingt alle Prüfungsbögen und Papiere, die Sie mit abgeben, mit Ihrem Namen und Ihrer Matrikelnummer! Geben Sie auf jeden Fall alle Prüfungsbögen ab, auch wenn Sie eine Aufgabe nicht bearbeitet haben.

Viel Erfolg!

1.2. Ohne Unterlagen

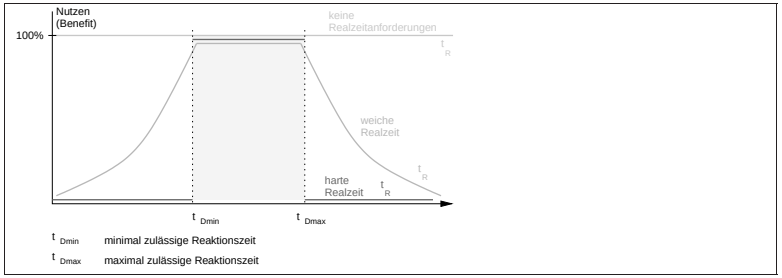
1. Definieren Sie den Begriff »Realzeitsystem«. (1P)

Ein Realzeitsystem muss neben den funktionalen Anforderungen auch noch zeitlichen Anforderungen genügen.

2. Wie lauten die beiden Realzeitbedingungen (genaue Angabe)? (2P)

1. RZB	$\rho_{ges} = \sum_{i=1}^{n} \frac{t_{Emax,i}}{t_{Pmin,i}} \leq c; \quad mit\ c = Anzahl\ Rechnerkerne$
2. RZB	Für alle Rechenzeitanforderungen *i* muss gelten: $t_{Dmin,i} \leq t_{Rmin,i} \leq t_{Rmax,i} \leq t_{Dmax,i}$

3. Erläutern Sie anhand der Nutzen-Funktion mithilfe einer **genauen** Skizze den Unterschied zwischen harter und weicher Realzeit! (2P)



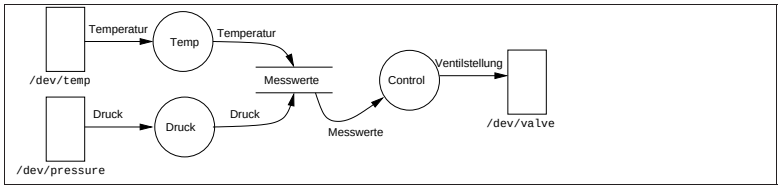
4. Bitoperationen
   a. Mit welchem logischen Operator können Sie einzelne Bits eines Bitfeldes löschen? (1,2=3P)

Mit einer bitweisen UND-Verknüpfung („&”).

   b. Skizzieren Sie die Codezeilen, die die niederwertigsten drei Bits (0, 1 und 2) in dem 16-Bit breiten Register `gpfsel0` auf den Wert `001` setzen, den Zustand der übrigen Bits aber unverändert lassen.

```
gpfsel0 = gpfsel0 & 0xfff8; // Bit 0,1 und 2 löschen
gpfsel0 = gpfsel0 | 0x0001; // Bit 0 setzen
```

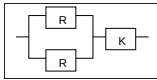
5. Das Taskgebilde einer Steuerung besteht aus drei Tasks. Die Task »Temp« liest über die Gerätedatei »/dev/temp« die Temperatur ein und legt sie in den gemeinsamen Speicher »messwerte«. Die Task »Druck« liest über die Gerätedatei »/dev/pressure« den Druck ein und legt ihn ebenfalls in dem gemeinsamen Speicher »messwerte« ab. Die Task »control« liest die Messwerte aus dem gemeinsamen Speicher aus und berechnet den Ausgabewert für das Ventil »/dev/valve«. Zeichnen Sie das Datenflussdiagramm (DFD) des beschriebenen Taskgebildes. Tragen Sie auch die Kontrollflüsse ein. Beschriften Sie Verarbeitungseinheiten, Datenflüsse, Kontrollflüsse, Datenspeicher, Datenquellen und -senken! (6P)



6. Skizzieren Sie den Code einer **Funktion** `int copy_file(char *infilename, char *outfilename)`, die die Eingabedatei »infilename« in die Datei »outfilename« kopiert. Die Funktion gibt »-1« zurück, falls es ein Problem beim Öffnen der Dateien gibt, ansonsten »0«. (8P)

```
int copy_file(char *infilename, char *outfilename)
{
    int fdin, fdout;
    char buffer[4096];
    int count;

    fdin = open( infilename, O_RDONLY );
    fdout= open( outfilename,O_WRONLY|O_CREAT );
    if (fdin<0 || fdout<0) {
        printf("Fehler beim Oeffnen\n");
        return -1;
    }
    do {
        count = read( fdin, buffer, sizeof(buffer) );
        write( fdout, buffer, count );
    } while (count>0);
    return 0;
}
```

7. Die Verfügbarkeit einer Steuerung soll durch den Einsatz von zwei Rechnern mit gleicher Verfügbarkeit p_r=99.2% vergrößert werden. Die benötigte Rechnerkopplung hat die Verfügbarkeit p_k=99.4%. Das System ist verfügbar, wenn mindestens ein Rechner und die Rechnerkopplung verfügbar ist. (1,2,5=8P)
   a. Zeichnen Sie das Verfügbarkeitsersatzschaltbild der Konstellation.



   b. Berechnen Sie die Verfügbarkeit p_{ges} der Konstellation.

$p_{sys} = p_K) (1-(1-p_k)_2) = 0.994 * (1-(0.0008)_2) = 0.99394$

   c. Geben Sie allgemein an, wie groß $p_k(p_r)$ sein muss, damit die Systemverfügbarkeit p_{sys} gleich oder höher als die Verfügbarkeit eines Einzelrechners ist.

$p_K (1 - (1-p_K)_2) \geq p_R$
$p_R \geq p_R / (1-(1-p_R)_2$

$p_R \geq 1/(2-p_R)$

Mit Unterlagen

Name: _____

Matrikelnr.: _____

1.2.1. Aufgabe 1

Die Anwendungssoftware einer Steuerung besteht aus fünf Jobs:

RZ-Anf.	$t_{Pmin,i}$	$t_{Dmin,i}$	$t_{Dmax,i}$	$t_{Ph,i}$	$t_{Emin,i}$	$t_{Emax,i}$	$\rho_{max,i}$
A	45 ms	0 ms	35 ms	0 ms	12 ms	15 ms	
B	45 ms	0 ms	35 ms	0 ms	12 ms	15 ms	
C	90 ms	0 ms	80 ms	0 ms	60 ms	60 ms	0.667
D	90 ms	0 ms	90 ms	0 ms	60 ms	60 ms	0.667
E	180 ms	0 ms	150 ms	0 ms	80 ms	150 ms	

1. Leiten Sie die folgenden Kenndaten her (1,1,1,1=4P)
   a. $\rho_{max,A}$ und $\rho_{max,B}$

   $\rho_{max,A}$=15ms/45ms =0,333
   $\rho_{max,B}$=15ms/45ms =0,333

   b. $\rho_{max,E}$

   $\rho_{max,E}$=150ms/180ms =0,833

   c. $\rho_{max,ges}$ (maximale Gesamtauslastung aller fünf Tasks)

   $\rho_{max,ges}$= $\rho_{max,A}$ + $\rho_{max,B}$ + $\rho_{max,C}$ + $\rho_{max,D}$ + $\rho_{max,E}$=2,8333

   d. Ist schritthaltende Verarbeitung *prinzipiell* möglich?

   Nein, schritthaltende Verarbeitung ist nicht möglich.

*Für die nachfolgenden Teilaufgaben wird **eine** CPU eingesetzt, für die der Hersteller eine mindestens **dreifache Verarbeitungsleistung** garantiert.*

2. Tragen Sie in den Tabellenkopf der nachfolgenden Tabelle die Zeitarten ein, die sich durch die stärkere CPU ändern. Vervollständigen Sie danach die Tabelle mit den zugehörigen, neuen Zeiten. (4P)

Task	$t_{Emin,i}$	$t_{Emax,i}$
A	4 ms	5 ms
B	4 ms	5 ms

Task	$t_{Emin,i}$	$t_{Emax,i}$
C	20 ms	20 ms
D	20 ms	20 ms
E	80/3 ms	50 ms

3. Welche Auslastung ρ_{ges,cpu_neu} ergibt sich mit den neuen Zeiten? (1P)

   ρ_{ges,cpu_neu}= ρ_{ges}/3 = 0.9444

4. Das verwendete Betriebssystem stellt genau drei Prioritätsebenen zur Verfügung (Prioritäten 1=hoch, 2=mittel, 3=niedrig). Verteilen Sie sinnvoll die Prioritäten. (1P)

   A,B=1 (hoch), C,D=2 (mittel), E=3 (niedrig)

5. Wie lautet allgemein die Bedingung für den hinreichenden Schedulingtest (Utilization u) für prioritätengesteuertes Scheduling? Ist demzufolge der notwendige Schedulingtest durchzuführen? (2P)

   $$u = \sum_{j=1}^{n} \frac{t_{Emax,j}}{\min\left(t_{Dmax,j},\, t_{Pmin,j}\right)} \leq n(2^{\frac{1}{n}} - 1)$$

6. Bis zu welcher Auslastungsgrenze (in Prozent) ist bei prioritätengesteuertem Scheduling und **fünf Tasks** eine schritthaltende Verarbeitung immer gewährleistet? (1P)

   u $\leq$ 0.743 (74.3 Prozent)

7. Realzeitnachweis bei Einsatz eines prioritätengesteuerten Schedulings und der leistungsstarken CPU. (1,1,1,2,1,3,5=14P)
   a. Geben Sie die Rechenzeitanforderungsfunktion $t_{C,1}(t)$ für Jobs mit Priorität 1 an.

   $$t_{c,1}(t) = 2 \cdot \lceil \frac{t}{45ms} \rceil \cdot 5ms$$

   b. Berechnen Sie die maximale Reaktionszeit $t_{Rmax,1}$ für die zugehörigen Jobs.

   $$t_c^{(1)} = 10ms$$
   $$t_c^{(2)} = \lceil \frac{10ms}{45ms} \rceil \cdot 10ms = 10ms$$
   $$t_{Rmax,1} = 10ms$$

   c. Geben Sie die Rechenzeitanforderungsfunktion $t_{C,2}(t)$ für Jobs mit Priorität 2 an.

   $$t_{c,2}(t) = 2 \cdot \lceil \frac{t}{45ms} \rceil \cdot 5ms + 2 \cdot \lceil \frac{t}{90ms} \rceil \cdot 20ms = \lceil \frac{t}{45ms} \rceil \cdot 10ms + \lceil \frac{t}{90ms} \rceil \cdot 40ms$$

   d. Berechnen Sie die maximale Reaktionszeit $t_{Rmax,2}$ für Jobs der Priorität 2.

$$t_c^{(1)} = 50ms$$

$$t_c^{(2)} = \lceil\frac{50ms}{45ms}\rceil \cdot 10ms + \lceil\frac{50ms}{90ms}\rceil \cdot 40ms = 60ms$$

$$t_c^{(3)} = \lceil\frac{60ms}{45ms}\rceil \cdot 10ms + \lceil\frac{60ms}{90ms}\rceil \cdot 40ms = 60ms$$

$$t_{Rmax,2} = 60ms$$

e. Geben Sie die Rechenzeitanforderungsfunktion $t_{C,3}(t)$ für Jobs mit Priorität 3 an.

$$t_{c,3}(t) = \lceil\frac{t}{45ms}\rceil \cdot 10ms + \lceil\frac{t}{90ms}\rceil \cdot 40ms + \lceil\frac{t}{180ms}\rceil \cdot 50ms$$

f. Berechnen Sie die maximale Reaktionszeit $t_{Rmax,3}$ für Jobs der Priorität 3.

$$t_c^{(1)} = 100ms$$

$$t_c^{(2)} = \lceil\frac{100ms}{45ms}\rceil \cdot 10ms + \lceil\frac{100ms}{90ms}\rceil \cdot 40ms + \lceil\frac{100ms}{180ms}\rceil \cdot 50ms = 160ms$$

$$t_c^{(3)} = \lceil\frac{160ms}{45ms}\rceil \cdot 10ms + \lceil\frac{160ms}{90ms}\rceil \cdot 40ms + \lceil\frac{160ms}{180ms}\rceil \cdot 50ms = 170ms$$

$$t_c^{(4)} = \lceil\frac{170ms}{45ms}\rceil \cdot 10ms + \lceil\frac{170ms}{90ms}\rceil \cdot 40ms + \lceil\frac{170ms}{180ms}\rceil \cdot 50ms = 170ms$$

$$t_{Rmax,3} = 170ms$$

g. Zeigen Sie anhand der 2. Realzeitbedingung (Rechtzeitigkeitsbedingung), ob schritthaltende Verarbeitung für sämtliche Jobs möglich ist oder nicht.

A: $0ms \leq 4ms \leq 10ms \leq 35ms$ Ja, diese Bedingung ist erfüllt.

B: $0ms \leq 4ms \leq 10ms \leq 35ms$ Ja, diese Bedingung ist erfüllt.

C: $0ms \leq 20ms \leq 60ms \leq 80ms$ Ja, diese Bedingung ist erfüllt.

D: $0ms \leq 20ms \leq 60ms \leq 90ms$ Ja, diese Bedingung ist erfüllt.

E: $0ms \leq 26.67ms \leq 170ms \leq 150ms$ Nein, diese Bedingung ist nicht erfüllt.

Schritthaltende Verarbeitung ist bei Einsatz eines prioritätengesteuerten Schedulings möglich!

8. Echtzeitnachweis bei Einsatz eines Deadline-Schedulings. (4,3,6=13P)

a. Geben Sie die Gesamt-Rechenzeitanforderungsfunktion $t_{C,ges}(I)$ an.

$$t_{C,ges}(I) = \sum \lfloor\frac{I + t_{Pmin,i} - t_{Dmax,i} - t_{PH,i}}{t_{P,i}}\rfloor \cdot t_{Emax,i}$$

$$t_{C,ges}(I) = 2 \cdot \lfloor\frac{I - 35ms + 45ms - 0ms}{45ms}\rfloor \cdot 5ms + \lfloor\frac{I - 80ms + 90ms}{90ms}\rfloor \cdot 20ms + \lfloor\frac{I - 90ms + 90ms}{90ms}\rfloor \cdot 2$$

$$t_{C,ges}(I) = \lfloor\frac{I + 10ms}{45ms}\rfloor \cdot 10ms + \lfloor\frac{I + 10ms}{90ms}\rfloor \cdot 20ms + \lfloor\frac{Ims}{90ms}\rfloor \cdot 20ms + \lfloor\frac{I + 30ms}{180ms}\rfloor \cdot 50ms$$

b. Für welche I müssen Sie $t_{C,ges}(I)$ konkret untersuchen? Geben Sie den Bereich an, in dem I zu untersuchen ist und zusätzlich die zu untersuchenden Intervalle in diesem Bereich.

I ist zu untersuchen im Bereich von $0 < I < kgV(45ms, 90ms, 180ms) = 180ms$; (da $t_S = 0$)
$I_{A,B}$: 35ms, 80ms, 125ms, 170ms
I_C: 80ms, 170ms
I_D: 90ms
I_E: 150ms
I: 35ms, 80ms, 90ms, 125ms, 150ms, 170ms

c. Führen Sie den Realzeitnachweis durch. Ist schritthaltende Verarbeitung möglich?

$$t_{C,ges}(35ms) = \lfloor\frac{45ms}{45ms}\rfloor \cdot 10ms + \lfloor\frac{45ms}{90ms}\rfloor \cdot 20ms + \lfloor\frac{35ms}{90ms}\rfloor \cdot 20ms + \lfloor\frac{65ms}{180ms}\rfloor \cdot 50ms = 10ms$$

$$t_{C,ges}(80ms) = \lfloor\frac{90ms}{45ms}\rfloor \cdot 10ms + \lfloor\frac{90ms}{90ms}\rfloor \cdot 20ms + \lfloor\frac{80ms}{90ms}\rfloor \cdot 20ms + \lfloor\frac{110ms}{180ms}\rfloor \cdot 50ms = 40ms$$

$$t_{C,ges}(90ms) = \lfloor\frac{100ms}{45ms}\rfloor \cdot 10ms + \lfloor\frac{100ms}{90ms}\rfloor \cdot 20ms + \lfloor\frac{90ms}{90ms}\rfloor \cdot 20ms + \lfloor\frac{120ms}{180ms}\rfloor \cdot 50ms = 60ms$$

$$t_{C,ges}(125ms) = \lfloor\frac{135ms}{45ms}\rfloor \cdot 10ms + \lfloor\frac{135ms}{90ms}\rfloor \cdot 20ms + \lfloor\frac{125ms}{90ms}\rfloor \cdot 20ms + \lfloor\frac{155ms}{180ms}\rfloor \cdot 50ms = 70ms$$

$$t_{C,ges}(150ms) = \lfloor\frac{160ms}{45ms}\rfloor \cdot 10ms + \lfloor\frac{160ms}{90ms}\rfloor \cdot 20ms + \lfloor\frac{150ms}{90ms}\rfloor \cdot 20ms + \lfloor\frac{180ms}{180ms}\rfloor \cdot 50ms = 120ms$$

$$t_{C,ges}(170ms) = \lfloor\frac{180ms}{45ms}\rfloor \cdot 10ms + \lfloor\frac{180ms}{90ms}\rfloor \cdot 20ms + \lfloor\frac{170ms}{90ms}\rfloor \cdot 20ms + \lfloor\frac{210ms}{180ms}\rfloor \cdot 50ms = 150ms$$

Da die Bedingung $t_{C,ges}(I) \leq I$ für alle $0ms < I < 180ms$ erfüllt ist, ist schritthaltende Verarbeitung möglich.

1.2.2. Aufgabe 2

Zwei vernetzte Rechner sollen über das PTP (Precision Time Protocol) zeitsynchronisiert werden.

1. Wie viele Pakete werden dazu zwischen den beiden Rechnern (Zeitserver, Client) ausgetauscht? (1P)

Drei Pakete.

2. Der Zeitserver schickt dem Client initial ein Paket. Welche zwei Informationen zieht der Client aus dem Paket? (4P)

Den Absendezeitpunkt beim Server (t1) und den Empfangszeitpunkt auf dem Client selbst (t2).

3. Nachdem der Server ein Paket mit einem Zeitstempel t3 vom Client erhalten hat, antwortet er mit einem weiteren Paket. Welche Information zieht der Client aus diesem Paket? (3P)

Den Empfangszeitpunkg vom dritten Paket t4. Dieser wird beim Server erfasst und muss dann zur Berechnung dem Client übermittelt werden.

4. Der Client ermittelt die folgenden vier Zeitstempel: t1=50ms, t2=20ms, t3=30ms und t4=100ms. Zeigen Sie mithilfe einer Rechnung, welchen Offset der Client zum Server hat. (3P)

o = (20ms-50ms-100ms+30ms)/2 = -50ms

5. Warum wird die notwendige Zeitkorrektur nicht in Form eines Zeitsprungs realisiert? (1P)

Bei Zeitsprüngen werden unter Umständen zeitgesteuerte Aktionen nicht oder doppelt ausgeführt.

6. Eine Applikation legt sich per `clock_nanosleep()` während der Zeitkorrektur über den Zeitgeber `CLOCK_REALTIME` für 120ms schlafen. Geben Sie die effektive Schlafenszeit der Applikation an, wenn die Uhrzeit 50ms zurückgesetzt wird. (2P)

Die Uhr wird vorgestellt, daher schläft die Applikation kürzer. Die Schlafenszeit beträgt demnach 70ms.

1.2.3. Aufgabe 3

Zur Bestimmung der Zeitdauer einer Messwertaufnahme wird eine Differenzzeitmessung durchgeführt.

1. Mit welcher für Realzeitsysteme geeigneten Systemfunktion erfassen Sie die Zeit? (1P)

`clock_gettime()`

2. Welche Zeitgeber haben Sie in der Vorlesung kennen gelernt? Geben Sie deren Namen, wesentliches Merkmal und den Bezugspunkt an. (3P)

Name	Kennzeichen	Bezugspunkt
CLOCK_MONOTONIC	reagiert nicht auf Zeitsprünge	nicht definiert, Start des Systems
CLOCK_REALTIME	reagiert auf Zeitsprünge	1.1.1970

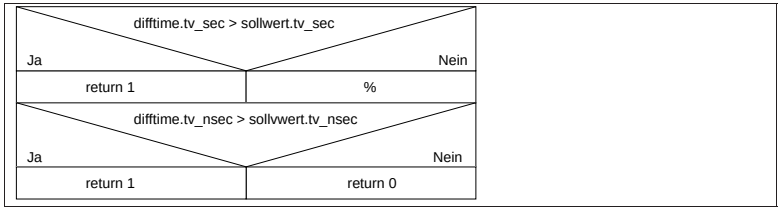
3. Skizzieren Sie ein Codefragment, das die Reaktionszeit der Funktion `auszumessende_codsequenz()` per Differenzzeitmessung über die Variablen `struct timespec vorher` und `struct timespec nachher` erfasst und mit der in der Vorlesung vorgestellten Poormans-Methode in der Variablen `long reaktionszeit_nsec` ablegt. (6P)


```
struct timespec vorher, nachher;
long reaktionszeit_nsec;

clock_gettime( CLOCK_MONOTONIC, &vorher );
auszumessende_codesequenz();
clock_gettime( CLOCK_MONOTONIC, &nachher );

reaktionszeit_nsec = ((nachher.tv_sec - vorher.tv_sec)*1000000000) +
     (nachher.tv_nsec - vorher.tv_nsec);
```

4. Warum eignet sich die Poormans-Lösung nur zur Erfassung kleiner Zeiten? Geben Sie auch die Größenordnung an, in der sich die Zeitdifferenz bei Verwendung von `struct timespec` (auf einem 32-Bit-System) bewegen darf. (2,2=4P)

Begründung: Durch die Multiplikation mit 1 Milliarde gehen im Sekundenteil Informationen verloren (Überlauf). Die Zeit muss also in dem Bereich liegen, innerhalb dem keine Informationen verloren gehen.
Größenordnung: 4 Sekunden (also 4*1 Mrd. entspricht in etwa dem Wertebereich einer 32-Bit Speicherzelle).

5. Das Ergebnis einer Differenzzeitmessung liegt in der Variablen `struct timespec difftime` vor. Zeichnen Sie ein **Struktogramm**, das »difftime« mit einem Sollwert `struct timespec sollwert` vergleicht. Ist der Sollwert überschritten, liefert die Funktion »1« zurück, ansonsten (gleich oder kleiner) »0«. (6P)

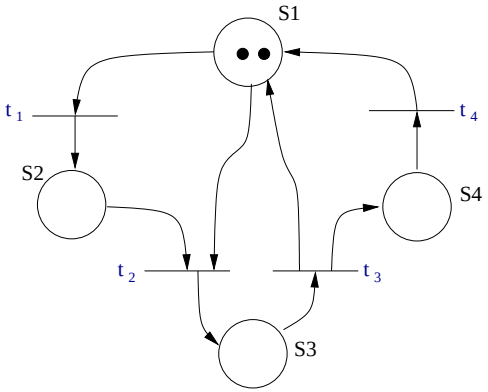


1.2.4. Aufgabe 4

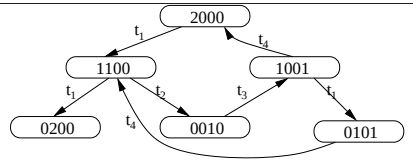
Eine Fertigungsanlage besteht aus 4 Stationen. Werkzeugträger transportieren hierin die Werkstücke zwischen den Stationen. Werkzeugträger in der Station 1 wandern entweder in die Station 2 oder - immer zusammen mit einem Werkzeugträger aus Station 2 - in die Station 3. Nach der Bearbeitung in Station 3 wird ein Träger in die Station 4 weitergeführt und ein zweiter Träger zurück in die Station 1. Auch der Träger aus Station 4 wandert nach der Bearbeitung wieder in die Station 1.

Zu Beginn sollen sich zwei Werkzeugträger in der Station 1 befinden.

1. Zeichnen Sie das aus vier Stellen bestehende Petrinetz des Systems (als Bedingungs-/Ereignisnetz) und geben Sie eine sinnvolle Anfangsmarkierung an. (8P)



2. Kann es zu einem Deadlock kommen? Erstellen Sie den Erreichbarkeitsgraphen und werten Sie diesen aus. (8P)



Der Erreichbarkeitsgraph hat zwei Knoten ohne abgehende Kanten. Das System ist also deadlockgefährdet.

Kapitel 2. Klausur Realzeitsysteme WS2012/2013

2.1. Hinweise zum Prüfungsablauf

Tag der Prüfung	7.02.2013
Beginn	7:30 Uhr
Dauer	120 Minuten
Ort	Audimax

Die Prüfung besteht aus einem Fragenteil ohne Unterlagen und einem Aufgabenteil mit Unterlagen.

Die Bearbeitungszeit für den Fragenteil, zu dem keinerlei Hilfsmittel und Unterlagen zugelassen sind, beträgt 30 Minuten. Beantworten Sie die Fragen auf einem eigenen Prüfungsbogen. Sobald Sie mit der Bearbeitung des ersten Prüfungsteiles fertig sind, können Sie mit dem zweiten Teil fortfahren. Unterlagen dürfen allerdings erst verwendet werden, nachdem sämtliche Lösungsblätter des ersten Teiles eingesammelt worden sind.

Der zweite Teil der Prüfung besteht aus insgesamt 3 Aufgaben. Die einzelnen Aufgaben und größtenteils auch die einzelnen Teilaufgaben sind unabhängig voneinander lösbar. Verwenden Sie bitte für jede Aufgabe ein eigenes Lösungsblatt. Die für diesen Teil zur Verfügung stehende Bearbeitungszeit beträgt ingesamt 90 Minuten.

Bitte legen Sie am Ende der Prüfungszeit alle Lösungblätter ineinander.

Die über den Aufgaben stehende Punktzahl ist vorläufig und unverbindlich.

Achtung

Versehen Sie unbedingt alle Prüfungsbögen und Papiere, die Sie mit abgeben, mit Ihrem Namen und Ihrer Matrikelnummer! Geben Sie auf jeden Fall alle Prüfungsbögen ab, auch wenn Sie eine Aufgabe nicht bearbeitet haben.

Viel Erfolg!

2.2. Ohne Unterlagen

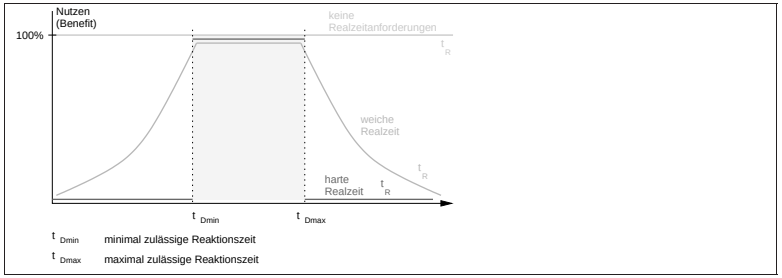
1. Definieren Sie den Begriff »Realzeitsystem«. (1P)

Ein Realzeitsystem muss neben den funktionalen Anforderungen auch noch zeitlichen Anforderungen genügen.

2. Wie lauten die beiden Realzeitbedingungen (genaue Angabe)? (2P)

1. RZB	$\rho_{ges} = \sum_{i=1}^{n} \frac{t_{Emax,i}}{t_{Pmin,i}} \leq c; \quad mit\ c = Anzahl\ Rechnerkerne$
2. RZB	Für alle Rechenzeitanforderungen i muss gelten: $t_{Dmin,i} \leq t_{Rmin,i} \leq t_{Rmax,i} \leq t_{Dmax,i}$

3. Erläutern Sie anhand der Nutzen-Funktion mithilfe einer **genauen** Skizze den Unterschied zwischen harter und weicher Realzeit! (2P)



4. Bitoperationen
   a. Mit welchem logischen Operator können Sie einzelne Bits eines Bitfeldes invertieren? (1,2=3P)

Mit einer bitweisen XOR-Verknüpfung („~”).

   b. Skizzieren Sie die Codezeile, die das zweite und zwölfte Bit (Wertigkeit 2^1 und 2^{11}) des Bitfeldes `bit_feld` invertiert und gleichzeitig den Zustand der übrigen Bits unverändert lässt.

```
bit_feld ~= 0x0802;
```

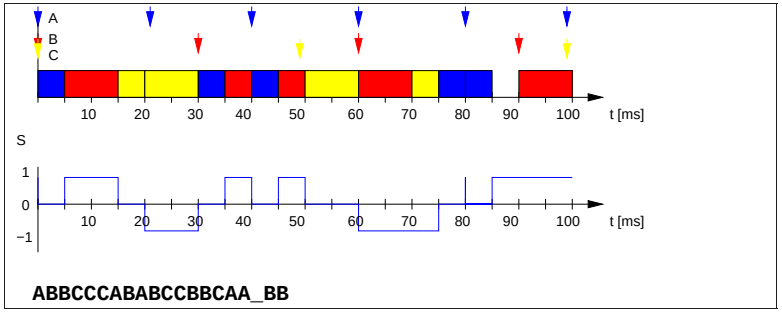
5. Gegeben sind die Zeitparameter des folgenden Rechenprozessgebildes: (1,6,1,2=10P)

Job	$t_{Emax,j}$	$t_{Pmin,j}$	$t_{Dmin,j}$	$t_{Dmax,j}$
A	5ms	20ms	0ms	15ms
B	10ms	30ms	0ms	30ms
C	15ms	50ms	0ms	50ms

   a. Verteilen Sie die Prioritäten.

A=1, B=2, C=3 (niedrig)

   b. Job A und Job C haben einen gemeinsamen kritischen Abschnitt, der durch ein Semaphor geschützt ist. Zeichnen Sie die Rechnerkernbelegung und den Wert des Semaphors im Bereich von $0 \leq t \leq 100ms$.



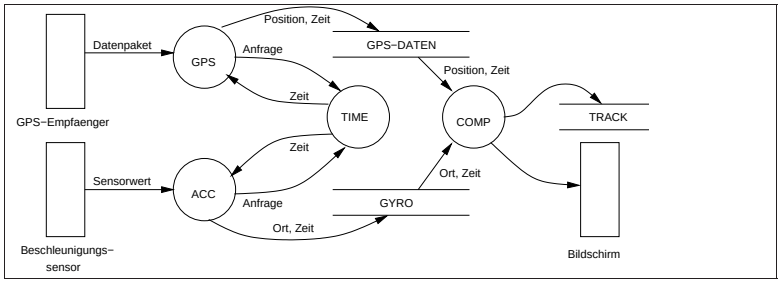
   c. Zu welchem Zeitpunkt kommt es zu einer Deadline-Verletzung?

75ms

   d. Wie lautet der Name der vorliegenden Rechenprozess-Konstellation und mit welchem Verfahren (Protokoll) kann die Deadline-Verletzung vermieden werden?

Prioritätsinversion. Prioritätsvererbung.

6. Das Taskgebilde eines Trackingsystems besteht aus vier Tasks. Die Task TIME liefert auf Anfrage aktuelle Zeitinformationen. Die Task GPS liest vom GPS-Empfänger Datenpakete ein und legt die daraus berechnete Position zusammen mit der vom Betriebssystem zur Verfügung gestellten Zeit in den Datenspeicher GPS-DATEN ab. Die Task ACC holt sich die Daten des Beschleunigungssensors und berechnet aus diesen Daten ebenfalls eine Ortsinformation, die zusammen mit der Zeit im Datenspeicher GYRO abgelegt wird. Die Task COMP berechnet aus den Daten in den beiden Datenspeicher die aktuelle Position, legt diese in den Datenspeicher TRACK und gibt zugleich die Position auf den Bildschirm aus. Skizzieren Sie das Datenflussdiagramm (DFD) des Rechnersystems. Beschriften Sie Verarbeitungseinheiten, Datenflüsse, Datenquellen und -senken! (6P)



7. Skizzieren Sie den Code einer **Funktion** `int read_with_time( struct timespec *mtime )`. Die Funktion legt die aktuelle Uhrzeit in `mtime` ab und liest vom Gerät `/dev/carrera` einen Integer-Wert (Status) ein. Im Fehlerfall gibt sie »-1« zurück, ansonsten den eingelesenen Integer-Wert. (6P)

```
int digital_in( struct timespec *mtime )
{
    int fd;
    int value;

    if (clock_gettime(CLOCK_MONOTONIC,&mtime)==(-1)) {
     return -1;
    }
    fd = open( "/dev/carrera", O_RDONLY );
    if (fd<0) {
        return -1;
    }
    if (read(fd,value,sizeof(value))≤0) {
        close( fd );
        return -1;
    }
    close( fd );
    return value;
}
```

Mit Unterlagen

Name: _____

Matrikelnr.: _____

2.2.1. Aufgabe 1

Gegeben sind vier Rechenzeitanforderungen, die über die Tasks T1, T2, T3 und T4 mit den folgenden Kenndaten bearbeitet werden:

RZ-Anf.	$t_{Pmin,i}$	$t_{Dmin,i}$	$t_{Dmax,i}$	$t_{Ph,i}$	$t_{Emin,i}$	$t_{Emax,i}$
T1	40 ms	0 ms	20 ms	0 ms	0 ms	10 ms
T2	100 ms	0 ms	100 ms	0 ms	0 ms	12 ms
T3	50 ms	0 ms	50 ms	0 ms	0 ms	30 ms
T4	100 ms	0 ms	100 ms	0 ms	0 ms	90 ms

1. Leiten Sie die folgenden Kenndaten her (1,1,1,1=4P)
   a. $\rho_{max,T1}$

   $\rho_{max,T1}=t_{Emax,T1}$ /$t_{Pmin,T1}$=10ms/40ms=0.25

   b. $\rho_{max,T2}$

   $\rho_{max,T2}=t_{Emax,T2}$ /$t_{Pmin,T2}$=12ms/100ms=0.12

   c. $\rho_{max,T3}$

   $\rho_{max,T3}=t_{Emax,T3}$ /$t_{Pmin,T3}$=30ms/50ms=0.6

   d. $\rho_{max,T4}$

   $\rho_{max,T4}=t_{Emax,T4}$ /$t_{Pmin,T4}$=90ms/100ms=0.9

2. Wie lautet allgemein die Bedingung für den hinreichenden Schedulingtest (Utilization u für prioritätengesteuertes Scheduling)? (2P)

   $$u = \sum_{j=1}^{n} \frac{t_{Emax,j}}{\min\left(t_{Dmax,j},\, t_{Pmin,j}\right)} \leq n(2^{\frac{1}{n}} - 1)$$

3. Bis zu welcher Auslastungsgrenze (in Prozent) ist bei prioritätengesteuertem Scheduling und **vier Tasks** eine schritthaltende Verarbeitung immer gewährleistet? (1P)

   u $\leq$ 0.757 (75.7 Prozent)

4. Berechnen Sie die Gesamtauslastung ρ_{ges} für die vier Tasks. (1P)

   $\rho_{max,ges}$=0.25+0.12+0.6+0.9=1.87

5. Wie viele CPU-Kerne sind notwendig, um die vier Tasks (prinzipiell) schritthaltend verarbeiten zu können? (1P)

   Zwei CPU-Kerne (da die maximale Auslastung zwischen 1 und 2 liegt).

Zur Bearbeitung der Aufgabe soll jetzt eine Dualcore-CPU eingesetzt werden. Die angegebenen Verarbeitungszeiten gelten weiterhin (Verarbeitungszeit auf einem Core).

6. Verteilen Sie die vier Tasks auf die beiden CPU-Kerne 0 und 1. (1P)

CPU-Kern 0	T1, T2, T3
CPU-Kern 1	T4

*Im folgenden soll nur der CPU-Kern 0 betrachtet werden. Dieser bearbeitet ausschließlich die **drei** Tasks T1, T2 und T3. Die Tasks bekommen die Prioritäten T1=1 (hoch), T2=2, T3=3 (niedrig) zugewiesen.*

7. Echtzeitnachweis bei Einsatz eines prioritätengesteuerten Schedulings. (1,1,1,1,1,7,3=15P)

a. Geben Sie die Rechenzeitanforderungsfunktion $t_{C,1}(t)$ für Jobs mit Priorität 1 an.

$$t_{c,1}(t) = \lceil \frac{t}{40ms} \rceil \cdot 10ms$$

b. Berechnen Sie die maximale Reaktionszeit $t_{Rmax,1}$ für die zugehörigen Jobs.

$$t_{c,1}^{(1)} = 10ms$$
$$t_{c,1}^{(2)} = \lceil \frac{10ms}{40ms} \rceil \cdot 10ms = 10ms$$
$$t_{Rmax,1} = 10ms$$

c. Geben Sie die Rechenzeitanforderungsfunktion $t_{C,2}(t)$ für Jobs mit Priorität 2 an.

$$t_{c,2}(t) = \lceil \frac{t}{40ms} \rceil \cdot 10ms + \lceil \frac{t}{100ms} \rceil \cdot 12ms$$

d. Berechnen Sie die maximale Reaktionszeit $t_{Rmax,2}$ für Jobs der Priorität 2.

$$t_{c,2}^{(1)} = 22ms$$
$$t_{c,2}^{(2)} = \lceil \frac{22ms}{40ms} \rceil \cdot 10ms + \lceil \frac{22ms}{100ms} \rceil \cdot 12ms = 22ms$$
$$t_{Rmax,2} = 22ms$$

e. Geben Sie die Rechenzeitanforderungsfunktion $t_{C,3}(t)$ für Jobs mit Priorität 3 an.

$$t_{c,3}(t) = \lceil \frac{t}{40ms} \rceil \cdot 10ms + \lceil \frac{t}{100ms} \rceil \cdot 12ms + \lceil \frac{t}{50ms} \rceil \cdot 30ms$$

f. Berechnen Sie die maximale Reaktionszeit $t_{Rmax,3}$ für Jobs der Priorität 3.

$$t_{c,3}^{(1)} = 52ms$$

$$t_{c,3}^{(2)} = \lceil \frac{52ms}{40ms} \rceil \cdot 10ms + \lceil \frac{52ms}{100ms} \rceil \cdot 12ms + \lceil \frac{52ms}{50ms} \rceil \cdot 30ms = 92ms$$

$$t_{c,3}^{(3)} = \lceil \frac{92ms}{40ms} \rceil \cdot 10ms + \lceil \frac{92ms}{100ms} \rceil \cdot 12ms + \lceil \frac{92ms}{50ms} \rceil \cdot 30ms = 102ms$$

$$t_{c,3}^{(4)} = \lceil \frac{102ms}{40ms} \rceil \cdot 10ms + \lceil \frac{102ms}{100ms} \rceil \cdot 12ms + \lceil \frac{102ms}{50ms} \rceil \cdot 30ms = 144ms$$

$$t_{c,3}^{(5)} = \lceil \frac{144ms}{40ms} \rceil \cdot 10ms + \lceil \frac{144ms}{100ms} \rceil \cdot 12ms + \lceil \frac{144ms}{50ms} \rceil \cdot 30ms = 154ms$$

$$t_{c,3}^{(6)} = \lceil \frac{154ms}{40ms} \rceil \cdot 10ms + \lceil \frac{154ms}{100ms} \rceil \cdot 12ms + \lceil \frac{154ms}{50ms} \rceil \cdot 30ms = 184ms$$

$$t_{c,3}^{(7)} = \lceil \frac{184ms}{40ms} \rceil \cdot 10ms + \lceil \frac{184ms}{100ms} \rceil \cdot 12ms + \lceil \frac{184ms}{50ms} \rceil \cdot 30ms = 194ms$$

$$t_{c,3}^{(8)} = \lceil \frac{194ms}{40ms} \rceil \cdot 10ms + \lceil \frac{194ms}{100ms} \rceil \cdot 12ms + \lceil \frac{194ms}{50ms} \rceil \cdot 30ms = 194ms$$

$$t_{Rmax,3} = 194ms$$

g. Zeigen Sie anhand der 2. Echtzeitbedingung (Rechtzeitigkeitsbedingung), ob schritthaltende Verarbeitung für die drei Tasks möglich ist oder nicht.

T1: $0ms \leq 0ms \leq 10ms \leq 20ms$ Ja, diese Bedingung ist erfüllt.

T2: $0ms \leq 0ms \leq 22ms \leq 100ms$ Ja, diese Bedingung ist erfüllt.

T3: $0ms \leq 0ms \leq 194ms \leq 50ms$ Nein, diese Bedingung ist nicht erfüllt.

Schritthaltende Verarbeitung ist nicht möglich!

8. Echtzeitnachweis bei Einsatz eines Deadline-Schedulings. (3,4,8=15P)

a. Geben Sie die Gesamt-Rechenzeitanforderungsfunktion $t_{C,ges}(I)$ an.

$$t_{C,ges}(I) = \sum_{j=1}^{n} \lfloor \frac{I + t_{Pmin,j} - t_{Dmax,j} - t_{Ph,j}}{t_{Pmin,j}} \rfloor \cdot t_{Emax,j}$$

$$t_{C,ges}(I) = \lfloor \frac{I + 40ms - 20ms + 0ms}{40ms} \rfloor \cdot 10ms + \lfloor \frac{I + 100ms - 100ms}{100ms} \rfloor \cdot 12ms$$

$$+ \lfloor \frac{I + 50ms - 50ms}{50ms} \rfloor \cdot 30ms$$

$$t_{C,ges}(I) = \lfloor \frac{I + 20ms}{40ms} \rfloor \cdot 10ms + \lfloor \frac{I}{100ms} \rfloor \cdot 12ms + \lfloor \frac{I}{50ms} \rfloor \cdot 30ms$$

b. Für welche I müssen Sie $t_{C,ges}(I)$ konkret untersuchen? Geben Sie den Bereich an, in dem I zu untersuchen ist und zusätzlich die zu untersuchenden Intervalle in diesem Bereich.

I ist zu untersuchen im Bereich von $0 < I < kgV(40ms, 100ms, 50ms) = 200ms$; (da $t_{Ph} = 0$)
I_{T1}: {20ms, 60ms, 100ms, 140ms, 180ms}
I_{T2}: {100ms}
I_{T3}: {50ms, 100ms, 150ms}
I: {20ms, 50ms, 60ms, 100ms, 140ms, 150ms, 180ms}

c. Führen Sie den Echtzeitnachweis durch. Ist schritthaltende Verarbeitung möglich?

$$t_{C,ges}(I) = \lfloor\frac{I+20ms}{40ms}\rfloor \cdot 10ms + \lfloor\frac{I}{100ms}\rfloor \cdot 12ms + \lfloor\frac{I}{50ms}\rfloor \cdot 30ms$$

$$t_{C,ges}(20ms) = \lfloor\frac{40ms}{40ms}\rfloor \cdot 10ms + \lfloor\frac{20ms}{100ms}\rfloor \cdot 12ms + \lfloor\frac{20ms}{50ms}\rfloor \cdot 30ms = 10ms$$

$$t_{C,ges}(50ms) = \lfloor\frac{70ms}{40ms}\rfloor \cdot 10ms + \lfloor\frac{50ms}{100ms}\rfloor \cdot 12ms + \lfloor\frac{50ms}{50ms}\rfloor \cdot 30ms = 40ms$$

$$t_{C,ges}(60ms) = \lfloor\frac{80ms}{40ms}\rfloor \cdot 10ms + \lfloor\frac{60ms}{100ms}\rfloor \cdot 12ms + \lfloor\frac{60ms}{50ms}\rfloor \cdot 30ms = 50ms$$

$$t_{C,ges}(100ms) = \lfloor\frac{120ms}{40ms}\rfloor \cdot 10ms + \lfloor\frac{100ms}{100ms}\rfloor \cdot 12ms + \lfloor\frac{100ms}{50ms}\rfloor \cdot 30ms = 102ms$$

$$t_{C,ges}(140ms) = \lfloor\frac{160ms}{40ms}\rfloor \cdot 10ms + \lfloor\frac{140ms}{100ms}\rfloor \cdot 12ms + \lfloor\frac{140ms}{50ms}\rfloor \cdot 30ms = 112ms$$

$$t_{C,ges}(150ms) = \lfloor\frac{170ms}{40ms}\rfloor \cdot 10ms + \lfloor\frac{150ms}{100ms}\rfloor \cdot 12ms + \lfloor\frac{150ms}{50ms}\rfloor \cdot 30ms = 142ms$$

$$t_{C,ges}(180ms) = \lfloor\frac{200ms}{40ms}\rfloor \cdot 10ms + \lfloor\frac{180ms}{100ms}\rfloor \cdot 12ms + \lfloor\frac{180ms}{50ms}\rfloor \cdot 30ms = 152ms$$

Da die Bedingung $t_{C,ges}(I) \leq I$ für I=100ms NICHT erfüllt ist, ist schritthaltende Verarbeitung nicht möglich.

2.2.2. Aufgabe 2

Ein Job soll nach exakt 9999ms eine Ausgabe tätigen. Diese Zeit soll per Schlafen überbrückt werden.

1. Nennen Sie drei Systemaufrufe, die in der Vorlesung vorgestellt wurden, mit denen Sie das Schlafen realisieren können. (3P)

   `clock_nanosleep()`, `nanosleep()` und `usleep()`.

2. Welche Funktion zum Schlafenlegen verwenden Sie für eine professionelle Lösung? Begründen Sie Ihre Auswahl. (3P)

`clock_nanosleep()`, weil hier der Zeitgeber ausgewählt werden kann und auch ob relativ oder absolut geschlafen werden soll.

3. Ein Programmierer wählt für die gestellte Aufgabe die Funktion `int clock_nanosleep(clockid_t clock_id, int flags, const struct timespec *trequest, struct timespec *tremain)`. Welchen Zeitgeber verwendet er, wenn das Aufwecken unabhängig von Manipulationen an der Systemuhr nach der genannten Zeit stattfinden soll? (1P)

   CLOCK_MONOTONIC.

4. Mit welchem Wert wird er die Datenstruktur `struct timespec *trequest` initialisieren, wenn relativ 9999ms geschlafen werden soll? (2P)

   - tv_sec = 9 - tv_nsec= 999.000.000

Der Programmierer findet im Internet die folgende Codesequenz:

```
...
trequest.tv_sec  = 0;
trequest.tv_nsec = 250000000; /* 250 Millisekunden */
do {
    error=clock_nanosleep(CLOCK_MONOTONIC,0,&trequest,&tremain);
    trequest = tremain;
} while (error==EINTR);
if (error==0) {
    // Ausgabe
    ...
} else {
    // Fehlerbehandlung
    ...
}
```

5. Welche Aufgabe hat die Variable `tremain`? (1P)

   Sie nimmt im Fall einer Unterbrechung der Funktion clock_nanosleep() die Restzeit auf, die noch geschlafen werden muss.

6. Unter welchen Umständen wird in der gegebenen Codesequenz die do-while-Schleife mehrmals durchlaufen? Warum ist die do-while-Schleife notwendig? (3P)

   Die Funktion `clock_nanosleep()` kann durch Signale (Interrupts) unbeabsichtigt unterbrochen werden. Der erneute Aufruf stellt sicher, dass nicht zu früh weitergearbeitet wird.

7. Welchen Wert hat `tremain`, wenn die gegebene Codesequenz angepasst auf 9999ms abläuft und die zugehörige Task 1099ms nach Aufruf der Funktion `clock_nanospleep()` durch ein Signal unterbrochen wird? Wie lang wird die Task durch die gegebene Codesequenz insgesamt schlafen, wenn ein Schleifendurchlauf 100ns beträgt (Beide **Angaben in Sekunden und Nanosekunden**)? (2,2=4P)

tremain.tv_sec=8;
tremain.tv_nsec=900000000;

9 Sekunden und 999000100 Nanosekunden

8. Wie wird der Programmierer die Funktion `clock_nanosleep()` einsetzen, um den Fehler, der sich durch die unbeabsichtigte Unterbrechung ergibt, zu vermeiden? (2P)

Er wird auf einen Absolutzeitpunkt warten (`TIMER_ABSTIME`).

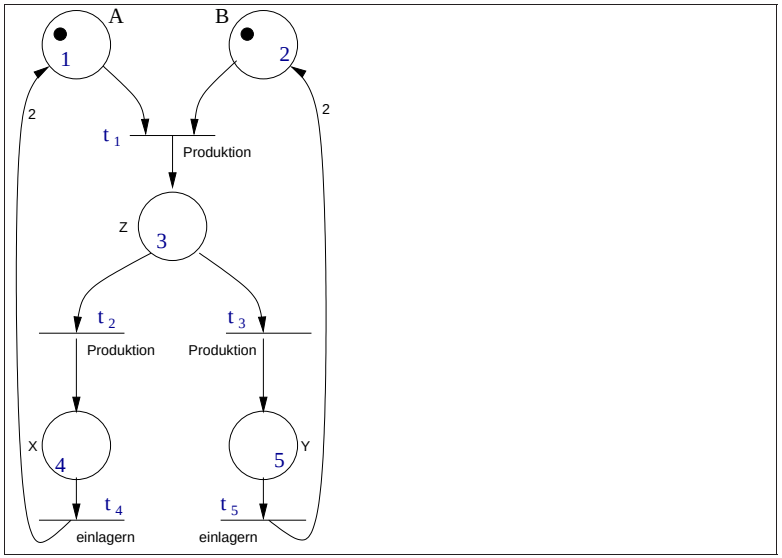
2.2.3. Aufgabe 3

In einem Lager A steht ein Werkstück X zur Verfügung, in einem Lager B ein Werkstück Y. In einem Produktionsschritt wird aus einem X und einem Y ein Werkstück Z produziert. Aus Z wiederum werden entweder zwei X oder zwei Y hergestellt, die in Lager A beziehungsweise Lager B abgelegt werden.

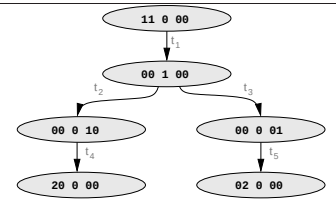
1. Erläutern Sie den Unterschied zwischen einem Bedingungs-/Ereignisnetz und einem Stellen-/Transitionsnetz. (1P)

Ein Stellen-/Transitionsnetz ermöglicht die Gewichtung von Übergängen. Bei einem Bedingungs-/Ereignisnetz ist jeder Übergang mit eins gewichtet.

2. Modellieren Sie die Vorgänge mit Hilfe eines aus fünf Stellen bestehenden Petrinetzes in Form eines Stellen-/Transitionsnetzes. Beschriften Sie die Plätze und Transitionen und geben Sie eine sinnvolle Anfangsmarkierung an. (10P)



3. Stellen Sie den zugehörigen Erreichbarkeitsgraphen auf. Kommt es zu einer Verklemmung? Begründen Sie Ihre Antwort! (10P)



Der Erreichbarkeitsgraph hat zwei Knoten ohne abgehende Kanten. Das System ist also deadlockgefährdet.

2.2.4. Aufgabe 4

Die Verfügbarkeit eines Systems sei gegeben durch

$p_{sys}=[1-(1-p_A^2)(1-p_B)]p_C$.

1. Leiten Sie aus der Gleichung die Konfiguration des Gesamstystems ab und stellen Sie diese in Form eines Verfügbarkeitsersatzschaltbildes dar. Beschriften Sie die einzelnen Komponenten! (5P)

2. Geben Sie die Dauerverfügbarkeit des Systems an, falls gilt: $p_A = p_B = p_C = 0.981$ (Rechnung mit fünfstelliger Genauigkeit). (2P)

$p_{sys}=[1-(1-0.981^2)\,(1-0.981)]\cdot 0.981=.980298446679$

3. Sicherheitsbetrieb. (1,1,1=3P)
   a. Zeichnen Sie das Verfügbarkeitsersatzschaltbild, falls alle Komponenten (p_A, p_A, p_B und p_C) für einen sicheren Betrieb des Systems notwendig sind.

   b. Geben Sie für diesen Fall die Gesamtverfügbarkeit des Systems formelmäßig an.

$p_{sys}= p_A \cdot p_A \cdot p_B \cdot p_C$

   c. Berechnen Sie die Gesamtverfügbarkeit. Alle Einzel-Komponenten besitzen eine Dauerverfügbarkeit von p=0.981.

$p_{sys}= 0.926138694321$

Kapitel 3. Klausur Realzeitsysteme SS2012

3.1. Hinweise zum Prüfungsablauf

Tag der Prüfung	18.09.2012
Beginn	8:30 Uhr
Dauer	120 Minuten
Ort	B320

Die Prüfung besteht aus einem Frageteil ohne Unterlagen und einem Aufgabenteil mit Unterlagen.

Die Bearbeitungszeit für den Frageteil, zu dem keinerlei Hilfsmittel und Unterlagen zugelassen sind, beträgt 30 Minuten. Beantworten Sie die Fragen auf einem eigenen Prüfungsbogen. Sobald Sie mit der Bearbeitung des ersten Prüfungsteiles fertig sind, können Sie mit dem zweiten Teil fortfahren. Unterlagen dürfen allerdings erst verwendet werden, nachdem sämtliche Lösungsblätter des ersten Teiles eingesammelt worden sind.

Der zweite Teil der Prüfung besteht aus insgesamt 3 Aufgaben. Die einzelnen Aufgaben und größtenteils auch die einzelnen Teilaufgaben sind unabhängig voneinander lösbar. Verwenden Sie bitte für jede Aufgabe ein eigenes Lösungsblatt. Die für diesen Teil zur Verfügung stehende Bearbeitungszeit beträgt ingesamt 90 Minuten.

Bitte legen Sie am Ende der Prüfungszeit alle Lösungblätter ineinander.

Die über den Aufgaben stehende Punktzahl ist vorläufig und unverbindlich.

Achtung

Versehen Sie unbedingt alle Prüfungsbögen und Papiere, die Sie mit abgeben, mit Ihrem Namen und Ihrer Matrikelnummer! Geben Sie auf jeden Fall alle Prüfungsbögen ab, auch wenn Sie eine Aufgabe nicht bearbeitet haben.

Viel Erfolg!

3.2. Ohne Unterlagen

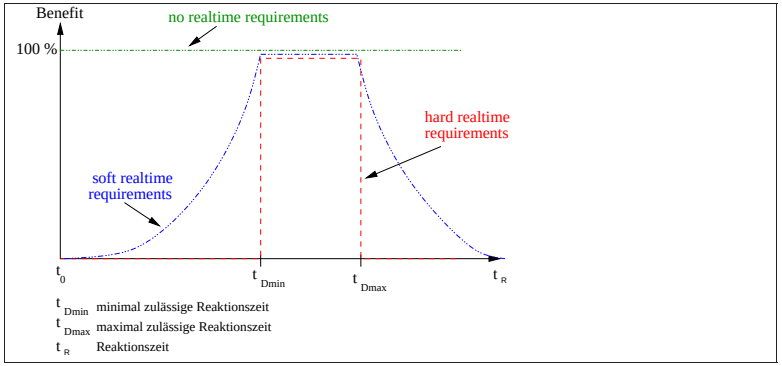
1. Definieren Sie den Begriff »Realzeitsystem«. (1P)

Ein Realzeitsystem muss neben den funktionalen Anforderungen auch noch zeitlichen Anforderungen genügen.

2. Wie lauten die beiden Realzeitbedingungen (genaue Angabe)? (2P)

1. RZB	$\rho_{ges} = \sum_{i=1}^{n} \frac{t_{Emax,i}}{t_{Pmin,i}} \leq c; \quad mit\ c = Anzahl\ Rechnerkerne$
2. RZB	Für alle Rechenzeitanforderungen *i* muss gelten: $t_{Dmin,i} \leq t_{Rmin,i} \leq t_{Rmax,i} \leq t_{Dmax,i}$

3. Erläutern Sie anhand der Nutzen-Funktion mithilfe einer **genauen** Skizze den Unterschied zwischen harter und weicher Realzeit! (2P)



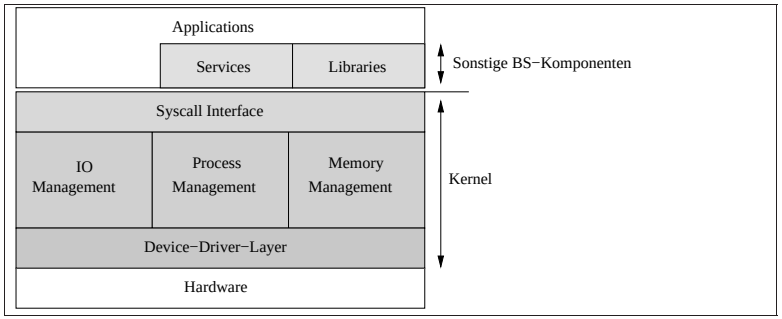
4. Bitoperationen
   a. Mit welchem logischen Operator können Sie einzelne Bits eines Bitfeldes setzen? (1,2=3P)

Mit einer bitweisen Oder-Verknüpfung („|").

   b. Skizzieren Sie die Codezeile, die das vierte und sechste Bit (Wertigkeit 2^3 und 2^5) des Bitfeldes `bit_feld` setzt und gleichzeitig den Zustand der übrigen Bits unverändert lässt.

```
bit_feld |= 0x0028;
```

5. Skizzieren Sie die in der Vorlesung vorgestellte Architektur eines (Realzeit-) Betriebssystems. (3P)



6. Schutz kritischer Abschnitte
   a. Was ist beim Zugriff auf globale Variablen zu beachten? (1P)

Greifen Codesequenzen quasi gleichzeitig auf die Daten zu, kann es zu Inkonsistenzen kommen.

   b. Welche Funktionen haben Sie in der Vorlesung kennengelernt, um derartig kritische Codesequenzen zu schützen? (1P)

Der kritische Abschnitt kann durch ein Semaphor geschützt werden. Die zugehörigen Funktionen sind P(S) und V(S).

   c. Ergänzen Sie den Code, damit es zu keiner Race-Condition kommt. (2P)

Codesequenz A

```
P(S)
glob_var = glob_var + 1;
V(S)
```

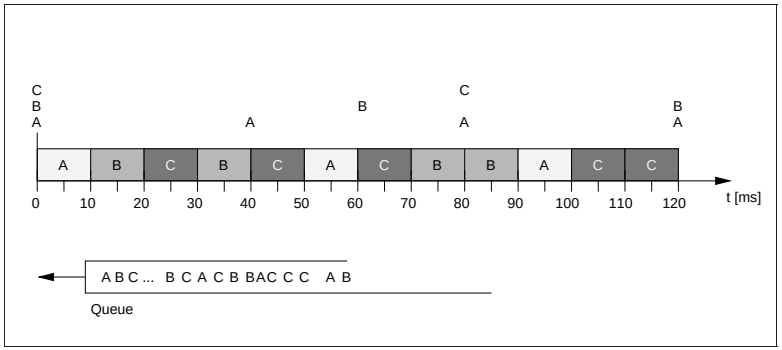
Codesequenz B

```
P(S)
if (glob_var > 0 )
    glob_var = glob_var − 1;
V(S)
```

7. Von drei Tasks sind die folgenden konstanten Prozess- und konstanten Verarbeitungszeiten bekannt.

Task	t_P	t_E
A	40ms	10ms
B	60ms	20ms
C	80ms	30ms

Skizzieren Sie die Rechnerkernbelegung im Bereich von $0 \leq t \leq 120$ms im Fall eines Zeitscheibenverfahrens. Eine Zeitscheibe sei dabei 10ms lang. (6P)

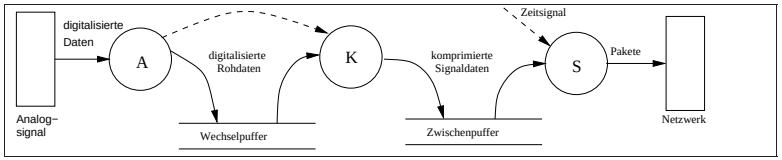


8. Skizzieren Sie den Code einer **Funktion** `int digital_in( unsigned int *value );`, mit der Sie vom Gerät `/dev/digital_in` einen Wert einlesen und an die übergebene Variablenadresse von `value` ablegen. Im Fehlerfall gibt die Funktion »-1«, ansonsten »0« zurück. (4P)

```
int digital_in( unsigned int *value )
{
    int fd;

    fd = open( "/dev/digital_in", O_RDONLY );
    if (fd<0) {
        return -1;
    }
    if (read(fd,value,sizeof(value))≤0) {
        close( fd );
        return -1;
    }
    close( fd );
    return 0;
}
```

9. Ein Rechnersystem wird zur Verarbeitung von Echtzeitdaten eingesetzt: Ein Thread A ist dafür verantwortlich, dass ein Analogsignal periodisch abgetastet wird. Er legt die digitalisierten Daten in einem Wechselpuffer ab. Sobald der Puffer gefüllt ist, wird ein Thread K geweckt, der die Daten ausliest und komprimiert in einen Zwischenpuffer ablegt. Ein (periodisch aktivierter) Thread S ist schließlich für die zeitversetzte Übertragung der Daten als Pakete über ein Netzwerk zuständig.

   Skizzieren Sie das Datenflussdiagramm (DFD) des Rechnersystems. Beschriften Sie Verarbeitungseinheiten, Datenflüsse, Datenquellen und -senken! Eränzen Sie das DFD um den Kontrollfluss. (5P)



Mit Unterlagen

Name: _____

Matrikelnr.: _____

3.2.1. Aufgabe 1

Zur Regelung einer Produktionsmaschine werden die drei Tasks A, E und V benötigt, die die folgenden Kenndaten besitzen:

Job	$t_{Pmin,i}$	$t_{Dmin,i}$	$t_{Dmax,i}$	$t_{A,i}$	$t_{Emin,i}$	$t_{Emax,i}$	$t_{Rmin,i}$	$\rho_{max,i}$
V	200 ms	0 ms	200 ms	0 ms	10 ms	20 ms		
E	100 ms	0 ms	100 ms	0 ms	0 ms	10 ms		
A	250 ms	0 ms	220 ms	0 ms	0 ms	30 ms		

1. Leiten Sie die folgenden Kenndaten her
   a. $\rho_{max,V}$ (Job V) (1P)

   $\rho_{max,V}=t_{Emax,V}/t_{Pmin,V}=20ms/200ms=0.1$

   b. $\rho_{max,E}$ (Job E) (1P)

   $\rho_{max,E}=t_{Emax,E}/t_{Pmin,E}=10ms/100ms=0.1$

   c. $\rho_{max,A}$ (Job A) (1P)

   $\rho_{max,A}=t_{Emax,A}/t_{Pmin,A}=30ms/250ms=0.12$

   d. ρ_{ges} (Gesamtauslastung) (1P)

   $\rho_{max,ges}=0.1+0.1+0.12=0.32$

2. Wie viele Produktionsmaschinen *n* können durch den Rechner gesteuert werden? Geben Sie zunächst die maximale Auslastung $\rho_{max,ges}(n)$ in Abhängigkeit von der Anzahl Produktionsmaschinen an und lösen Sie die Gleichung dann nach n auf. (2P)

   $\rho_{max,ges}(n) = n \cdot 0.32 \leq 1$
   $n \leq 1/0.32 = 3.125$

   Es können maximal drei Produktionsmaschinen gesteuert werden.

Für die folgenden Teilaufgaben sollen drei Produktionsmaschinen gesteuert werden (n=3).

3. Wie viele Tasks (V, E, A) sind im Fall von n=3 für die weiteren Berechnungen zu berücksichtigen? (1P)

   Dreimal Task V, dreimal Task E, dreimal Task A.

4. Geben Sie den Tasks Prioritäten (1=hoch, 2=mittel, 3=niedrig). (1P)

Gemäß Faustregel: Alle Tasks E Priorität 1, V Priorität 2, A Priorität 3.

5. Echtzeitnachweis bei Einsatz eines prioritätengesteuerten Schedulings. (1,1,1,1,1,9,3=17P)

a. Geben Sie die Rechenzeitanforderungsfunktion $t_{C,1}(t)$ für die Jobs mit Priorität 1 an.

$$t_{c,1}(t) = n \cdot \lceil \frac{t}{100ms} \rceil \cdot 10ms = \lceil \frac{t}{100ms} \rceil \cdot 30ms$$

b. Berechnen Sie die maximale Reaktionszeit $t_{Rmax,1}$ für die zugehörigen Jobs.

$$t_{c,1}^{(1)} = 30ms$$
$$t_{c,1}^{(2)} = \lceil \frac{30ms}{100ms} \rceil \cdot 30ms = 30ms$$
$$t_{Rmax,1} = 30ms$$

c. Geben Sie die Rechenzeitanforderungsfunktion $t_{C,2}(t)$ für Jobs mit Priorität 2 an.

$$t_{c,2}(t) = 3 \cdot \lceil \frac{t}{100ms} \rceil \cdot 10ms + 3 \cdot \lceil \frac{t}{200ms} \rceil \cdot 20ms = \lceil \frac{t}{100ms} \rceil \cdot 30ms + \lceil \frac{t}{200ms} \rceil \cdot 60ms$$

d. Berechnen Sie die maximale Reaktionszeit $t_{Rmax,2}$ für Jobs der Priorität 2.

$$t_{c,2}^{(1)} = 90ms$$
$$t_{c,2}^{(2)} = \lceil \frac{90ms}{100ms} \rceil \cdot 30ms + \lceil \frac{90ms}{200ms} \rceil \cdot 60ms = 90ms$$
$$t_{Rmax,2} = 90ms$$

e. Geben Sie die Rechenzeitanforderungsfunktion $t_{C,3}(t)$ für Jobs mit Priorität 3 an.

$$t_{c,3}(t) = \lceil \frac{t}{100ms} \rceil \cdot 30ms + \lceil \frac{t}{200ms} \rceil \cdot 60ms + n \cdot \lceil \frac{t}{250ms} \rceil \cdot 30ms = \lceil \frac{t}{100ms} \rceil \cdot 30ms + \lceil \frac{t}{200ms} \rceil \cdot 60ms + \lceil \frac{t}{250ms} \rceil \cdot 90ms$$

f. Berechnen Sie die maximale Reaktionszeit $t_{Rmax,3}$ für Jobs der Priorität 3.

$$t_{c,3}^{(1)} = 180ms$$
$$t_{c,3}^{(2)} = \lceil \frac{180ms}{100ms} \rceil \cdot 30ms + \lceil \frac{180ms}{200ms} \rceil \cdot 60ms + \lceil \frac{180ms}{250ms} \rceil \cdot 90ms = 210ms$$
$$t_{c,3}^{(3)} = \lceil \frac{210ms}{100ms} \rceil \cdot 30ms + \lceil \frac{210ms}{200ms} \rceil \cdot 60ms + \lceil \frac{210ms}{250ms} \rceil \cdot 90ms = 300ms$$
$$t_{c,3}^{(4)} = \lceil \frac{300ms}{100ms} \rceil \cdot 30ms + \lceil \frac{300ms}{200ms} \rceil \cdot 60ms + \lceil \frac{300ms}{250ms} \rceil \cdot 90ms = 390ms$$
$$t_{c,3}^{(5)} = \lceil \frac{390ms}{100ms} \rceil \cdot 30ms + \lceil \frac{390ms}{200ms} \rceil \cdot 60ms + \lceil \frac{390ms}{250ms} \rceil \cdot 90ms = 420ms$$
$$t_{c,3}^{(6)} = \lceil \frac{420ms}{100ms} \rceil \cdot 30ms + \lceil \frac{420ms}{200ms} \rceil \cdot 60ms + \lceil \frac{420ms}{250ms} \rceil \cdot 90ms = 510ms$$
$$t_{c,3}^{(7)} = \lceil \frac{510ms}{100ms} \rceil \cdot 30ms + \lceil \frac{510ms}{200ms} \rceil \cdot 60ms + \lceil \frac{510ms}{250ms} \rceil \cdot 90ms = 630ms$$
$$t_{c,3}^{(8)} = \lceil \frac{630ms}{100ms} \rceil \cdot 30ms + \lceil \frac{630ms}{200ms} \rceil \cdot 60ms + \lceil \frac{630ms}{250ms} \rceil \cdot 90ms = 720ms$$
$$t_{c,3}^{(9)} = \lceil \frac{720ms}{100ms} \rceil \cdot 30ms + \lceil \frac{720ms}{200ms} \rceil \cdot 60ms + \lceil \frac{720ms}{250ms} \rceil \cdot 90ms = 750ms$$
$$t_{c,3}^{(10)} = \lceil \frac{750ms}{100ms} \rceil \cdot 30ms + \lceil \frac{750ms}{200ms} \rceil \cdot 60ms + \lceil \frac{750ms}{250ms} \rceil \cdot 90ms = 750ms$$
$$t_{Rmax,3} = 750ms$$

g. Zeigen Sie anhand der 2. Echtzeitbedingung (Rechtzeitigkeitsbedingung), ob schritthaltende Verarbeitung für sämtliche Tasks möglich ist oder nicht.

E: $0ms \leq 0ms \leq 30ms \leq 100ms$	Ja, diese Bedingung ist erfüllt.
V: $0ms \leq 10ms \leq 90ms \leq 200ms$	Ja, diese Bedingung ist erfüllt.
A: $0ms \leq 0ms \leq 750ms \leq 220ms$	Nein, diese Bedingung ist nicht erfüllt.

Schritthaltende Verarbeitung ist nicht möglich!

6. Echtzeitnachweis bei Einsatz eines Deadline-Schedulings. (3,4,13=20P)

a. Geben Sie die Gesamt-Rechenzeitanforderungsfunktion $t_{C,ges}(I)$ an.

$$t_{C,ges}(I) = \sum \lfloor \frac{I + t_{Pmin,i} - t_{Dmax,i} - t_{A,i}}{t_{P,i}} \rfloor \cdot t_{Emax,i}$$

$$t_{C,ges}(I) = 3 \cdot \lfloor \frac{I - 100ms + 100ms + 0ms}{100ms} \rfloor \cdot 10ms + 3 \cdot \lfloor \frac{I - 200ms + 200ms}{200ms} \rfloor \cdot 20ms$$

$$+ 3 \cdot \lfloor \frac{I - 220ms + 250ms}{250ms} \rfloor \cdot 30ms$$

$$t_{C,ges}(I) = \lfloor \frac{I}{100ms} \rfloor \cdot 30ms + \lfloor \frac{Ims}{200ms} \rfloor \cdot 60ms + \lfloor \frac{I + 30}{250ms} \rfloor \cdot 90ms$$

b. Für welche I müssen Sie $t_{C,ges}$(I) konkret untersuchen? Geben Sie den Bereich an, in dem I zu untersuchen ist und zusätzlich die zu untersuchenden Intervalle in diesem Bereich.

I ist zu untersuchen im Bereich von $0 < I < kgV(100ms, 200ms, 250ms) = 1000ms$; (da $t_A = 0$)
I_E: {100ms, 200ms, 300ms, 400ms, 500ms, 600ms, 700ms, 800ms, 900ms}
I_V: {200ms, 400ms, 600ms, 800ms}
I_A: {220ms, 470ms, 720ms, 970ms}
I: {100, 200, 220, 300, 400, 470, 500, 600, 700, 720, 800, 900, 970}

c. Führen Sie den Echtzeitnachweis durch. Ist schritthaltende Verarbeitung möglich?

$$t_{C,ges}(I) = \lfloor \frac{I}{100ms} \rfloor \cdot 30ms + \lfloor \frac{Ims}{200ms} \rfloor \cdot 60ms + \lfloor \frac{I+30ms}{250ms} \rfloor \cdot 90ms$$

$$t_{C,ges}(100ms) = \lfloor \frac{100ms}{100ms} \rfloor \cdot 30ms + \lfloor \frac{100ms}{200ms} \rfloor \cdot 60ms + \lfloor \frac{130ms}{250ms} \rfloor \cdot 90ms = 30ms$$
$$t_{C,ges}(200ms) = \lfloor \frac{200ms}{100ms} \rfloor \cdot 30ms + \lfloor \frac{200ms}{200ms} \rfloor \cdot 60ms + \lfloor \frac{230ms}{250ms} \rfloor \cdot 90ms = 120ms$$
$$t_{C,ges}(220ms) = \lfloor \frac{220ms}{100ms} \rfloor \cdot 30ms + \lfloor \frac{220ms}{200ms} \rfloor \cdot 60ms + \lfloor \frac{250ms}{250ms} \rfloor \cdot 90ms = 210ms$$
$$t_{C,ges}(300ms) = \lfloor \frac{300ms}{100ms} \rfloor \cdot 30ms + \lfloor \frac{300ms}{200ms} \rfloor \cdot 60ms + \lfloor \frac{330ms}{250ms} \rfloor \cdot 90ms = 240ms$$
$$t_{C,ges}(400ms) = \lfloor \frac{400ms}{100ms} \rfloor \cdot 30ms + \lfloor \frac{400ms}{200ms} \rfloor \cdot 60ms + \lfloor \frac{430ms}{250ms} \rfloor \cdot 90ms = 330ms$$
$$t_{C,ges}(470ms) = \lfloor \frac{470ms}{100ms} \rfloor \cdot 30ms + \lfloor \frac{470ms}{200ms} \rfloor \cdot 60ms + \lfloor \frac{500ms}{250ms} \rfloor \cdot 90ms = 420ms$$
$$t_{C,ges}(500ms) = \lfloor \frac{500ms}{100ms} \rfloor \cdot 30ms + \lfloor \frac{500ms}{200ms} \rfloor \cdot 60ms + \lfloor \frac{530ms}{250ms} \rfloor \cdot 90ms = 450ms$$
$$t_{C,ges}(600ms) = \lfloor \frac{600ms}{100ms} \rfloor \cdot 30ms + \lfloor \frac{600ms}{200ms} \rfloor \cdot 60ms + \lfloor \frac{630ms}{250ms} \rfloor \cdot 90ms = 540ms$$
$$t_{C,ges}(700ms) = \lfloor \frac{700ms}{100ms} \rfloor \cdot 30ms + \lfloor \frac{700ms}{200ms} \rfloor \cdot 60ms + \lfloor \frac{730ms}{250ms} \rfloor \cdot 90ms = 570ms$$
$$t_{C,ges}(720ms) = \lfloor \frac{720ms}{100ms} \rfloor \cdot 30ms + \lfloor \frac{720ms}{200ms} \rfloor \cdot 60ms + \lfloor \frac{750ms}{250ms} \rfloor \cdot 90ms = 660ms$$
$$t_{C,ges}(800ms) = \lfloor \frac{800ms}{100ms} \rfloor \cdot 30ms + \lfloor \frac{800ms}{200ms} \rfloor \cdot 60ms + \lfloor \frac{830ms}{250ms} \rfloor \cdot 90ms = 750ms$$
$$t_{C,ges}(900ms) = \lfloor \frac{900ms}{100ms} \rfloor \cdot 30ms + \lfloor \frac{900ms}{200ms} \rfloor \cdot 60ms + \lfloor \frac{930ms}{250ms} \rfloor \cdot 90ms = 780ms$$
$$t_{C,ges}(970ms) = \lfloor \frac{970ms}{100ms} \rfloor \cdot 30ms + \lfloor \frac{970ms}{200ms} \rfloor \cdot 60ms + \lfloor \frac{1000ms}{250ms} \rfloor \cdot 90ms = 870ms$$

Da die Bedingung $t_{C,ges}(I) \leq I$ für alle zu untersuchenden I erfüllt ist, ist schritthaltende Verarbeitung möglich.

3.2.2. Aufgabe 2

Die Funktion `int clock_nanosleep(clockid_t clock_id, int flags, const struct timespec *request, struct timespec *remain);` wird verwendet, um den aufrufenden Job für eine definierte Zeit in den Zustand

schlafend zu versetzen. Sie bekommt die Zeit, die geschlafen werden soll, über eine Datenstruktur vom Typ `struct timespec` in Sekunden und Nanosekunden übergeben:

```
struct timespec {
    time_t tv_sec;
    long tv_nsec;
};
```

1. Was ist der essentielle Unterschied zwischen der Funktion `nanosleep()` und der Funktion `clock_nanosleep()`? (2P)

`clock_nanosleep()` ermöglicht die Spezifikation des Zeitgebers und des absoluten Schlafens.

2. Erläutern Sie die Bedeutung der Funktionsparameter `nanosleep( struct timespec *request, struct timespec *remaining)`. (2P)

request: Zeit, die geschlafen werden soll.
remaining: Zeit, die nach einer (vorzeitigen) Unterbrechung noch zu schlafen ist.

3. Der Parameter `clock_id` spezifiziert den Zeitgeber. Welche Zeitgeber haben Sie in der Vorlesung kennen gelernt? Welches Verhalten wird durch den Zeitgeber beeinflusst? (3P)

CLOCK_REALTIME und CLOCK_MONOTONIC.

Das Verhalten der Funktion im Fall von Änderungen an der Systemuhr: CLOCK_REALTIME bedeutet, dass die Funktion auf Änderungen an der Systemuhr reagiert. Im Fall des Zeitgebers CLOCK_MONOTONIC werden Änderungen an der Systemuhr für die Durchführung des Zeitauftrags ignoriert.

4. Der Parameter `flags` spezifiziert, ob relativ oder absolut geschlafen wird. Was versteht man unter absolutem und was unter relativem Schlafen? (2P)

Absolutes Schlafen: Die Task soll zu einem definierten **Zeitpunkt** aufgeweckt werden (z.B. schlafe **bis** 7:12 Uhr).

Relatives Schlafen: Die Task soll für die spezifizierte **Zeitdauer** schlafen (schlafe **für** 12 Sekunden).

5. Warum ist das *absolute Schlafen* professioneller als das *relative Schlafen*? (1P)

Im Falle einer Unterbrechung ist absolutes Schlafen genauer.

6. Skizzieren Sie ein Codefragment, welches einen Job mithilfe von `clock_nanosleep()` für 1 Stunde, 2 Minuten, 3 Sekunden, 4 Millisekunden und 5 Mikrosekunden schlafen legt. Verwenden Sie das relative Schlafen! (6P)

```
struct timespec request;
struct timespec remaining;
request.tv_sec = (1*3600)+(2*60)+3;     // 3723 Sekunden
request.tv_nsec= (4*1000000)+(5*1000);  // 4005000 Nanosekunden

if (clock_nanosleep( CLOCK_MONOTONIC, 0, &request, &remaining)) {
    perror( "clock_nanosleep" );
```

```
    return -1;
}
```

3.2.3. Aufgabe 3

Zur Steuerung eines Quadrocopters synchronisieren sich zwei Jobs mit Hilfe von drei Semaphoren: S1 signalisiert, dass die Motordaten vorliegen, S2 signalisiert, dass die Ausgabe jetzt erfolgen soll und S3 signalisiert, dass die Ausgabe erfolgt ist.

Im Folgenden soll untersucht werden, ob die entwickelte Synchronisation deadlock-gefährdet ist oder nicht. Die die Synchronisation betreffenden Codeteile sehen folgendermaßen aus:

Thread V (Berechnung)

```
while( 1 ) {
   Berechne_Motordaten();
   V(S1);
   Warte_auf_Sync-Zeitpunkt();
   V(S2);
   ...
   P(S3);
}
```

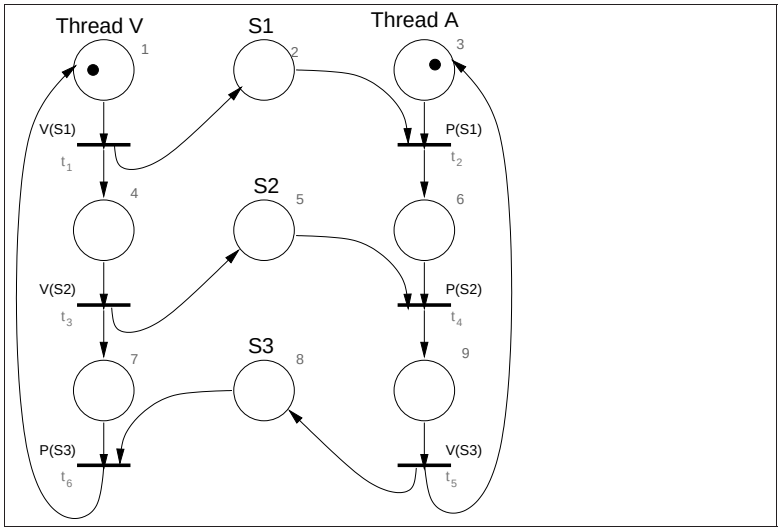
Thread A (Ausgabe)

```
while( 1 ) {
   P(S1);
   P(S2);
   Motordaten_ausgeben();
   V(S3);
}
```

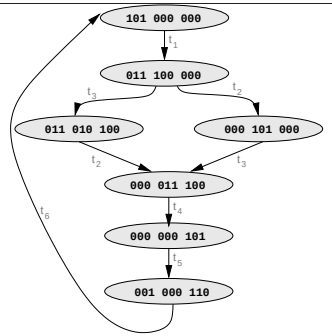
1. Warum müssen die Semaphore S1, S2 und S3 nicht wie gewohnt mit 1, sondern in diesem Fall mit 0 vorinitialisiert werden? (1P)

   Die Threads geben sonst unreservierte Semaphore frei, da beispielsweise V ohne zu warten nach dem Start die V-Operation ausführt.

2. Modellieren Sie die Synchronisation der *beiden Jobs* über die *drei Semaphore* mit Hilfe eines Petrinetzes. Vervollständigen Sie dazu den aus 9 Stellen vorliegenden Stellenplan. Beschriften Sie die Plätze und Transitionen und geben Sie eine sinnvolle Anfangsmarkierung an. (7P)



3. Stellen Sie den zugehörigen Erreichbarkeitsgraphen auf. Kommt es zu einer Verklemmung? Achtung: Semaphore sind mit 0 vorinitialisiert! Begründen Sie Ihre Antwort! (10P)

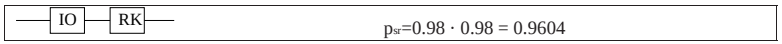


Da es im Erreichbarkeitsgraphen keinen Knoten ohne abgehende Kante gibt, ist das System nicht deadlockgefährdet.

3.2.4. Aufgabe 4

Der Steuerrechner einer Anlage besteht aus einer I/O-Kopplung und einem CPU-Board. Die I/O-Kopplung habe die Verfügbarkeit $p_{io}=0.98$, das CPU-Board ebenfalls $p_c=0.98$.

1. Zeichnen Sie das Verfügbarkeitsersatzschaltbild des Systems und geben Sie die Dauerverfügbarkeit p_{sr} des Gesamtsystems (Steuerrechner) an. (2P)



$p_{sr}=0.98 \cdot 0.98 = 0.9604$

2. Um die Verfügbarkeit zu erhöhen, soll der Steuerrechner redundant (zweifach) eingesetzt werden. Für die dazu notwendige Rechnerkopplung wird eine `MTBF=12000h` und eine `MTTR=2h` angegeben. (1,1,3=5P)

   a. Zeichnen Sie das Verfügbarkeitsersatzschaltbild.

   b. Berechnen Sie die Dauerverfügbarkeit der Rechnerkopplung.

$p_k=12000/12002=0.9998$

   c. Geben Sie die Verfügbarkeit des Gesamtsystems $p_{sys,A}$ an (Formel plus Ergebnis).

$p_{sys,A}= (1 - q_{sr2}) \cdot p_k = 0.99802$

3. Eine zweite Variante sieht vor, die Komponenten I/O-Kopplung und CPU-Board unabhängig voneinander zu verdoppeln. In diesem Fall ist für jede doppelte Komponente eine Rechnerkopplung notwendig.

   a. Zeichnen Sie das Verfügbarkeitsersatzschaltbild. (1P)

   b. Geben Sie die Verfügbarkeit des Gesamtsystems $p_{sys,B}$ an (Formel plus Ergebnis). (3P)

$p_{sys,B}= (1 - q_{io2}) (1 - q_{c2}) \cdot p_{k2} = 0.9988$

Kapitel 4. Klausur Realzeitsysteme WS2011/2012

4.1. Hinweise zum Prüfungsablauf

Tag der Prüfung	14.02.2012
Beginn	9:00 Uhr
Dauer	120 Minuten
Ort	Audimax

Die Prüfung besteht aus einem Fragenteil ohne Unterlagen und einem Aufgabenteil mit Unterlagen.

Die Bearbeitungszeit für den Fragenteil, zu dem keinerlei Hilfsmittel und Unterlagen zugelassen sind, beträgt 30 Minuten. Beantworten Sie die Fragen auf einem eigenen Prüfungsbogen. Sobald Sie mit der Bearbeitung des ersten Prüfungsteiles fertig sind, können Sie mit dem zweiten Teil fortfahren. Unterlagen dürfen allerdings erst verwendet werden, nachdem sämtliche Lösungsblätter des ersten Teiles eingesammelt worden sind.

Der zweite Teil der Prüfung besteht aus insgesamt 3 Aufgaben. Die einzelnen Aufgaben und größtenteils auch die einzelnen Teilaufgaben sind unabhängig voneinander lösbar. Verwenden Sie bitte für jede Aufgabe ein eigenes Lösungsblatt. Die für diesen Teil zur Verfügung stehende Bearbeitungszeit beträgt ingesamt 90 Minuten.

Bitte legen Sie am Ende der Prüfungszeit alle Lösungblätter ineinander.

Die über den Aufgaben stehende Punktzahl ist vorläufig und unverbindlich.

Achtung

Versehen Sie unbedingt alle Prüfungsbögen und Papiere, die Sie mit abgeben, mit Ihrem Namen und Ihrer Matrikelnummer! Geben Sie auf jeden Fall alle Prüfungsbögen ab, auch wenn Sie eine Aufgabe nicht bearbeitet haben.

Viel Erfolg!

4.2. Ohne Unterlagen

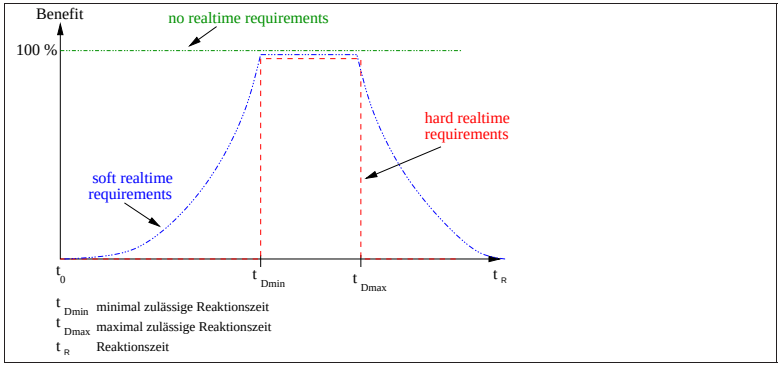
1. Definieren Sie den Begriff »Realzeitsystem«. (1P)

Ein Realzeitsystem muss neben den funktionalen Anforderungen auch noch zeitlichen Anforderungen genügen.

2. Wie lauten die beiden Realzeitbedingungen (genaue Angabe)? (2P)

1. RZB	$\rho = \sum_{i=0}^{n} \frac{t_v}{t_{ri}} \leq n;$ mit n=Anzahl Rechnerkerne
2. RZB	Für alle Rechenzeitanforderungen *i* muss gelten: $t_{Dmin,i} \leq t_{Rmin,i} \leq t_{Rmax,i} \leq t_{Dmax,i}$

3. Erläutern Sie anhand der Nutzen-Funktion mithilfe einer **genauen** Skizze den Unterschied zwischen harter und weicher Realzeit! (2P)



4. Welche drei Realzeit-Architekturen haben Sie im Rahmen der Vorlesung kennengelernt, um Systeme mit harten Zeitanforderungen zu realisieren? (3P)

Zeitkritische Teile der Applikation in den Kernel verlagern.
Multikern-Ansatz: Einzelen Prozessorkerne werden für Realzeitaufgaben reserviert.
Multikernel-Ansatz: Ein Standardbetriebssystem (z.B. Linux) wird als Task eines Echtzeitkernels abgearbeitet.

5. Bitoperationen
   a. Mit welchem logischen Operator können Sie einzelne Bits eines Bitfeldes testen? (1,2,2=5P)

Mit einer bitweisen UND-Verknüpfung („&”).

   b. Es soll überprüft werden, ob das 1. **oder** das 4. Bit (Wertigkeit 2^0 und 2^3) einer 16-Bit Variablen

gesetzt sind. Geben Sie die zugehörige Maske als Hexwert an.

0x0009

   c. Skizzieren Sie jetzt das Codefragment, das eine Meldung ausgibt, ob eines der beiden Bits (1. oder 4.) in der Variablen `tasten` gesetzt ist oder nicht.

```
if ( (tasten&0x0009)==0 ) {
    printf("Kein Bit gesetzt.\n");
} else {
    printf("Mindestens ein Bit gesetzt.\n");
}
```

6. Moderne Realzeit-Betriebssysteme ermöglichen Funktionen wie beispielsweise `clock_nansoleep()` Zeitgeber vom Typ CLOCK_MONOTONIC oder CLOCK_REALTIME zu verwenden. (1,2,2=5P)
   a. In welcher Situation kommt die Art des Zeitgebers zum Tragen?

Modifikationen an der Systemuhr.

   b. Welches Verhalten zeigt die Funktion `clock_nanosleep()` beim Zeitgeber CLOCK_MONOTONIC, welches bei CLOCK_REALTIME?

CLOCK_MONOTONIC: Modifikationen an der Systemuhr bleiben unberücksichtigt. CLOCK_REALTIME: Modifikationen an der Systemuhr werden berücksichtigt.

   c. Ein Thread legt sich mithilfe eines Zeitgebers vom Typ CLOCK_REALTIME für 10 Sekunden schlafen. Zwei Sekunden später wird die Systemuhr um 30 Sekunden vorgestellt. Wie verhält sich der schlafende Thread?

Der schlafende Thread wird sofort aufgeweckt.

7. Gegeben sind die folgenden Adressen im Segment/Offset-Format: `0x394A:5BCA` und `0xCC1A:0x28B1`. (2,1,1,1=5P)
   a. Welche physikalischen Adressen werden angesprochen?

`0x394A0+0x5BCA=0x3F06A` `0xCC1A0+0x28B1=0xCEA5A`

   b. Welches Problem gibt es, wenn Adressen im Segment-/Offset-Format verarbeitet werden?

Adressvergleiche sind aufgrund des durch den Compiler verwendeten Algorithmus kritisch: Dieselbe physikalische Adresse kann durch unterschiedliche logische Adressen dargestellt werden.

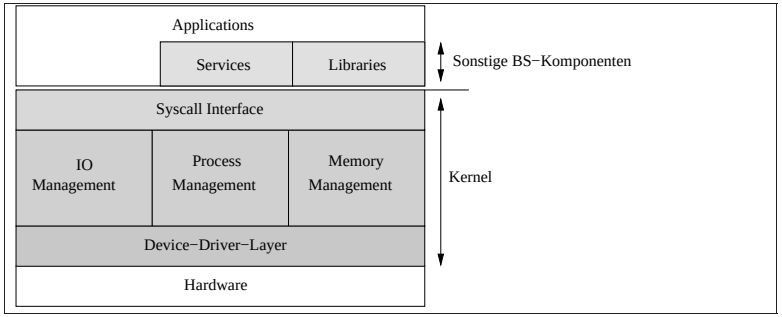
   c. Geben Sie für jede der beiden Adressen die normalisierte Darstellung an.

```
0x3F06:0x000A       0xCEA5:0x000A
```

d. Wie viele Segmentregister gibt es in einer CPU typischerweise?

Vier: DS (Datensegment), CS (Codesegment), ES (Extra-Datensegment), SS (Stacksegment)

8. Skizzieren Sie die in der Vorlesung vorgestellte Architektur eines (Realzeit-) Betriebssystems. (3P)



9. Skizzieren Sie den Code einer **Funktion** `int digital_out( unsigned int value );`, mit der Sie den übergebenen Wert `value` auf das Gerät `/dev/digital_out` ausgeben. Im Fehlerfall gibt die Funktion »-1«, ansonsten »0« zurück. (4P)

```
int digital_out( unsigned int value )
{
    int fd;

    fd = open( "/dev/digital_out", O_WRONLY );
    if (fd<0) {
        return -1;
    }
    if (write(fd,&value,sizeof(value))≤0) {
        close( fd );
        return -1;
    }
    close( fd );
    return 0;
}
```

4.3. Mit Unterlagen

4.3.1. Aufgabe 1

Aufgrund der Systemspezifikation und aufgrund von Messungen konnten die auf einer Steuerung laufenden Rechenprozesse und einige der zugehörigen zeitlichen Eckdaten evaluiert werden:

Job	$t_{Pmin,i}$	$t_{Dmin,i}$	$t_{Dmax,i}$	$t_{A,i}$	$t_{Emin,i}$	$t_{Emax,i}$	$t_{Rmin,i}$	$\rho_{max,i}$
V	250 ms	60 ms	250 ms	0 ms	40 ms	75 ms	40 ms	
E	500 ms	40 ms	360 ms	0 ms	40 ms		90 ms	0,35
A	200 ms	20 ms	160 ms	0 ms	20 ms	36 ms	30 ms	

1. Leiten Sie die fehlenden Kenndaten her: (5P)
   a. $\rho_{max,A}$ (Job A)
   b. $\rho_{max,V}$ (Job V)
   c. $t_{Emax,E}$ (Job E)
   d. ρ_{ges} (Gesamtauslastung)
   e. Ist eine schritthaltende Verarbeitung prinzipiell möglich?

2. Verteilen Sie die Prioritäten. (1P)
3. Echtzeitnachweis bei Einsatz eines prioritätengesteuerten Schedulings. (1,1,1,1,1,2,3=10P)
   a. Geben Sie die Rechenzeitanforderungsfunktion $t_{C,1}(t)$ für die Jobs mit Priorität 1 an.
   b. Berechnen Sie die maximale Reaktionszeit $t_{Rmax,1}$ für die zugehörigen Jobs.
   c. Geben Sie die Rechenzeitanforderungsfunktion $t_{C,2}(t)$ für die Jobs mit Priorität 2 an.
   d. Berechnen Sie die maximale Reaktionszeit $t_{Rmax,2}$ für die zugehörigen Jobs.
   e. Geben Sie die Rechenzeitanforderungsfunktion $t_{C,3}(t)$ für die Jobs mit Priorität 3 an.
   f. Berechnen Sie die maximale Reaktionszeit $t_{Rmax,3}$ für die zugehörigen Jobs.
   g. Zeigen Sie anhand der 2. Echtzeitbedingung (Rechtzeitigkeitsbedingung), ob schritthaltende Verarbeitung für sämtliche Jobs möglich ist oder nicht.

*Jetzt sollen mit dem Gerät **ZWEI** technische Prozesse gesteuert werden. Dazu wird die CPU der eingebetteten Steuerung gegen ein Modell eingetauscht, das eine doppelte Rechenleistung mit sich bringt.*

4. Berechnung der neuen Zeitparameter (3,1=4P)
   a. Welche neuen maximalen Verarbeitungszeiten $t_{Emax,i}$ ergeben sich für die drei Rechenprozesse?
   b. Verändern sich auch die Prozesszeiten (Begründung)?

5. Echtzeitnachweis bei Einsatz eines Deadline-Schedulings. Verwenden Sie die folgenden Verarbeitungszeiten: $t_{Emax,V}$=40ms, $t_{Emax,E}$=100ms, $t_{Emax,A}$=20ms. Beachten Sie, dass neben den veränderten Verarbeitungszeiten alle Prozesse **zweifach** vorkommen (2x Job V, 2x Job E und 2x Job A)! (3,4,9=16P)
   a. Geben Sie die Gesamt-Rechenzeitanforderungsfunktion $t_{C,ges}(I)$ an.
   b. Für welche I müssen Sie $t_{C,ges}(I)$ konkret untersuchen? Geben Sie den Bereich an, in dem I zu untersuchen ist und zusätzlich die zu untersuchenden Intervalle in diesem Bereich.
   c. Führen Sie den Echtzeitnachweis durch. Ist schritthaltende Verarbeitung möglich?

Lösung

1.

   a. $\rho_{max,A}$=36ms/200ms=0,18

b. $\rho_{max,V}$=75ms/250ms=0,3
c. $t_{Emax,E} = \rho_{max,E}\ t_{P,E}$ = 0,35 * 500ms = 175ms
d. ρ_{ges}=0.18+0.3+0.35=0.83
e. Ja, schritthaltende Verarbeitung ist prinzipiell möglich.

2. A=1 (hohe Priorität), V=2, E=3 (niedrige Priorität)
3.
a)

$$t_{c,1}(t) = \lceil \frac{t}{200ms} \rceil \cdot 36ms$$

b)

$$t_{c,1}^{(1)} = 36ms$$

$$t_{c,1}^{(2)} = \lceil \frac{36ms}{200ms} \rceil \cdot 36ms = 36ms$$

$$t_{Rmax,1} = 36ms$$

c)

$$t_{c,2}(t) = \lceil \frac{t}{200ms} \rceil \cdot 36ms + \lceil \frac{t}{250ms} \rceil \cdot 75ms$$

d)

$$t_{c,2}^{(1)} = 111ms$$

$$t_{c,2}^{(2)} = \lceil \frac{111ms}{200ms} \rceil \cdot 36ms + \lceil \frac{111ms}{250ms} \rceil \cdot 75ms = 111ms$$

$$t_{Rmax,2} = 111ms$$

e)

$$t_{c,3}(t) = \lceil \frac{t}{200ms} \rceil \cdot 36ms + \lceil \frac{t}{250ms} \rceil \cdot 75ms + \lceil \frac{t}{500ms} \rceil \cdot 175ms$$

f)

$$t_{c,3}^{(1)} = 286ms$$

$$t_{c,3}^{(2)} = \lceil \frac{286ms}{200ms} \rceil \cdot 36ms + \lceil \frac{286ms}{250ms} \rceil \cdot 75ms + \lceil \frac{286ms}{500ms} \rceil \cdot 175ms = 397ms$$

$$t_{c,3}^{(3)} = \lceil \frac{397ms}{200ms} \rceil \cdot 36ms + \lceil \frac{397ms}{250ms} \rceil \cdot 75ms + \lceil \frac{397ms}{500ms} \rceil \cdot 175ms = 397ms$$

$$t_{Rmax,3} = 397ms$$

g)

$t_{Dmin} \leq t_{Rmin} \leq t_{Rmax} \leq t_{Dmax}$

V: 60ms $\leq$ 40ms $\leq$ 111ms $\leq$ 250ms ---> Bedingung nicht erfüllt.

E: 40ms $\leq$ 90ms $\leq$ 397 $\leq$ 360ms ---> Bedingung nicht erfüllt.

A: 20ms $\leq$ 30ms $\leq$ 36ms $\leq$ 160ms ---> Bedingung erfüllt.

Schritthaltende Verarbeitung ist nicht möglich, da Job V zu früh und Job E unter Umständen zu spät reagiert.

4.
a. $t_{Eneu,V}=t_{Emax,V}/2$ = 75ms/2 = 37.5ms

$t_{Eneu,E}=t_{Emax,E}/2$ = 175ms/2 = 87.5ms

$t_{Eneu,A}=t_{Emax,A}/2$ = 36ms/2 = 18ms

b. Nein, bei Prozesszeiten handelt es sich um Zeiten des Technischen Prozesses; sie sind von außen vorgegeben.

5.
a)

$$t_{C,ges}(I) = \sum \lfloor \frac{I + t_{Pmin,i} - t_{Dmax,i} - t_{A,i}}{t_{P,i}} \rfloor \cdot t_{Emax,i}$$

$$t_{C,ges}(I) = 2 \cdot \lfloor \frac{I - 250ms + 250ms + 0ms}{250ms} \rfloor \cdot 40ms + 2 \cdot \lfloor \frac{I - 360ms + 500ms}{500ms} \rfloor \cdot 100ms + 2 \cdot \lfloor \frac{I - 1600ms + 200ms}{200ms} \rfloor \cdot 20ms$$

$$t_{C,ges}(I) = \lfloor \frac{I}{250ms} \rfloor \cdot 80ms + \lfloor \frac{I + 140ms}{500ms} \rfloor \cdot 200ms + \lfloor \frac{I + 40}{200ms} \rfloor \cdot 40ms$$

b)

Ist zu untersuchen im Bereich von $0 < I < kgV(200ms, 500ms, 250ms) = 1000ms$; (da $t_A = 0$)

I_V: {250ms, 500ms, 750ms}
I_E: {360ms, 860ms}
I_A: {160ms, 360ms, 560ms, 760ms, 960ms}
I: {160ms, 250ms, 360ms, 500ms, 560ms, 750ms, 760ms, 860ms, 960ms}

c)

$$t_{C,ges}(I) = \lfloor \frac{I}{250ms} \rfloor \cdot 80ms + \lfloor \frac{I + 140ms}{500ms} \rfloor \cdot 200ms + \lfloor \frac{I + 40ms}{200ms} \rfloor \cdot 40ms$$

$$t_{C,ges}(160ms) = \lfloor \frac{160ms}{250ms} \rfloor \cdot 80ms + \lfloor \frac{300ms}{500ms} \rfloor \cdot 200ms + \lfloor \frac{200ms}{200ms} \rfloor \cdot 40ms = 40ms$$

$$t_{C,ges}(250ms) = \lfloor \frac{250ms}{250ms} \rfloor \cdot 80ms + \lfloor \frac{340ms}{500ms} \rfloor \cdot 200ms + \lfloor \frac{290ms}{200ms} \rfloor \cdot 40ms = 120ms$$

$$t_{C,ges}(360ms) = \lfloor \frac{360ms}{250ms} \rfloor \cdot 80ms + \lfloor \frac{500ms}{500ms} \rfloor \cdot 200ms + \lfloor \frac{400ms}{200ms} \rfloor \cdot 40ms = 360ms$$

$$t_{C,ges}(500ms) = \lfloor \frac{500ms}{250ms} \rfloor \cdot 80ms + \lfloor \frac{640ms}{500ms} \rfloor \cdot 200ms + \lfloor \frac{540ms}{200ms} \rfloor \cdot 40ms = 440ms$$

$$t_{C,ges}(560ms) = \lfloor \frac{560ms}{250ms} \rfloor \cdot 80ms + \lfloor \frac{700ms}{500ms} \rfloor \cdot 200ms + \lfloor \frac{600ms}{200ms} \rfloor \cdot 40ms = 480ms$$

$$t_{C,ges}(750ms) = \lfloor \frac{750ms}{250ms} \rfloor \cdot 80ms + \lfloor \frac{890ms}{500ms} \rfloor \cdot 200ms + \lfloor \frac{790ms}{200ms} \rfloor \cdot 40ms = 560ms$$

$$t_{C,ges}(760ms) = \lfloor \frac{760ms}{250ms} \rfloor \cdot 80ms + \lfloor \frac{900ms}{500ms} \rfloor \cdot 200ms + \lfloor \frac{800ms}{200ms} \rfloor \cdot 40ms = 600ms$$

$$t_{C,ges}(860ms) = \lfloor \frac{860ms}{250ms} \rfloor \cdot 80ms + \lfloor \frac{1000ms}{500ms} \rfloor \cdot 200ms + \lfloor \frac{900ms}{200ms} \rfloor \cdot 40ms = 800ms$$

$$t_{C,ges}(960ms) = \lfloor \frac{960ms}{250ms} \rfloor \cdot 80ms + \lfloor \frac{1100ms}{500ms} \rfloor \cdot 200ms + \lfloor \frac{1000ms}{200ms} \rfloor \cdot 40ms = 840ms$$

Da die Bedingung $t_{C,ges}(I) \leq I$ für alle zu untersuchenden I erfüllt ist, ist schritthaltende Verarbeitung möglich.

4.3.2. Aufgabe 2

Die Funktion `int nanosleep(struct timespec *req, struct timespec *rem)` wird verwendet, um den aufrufenden Job für eine definierte Zeit in den Zustand *schlafend* zu versetzen. Sie bekommt die Zeit, die geschlafen werden soll, über eine Datenstruktur vom Typ `struct timespec` in Sekunden und Nanosekunden übergeben:

```
struct timespec {
    time_t tv_sec;
    long tv_nsec;
};
```

Der Datentyp `time_t` sei 32 Bit breit und kann damit einen Wert von 0 bis 4294967295 annehmen.

1. Wie lang kann aufgrund der verwendeten Datenstruktur ein Job per einfachem Aufruf von `nanosleep()` maximal schlafen? (2P)
2. Die Funktion `nanosleep()` bekommt die beiden Parameter `req` und `rem` übergeben. Welche Bedeutung haben die Parameter? (2P)

3. `nanosleep()` bekommt die Schlafenszeit *normalisiert* übergeben. Was muss bei der Angabe des Nanosekundenanteils beachtet werden? (2P)
4. Skizzieren Sie ein Codefragment, welches einen Job mithilfe von `nanosleep()` für 4321ms schlafen legt. (4P)

Ein Programmierer führt folgende Berechnung mit Zeitvariablen vom Typ `struct timespec` durch:

```
to_sleep.tv_sec  = tv1.tv_sec + tv2.tv_sec;
to_sleep.tv_nsec = tv1.tv_nsec + tv2.tv_nsec;
```

Die beiden Zeitvariablen `tv1` und `tv2` haben die folgenden Werte:

tv1	tv_sec=2	tv_nsec=987654321
tv2	tv_sec=1	tv_nsec=300000000

5. Welchen Wert enthält die Variable `to_sleep` nach der oben beschriebenen Berechnung bei den gegebenen Werten? (2P)
6. Warum wird die Funktion `nanosleep()` nicht die erwartete Zeit schlafen, falls sie mit dem wie oben berechneten `to_sleep` als Parameter aufgerufen wird? (1P)
7. Geben Sie das normalisierte Ergebnis der Summe der beiden Zeitwerte `tv1` und `tv2` an. (2P)
8. Skizzieren Sie in Form eines **Struktogramms** einen Algorithmus, der die Summe der Zeitvariablen `to_sleep` korrigiert und in normalisierter Form ablegt. (4P)

Lösung

1. Zeitbereich: 4294967295 Sekunden und 999999999 Nanosekunden.
2. `req`: Dieser Parameter gibt die Zeit an, die geschlafen werden soll.

   `rem`: Nach Ablauf der Funktion enthält dieser Parameter die Restzeit, die noch geschlafen werden muss. Im Normalfall sind beide Teile von `rem` nach der Rückkehr der Funktion Null. Wird die Funktion aber beispielsweise durch ein Signal unterbrochen, enthält sie die Zeit, die noch zu schlafen übrig ist.
3. Der Nanosekundenanteil darf nicht größer als 999999999 werden, auch wenn die Variable als solches größere Werte aufnehmen kann.
4.
   ```
   struct tv_spec sleeptime, rem;

   sleeptime.tv_sec = 4;
   sleeptime.tv_nsec = 321000000;
   if (nanosleep( &sleeptime, &rem )) {
       printf("Schlafen wurde unterbrochen...\n");
   }
   ...
   ```
5.
   ```
   tv_sleep.tv_sec = 3;
   tv_sleep.tv_nsec = 1287654321;
   ```
6. Der Nanosekundenanteil ist größer als 999999999 Nanosekunden.
7.
   ```
   tv_sleep.tv_sec = 4;
   tv_sleep.tv_nsec = 287654321;
   ```
8. Sekunden- und Nanosekundenanteil des Zeitstempels lassen sich am einfachsten durch eine Division- und eine Modulo-Operation korrigieren:


```
Sekundenanteil korrigieren
tv_sleep.tv_sec = tv_sleep.tv_sec + (tv_sleep.tv_nsec/1000000000);
```

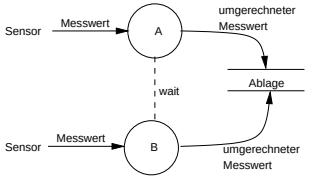
```
Nanosekundenanteil korrigieren
tv_sleep.tv_nsec = tv_sleep.tv_nsec % 1000000000;
```

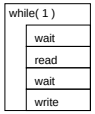
4.3.3. Aufgabe 3

Die Jobs A und B sollen im Endlosbetrieb von jeweils einem Sensor Messwerte aufnehmen (Schritt 1 »read«) und nach einer Umrechnung in einen gemeinsamen Speicher ablegen (Schritt 2 »write«). Da gefordert ist, dass die Jobs beide Arbeitsschritte möglichst gleichzeitig durchführen, synchronisieren sie sich (warten »wait«) vor jedem Arbeitsschritt.

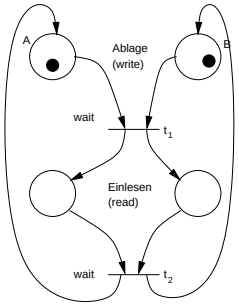
1. Skizzieren Sie das Datenflussdiagramm (DFD) der beschriebenen Konstellation. Vergessen Sie nicht die Beschriftung! (3P)
2. Zeichnen Sie das Struktogramm des Jobs A. (4P)
3. Zeichnen Sie das aus *vier* Stellen bestehende Petrinetz, das die Synchronisation (Rendezvous) zwischen Job A und Job B modelliert. ACHTUNG: Die parallele Verarbeitung der Arbeitsschritte 1 und 2 muss im Petrinetz erkennbar sein! Vergessen Sie nicht die Beschriftung von Stellen und Transitionen! (6P)
4. Deadlockuntersuchung (3P)
   a. Erstellen Sie den zugehörigen Erreichbarkeitsgraphen.
   b. Kann es zu einem Deadlock kommen? (Begründung)

Lösung

1.
   

2.
   

3.



4.

a.

b. Nein, jeder Knoten hat eine abgehende Kante.

4.3.4. Aufgabe 4

Der Steuerrechner eines Transportsystems soll für einen Zeitraum von ca. 5 Jahren (44000 h) “rund um die Uhr” betrieben werden. Er besteht aus den Komponenten Zentraleinheit und Peripherieankopplung, die beide die gleiche $MTBF_K$ besitzen. Der Hersteller nennt für den Rechner eine $MTBF_R$ von 20000h. Die vom Kundendienst benötigte Arbeitszeit zur Reparatur des Rechners sei t_R=4h. Folgende Varianten der Hardware-Wartung werden durch den Hersteller angeboten:

Keine Wartung

> Falls ein Defekt auftritt, wird der Rechner gegen Einzelrechnung repariert. Der Hersteller verspricht, dass der Kundendienst durchschnittlich innerhalb einer Woche (7 Tage) zur Stelle ist.

24-Stunden-Wartung

> Wenn ein Rechnerausfall bis spätestens 18:00 Uhr gemeldet wird, ist der Kundendienst am nächsten Tag um 8:00 Uhr zur Stelle, um die Reparatur vorzunehmen. Später als 18:00 Uhr gemeldete Ausfälle können erst am übernächsten Tag berücksichtigt werden.

1. Zeichnen Sie das Zuverlässigkeits-Ersatzschaltbild und berechnen Sie die $MTBF_K$ der Teilkomponenten. (3P)
2. Wie groß ist die Wahrscheinlichkeit p_1, dass innerhalb der 5 Jahre des Rechnerbetriebs keinerlei Ausfälle auftreten? (4P)
3. Berechnen Sie die Dauerverfügbarkeit p_2 des Rechners, wenn die Variante “Keine Wartung” gewählt wird (Genauigkeit 4 Stellen). (3P)
4. Wie groß ist bei der “24-Stunden-Wartung” die minimale und die maximale Ausfallzeit des Rechners? (4P)
5. Welche Dauerverfügbarkeit p_3 des Rechners ergibt sich für die Variante “24-Stunden-Wartung” (Genauigkeit 4 Stellen)? (3P)

Um geringere Ausfallzeiten zu erreichen und die sehr hohen Kosten für die Wartungsverträge zu sparen, wird überlegt, einen zweiten Rechner anzuschaffen. Für die Organisation der beiden Rechner wird folgende Strategie untersucht:

Beide Rechner werden parallel betrieben, so dass beim Ausfall eines Rechners der andere den technischen Prozess ohne Unterbrechung weiter steuern kann. Der defekte Rechner wird vom Hersteller repariert (siehe “Keine Wartung”) und nach der Reparatur wieder in Betrieb genommen.

6. Zeichnen Sie zunächst das Verfügbarkeitsersatzschaltbild. Berechnen Sie die Dauerverfügbarkeit p_4 des Systems für diese Variante auf vier Stellen genau. (3P)

Lösung

1.

$\lambda_R = 2 \cdot \lambda_K$

$MTBF_K = 40.000h$

2. “keinerlei Ausfälle” $\longrightarrow$ “keine Reparatur”: ρ=0

$p_1 = e^{-\lambda_R t_1}$

λ_R=1/$MTBF_R$; t_1=44000h

$p_1 = 0.1108$

3. $p_2 = MTBF_R / (MTTR_2 + MTBF_R)$

$MTTR_2 = 7\ Tage + t_R \longrightarrow MTTR_2 = 172h$

$p_2 = 0.9915$

4. Minimale Ausfallzeit: Meldung kurz vor 18:00 Uhr:

$TTR_{min} = 14h + 4h = 18h$

Maximale Ausfallzeit: Meldung kurz nach 18:00 Uhr:

$TTR_{max} = 14h + 24h + 4h = 42h$

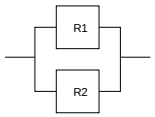
5. $p_3 = MTBF_R / (MTTR_3 + MTBF_R)$

$MTTR_3 = 0.5\ (TTR_{min} + TTR_{max})$

$MTTR_3 = 30h$

$p_3 = 0.9985$

6.



Parallelschaltung zweier Rechner:

$p_5 = 1 - (1-p_2)^2$

$p_5 = 0.9999$

Kapitel 5. Klausur Realzeitsysteme SS2011

5.1. Hinweise zum Prüfungsablauf

Tag der Prüfung	08.07.2011
Beginn	10:30 Uhr
Dauer	120 Minuten
Ort	Audimax

Die Prüfung besteht aus einem Fragenteil ohne Unterlagen und einem Aufgabenteil mit Unterlagen.

Die Bearbeitungszeit für den Fragenteil, zu dem keinerlei Hilfsmittel und Unterlagen zugelassen sind, beträgt 30 Minuten. Beantworten Sie die Fragen auf einem eigenen Prüfungsbogen. Sobald Sie mit der Bearbeitung des ersten Prüfungsteiles fertig sind, können Sie mit dem zweiten Teil fortfahren. Unterlagen dürfen allerdings erst verwendet werden, nachdem sämtliche Lösungsblätter des ersten Teiles eingesammelt worden sind.

Der zweite Teil der Prüfung besteht aus insgesamt 3 Aufgaben. Die einzelnen Aufgaben und größtenteils auch die einzelnen Teilaufgaben sind unabhängig voneinander lösbar. Verwenden Sie bitte für jede Aufgabe ein eigenes Lösungsblatt. Die für diesen Teil zur Verfügung stehende Bearbeitungszeit beträgt ingesamt 90 Minuten.

Bitte legen Sie am Ende der Prüfungszeit alle Lösungblätter ineinander.

Die über den Aufgaben stehende Punktzahl ist vorläufig und unverbindlich.

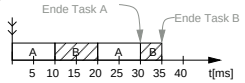
> **Achtung**
>
> Versehen Sie unbedingt alle Prüfungsbögen und Papiere, die Sie mit abgeben, mit Ihrem Namen und Ihrer Matrikelnummer! Geben Sie auf jeden Fall alle Prüfungsbögen ab, auch wenn Sie eine Aufgabe nicht bearbeitet haben.

Viel Erfolg!

5.2. Ohne Unterlagen

1. Definieren Sie den Begriff »Echtzeitsystem«. (1P)
2. Wie lauten die beiden Echtzeitbedingungen (Formeln!)? (1P)
3. Erläutern Sie anhand der Nutzen-Funktion den Unterschied zwischen harter und weicher Realzeit! (2P)

4. Skizzieren Sie den Code eines Proramms, mit dem Sie vom Sensor Feuchtigkeit (Gerätedatei `/dev/humidity`) einen Integerwert lesen und die gelesenen Daten auf dem Bildschirm per `printf()` ausgeben. (4P)
5. Latenzzeiten (1,1,2=4P)
   a. Was bezeichnet die Interrupt-Latenzzeit?
   b. Was bezeichnet die Prozess-Latenzzeit (Thread-Latenzzeit)?
   c.



   Geben Sie für die dargestellte Rechnerkernbelegung sowohl die Prozesslatenzzeit t_l als auch die Wartezeit t_w der beiden Rechenprozesse `A` und `B` an.

6. Die Steuerungssoftware einer Bewässerungsanlage besteht aus vier Threads: Thread *Feuchte* misst periodisch die Feuchtigkeit, Thread *Sonneneinstrahlung* die Lichtintensität und Thread *Temperatur* die Temperatur. Die jeweiligen Sensorwerte werden in den gemeinsamen Speicher `Umweltdaten` abgelegt. Der Thread *Steuerung* entnimmt die Umweltdaten und berechnet daraus Zeitpunkte, Zeitdauer und die notwendige Wassermenge. Die Ansteuerung eines Ventils auf Basis der berechneten Werte erfolgt ebenfalls in diesem Thread.
   a. Beschreiben Sie die »critical section«. Wie kann softwaretechnisch sichergestellt werden, dass es bei dieser »critical section« zu keiner *race condition* kommt? (2P)
   b. Skizzieren Sie das Datenflussdiagramm (DFD) der gegebenen Konstellation. Ergänzen Sie das DFD um den Kontrollfluss. Beschriftung nicht vergessen! (6P)

7. Gegeben sind in zwei unterschiedlichen Formaten die Adressen `0x4a3b:0x3a12` und `0x4b2ac`. (3,2,1,1,1,1,1=10P)
   a. Skizzieren Sie die in der Vorlesung vorgestellte Architektur eines Betriebssystems.
   b. Welche **vier** Aufgaben hat das Memory-Management?
   c. Geben Sie die Größe des maximalen Adressraumes auf Basis der gegebenen Adressen an.
   d. Wie nennen sich die zugrunde liegenden Adressierungsformate der Adressangaben?
   e. Formen Sie die Adressen in das jeweils andere Format um?
   f. Geben Sie beide Adressen in normalisierter Form an.
   g. Welche der beiden Adressen zeigt auf eine höhere Speicherzelle (Begründung)?

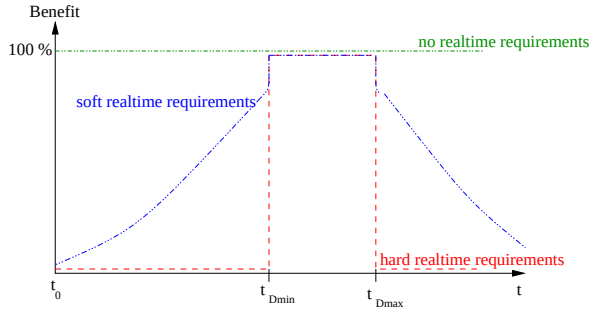
Lösung

1. Ein Echtzeitsystem muss neben den funktionalen Anforderungen auch noch zeitlichen Anforderungen genügen.
2. Auslastungsbedingung:
   $\rho = \sum_{i=0}^{n} \frac{t_v}{t_{p_i}} \leq 1$

   Forderung nach Pünktlichkeit, die für alle Ereignisse *i* erfüllt sein muss:

$t_{Dmin,i} \leq t_{Rmin,i} \leq t_{Rmax,i} \leq t_{Dmax,i}$

3. Die Realzeitanforderungen, die ein System stellt, lassen sich anhand der Nutzenfunktion klassifizieren:
   - Werden in Punkto Realzeit keinerlei Anforderungen gestellt, so ist der Nutzen unabhängig von der zeitlichen Lösung der Aufgabe immer gegeben.
   - Bei einer wirklich harten Realzeitanforderung muß, damit ein Nutzen vorhanden ist, innerhalb eines genau definierten Zeitfensters eine Reaktion auf ein Ereignis erfolgen. Das nicht Einhalten dieser Forderung führt zu hohen Kosten.
   - Man spricht von einer weichen Realzeit-Anforderung, wenn der Nutzen zwar von der Erfüllung der zeitlichen Anforderung abhängt, aber ein Nutzen auch dann gegeben ist, wenn die gegebenen Zeitschranken nicht einhunderprozentig eingehalten werden.




```
 4. #include <stdio.h>
#include <fcntl.h>
#include <unistd.h>

int main( inat argc, char **argv, char **envp )
{
    int fd;
    int humidity;

    fd = open( "/dev/humidity", O_RDONLY );
    if (fd<0) {
        printf( "error opening /dev/humidity\n" );
        return -1;
    }
    read( fd, &humidity, sizeof(humidity) );
    printf( "Feuchte: %d\n", humidity );
    return 0;
}
```

5.

a. Interrupt-Latenzzeit: Zeit zwischen dem Auftreten des Interrupts bis zum Start der zugehörigen Interrupt-Service-Routine.

b. Prozess-Latenzzeit: Zeit zwischen dem Auftreten des Ereignisses (Interrupt) bis zum Start des zugehörigen Rechenprozesses.

c. $t_{l,A} = 0ms$

$t_{l,B} = 10ms$

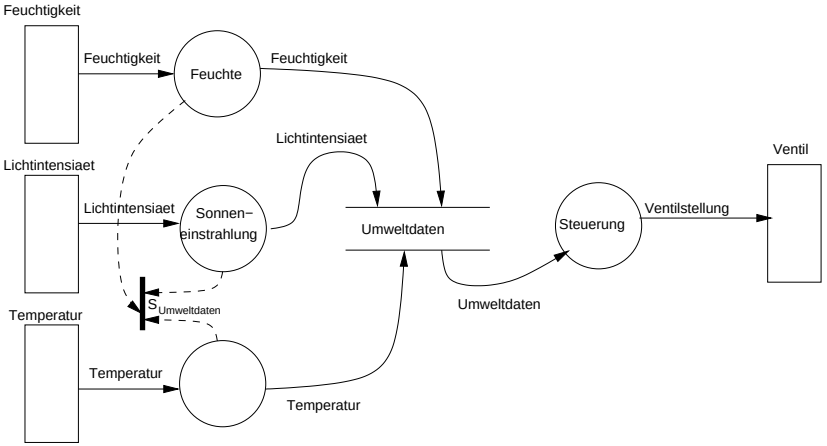
$t_{w,A} = 10ms$

$t_{w,B} = 20ms$

6.

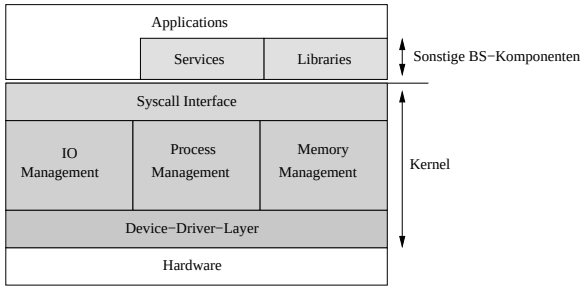
a. Der Zugriff auf den gemeinsamen Speicher *Umweltdaten* stellt einen kritischen Abschnitt dar. Dieser kann über ein Semaphor geschützt werden.

b. Daten- und Kontrollflußdiagramm



7.

a.



b. Adressumsetzung, Speicherschutz, virtuellen Speicher zur Verfügung stellen, erweiterten Speicher zur Verfügung stellen.

c. Die angegebenen Adressen werden durch fünf Nibbles, also 20 Bit repräsentiert. 2^{20}=1 048 576=1MByte

d. Adresse im Segment:Offset-Format, Physikalische Adresse.

e. `0x4a3b:0x3a12=0x4a3b0+0x3a12=0x4ddc2`

`4000:b2ac`

f. `4ddc:0002` und `4b2a:000c`

g. Die erste Adresse ist größer.

5.3. Mit Unterlagen

5.3.1. Aufgabe 1

Eine Fertigungseinrichtung wird mit Hilfe von vier Jobs automatisiert, deren Kenndaten sind:

Job	t_{Dmin}	t_{Dmax}	t_{Pmin}	t_A	t_{Emin}	t_{Emax}
A	20 ms	80 ms	80 ms	0 ms	20 ms	X
B	0 ms	140 ms	200 ms	0 ms	30 ms	Y
C	0 ms	40 ms	40 ms	0 ms	10 ms	10 ms
D	0 ms	100 ms	100 ms	0 ms	20 ms	20 ms

1. Berechnen Sie die folgenden Auslastungen im Worst Case: (2P)
   a. ρ_C (Job C)
   b. ρ_D (Job D)

2. Geben Sie die Formel für die Gesamtauslastung ρ_{ges}(X, Y) in Abhängigkeit von X und Y an. (2P)

3. Die maximale Verarbeitungszeit Y ($t_{Emax,B}$) des Jobs »B« ist dreimal so lang, wie die maximale Verarbeitungszeit X ($t_{Emax,A}$) des Jobs A. Bestimmen Sie unter dieser Randbedingung X und Y, so dass im Worst Case die Gesamtauslastung unter 100 Prozent bleibt. (4P)

Für die folgenden Teilaufgaben betrage X=20ms und Y=50ms.

4. Geben Sie den vier Rechenprozessen sinnvolle Prioritäten? (1P)
5. Echtzeitnachweis bei Einsatz eines prioritätengesteuerten Schedulings. (1,1,1,1,1,2,1,4,4=16P)
   a. Geben Sie die Rechenzeitanforderungsfunktion $t_{C,1}(t)$ für die Jobs mit Priorität 1 an.
   b. Berechnen Sie die maximale Reaktionszeit $t_{Rmax,1}$ für die zugehörigen Jobs.
   c. Geben Sie die Rechenzeitanforderungsfunktion $t_{C,2}(t)$ für die Jobs mit Priorität 2 an.
   d. Berechnen Sie die maximale Reaktionszeit $t_{Rmax,2}$ für die zugehörigen Jobs.
   e. Geben Sie die Rechenzeitanforderungsfunktion $t_{C,3}(t)$ für die Jobs mit Priorität 3 an.
   f. Berechnen Sie die maximale Reaktionszeit $t_{Rmax,3}$ für die zugehörigen Jobs.
   g. Geben Sie die Rechenzeitanforderungsfunktion $t_{C,4}(t)$ für die Jobs mit Priorität 4 an.
   h. Berechnen Sie die maximale Reaktionszeit $t_{Rmax,4}$ für die zugehörigen Jobs.
   i. Zeigen Sie anhand der 2. Echtzeitbedingung (Rechtzeitigkeitsbedingung), ob schritthaltende Verarbeitung für sämtliche Jobs möglich ist oder nicht.

6. Echtzeitnachweis bei Einsatz eines Deadline-Schedulers. (4,4,13=21P)
   a. Geben Sie die Gesamt-Rechenzeitanforderungsfunktion $t_{C,ges}(I)$ an.
   b. Für welche I müssen Sie $t_{C,ges}(I)$ konkret untersuchen? Geben Sie den Bereich an, in dem I zu untersuchen ist und zusätzlich die zu untersuchenden Intervalle in diesem Bereich.
   c. Führen Sie den Echtzeitnachweis durch. Ist schritthaltende Verarbeitung möglich?

Lösung

1.

   a. ρ_C=10ms/40ms=0,25
   b. ρ_D=20ms/100ms=0,2

2. ρ_{ges}=X/$t_{P,A}$ + Y/$t_{P,B}$ + ρ_C + ρ_D = X/80ms + Y/200ms + 0,45
3. Y = 3·X;

   ρ_{ges}=X/80ms + 3·X/200ms+0,45<1

   X(1/80ms + 3/200ms)<0,55

   X(11/400ms)<0,55

   X<0,55·400ms/11=20ms

   Y=3 · 20ms = 60ms
4. C=1,A=2,D=3,B=4
5. a)

$$t_{c,1}(t) = \lceil \frac{t}{40ms} \rceil \cdot 10ms$$

b)

$$t_{c,1}^{(1)} = 10ms$$

$$t_{c,1}^{(2)} = \lceil \frac{10ms}{80ms} \rceil \cdot 10ms = 10ms$$

$$t_{Rmax,1} = 10ms$$

c)

$$t_{c,2}(t) = \lceil \frac{t}{40ms} \rceil \cdot 10ms + \lceil \frac{t}{80ms} \rceil \cdot 20ms$$

d)

$$t_{c,2}^{(1)} = 30ms$$

$$t_{c,2}^{(2)} = \lceil \frac{30ms}{40ms} \rceil \cdot 10ms + \lceil \frac{20ms}{80ms} \rceil \cdot 20ms = 30ms$$

$$t_{Rmax,2} = 30ms$$

e)

$$t_{c,3}(t) = \lceil \frac{t}{40ms} \rceil \cdot 10ms + \lceil \frac{t}{80ms} \rceil \cdot 20ms + \lceil \frac{t}{100ms} \rceil \cdot 20ms$$

f)

$$t_{c,3}^{(1)} = 50ms$$

$$t_{c,3}^{(2)} = \lceil \frac{50ms}{40ms} \rceil \cdot 10ms + \lceil \frac{50ms}{80ms} \rceil \cdot 20ms + \lceil \frac{50ms}{100ms} \rceil \cdot 20ms = 60ms$$

$$t_{c,3}^{(3)} = \lceil \frac{60ms}{40ms} \rceil \cdot 10ms + \lceil \frac{60ms}{80ms} \rceil \cdot 20ms + \lceil \frac{60ms}{100ms} \rceil \cdot 20ms = 60ms$$

$$t_{Rmax,3} = 60ms$$

g)

$$t_{c,4}(t) = \lceil \frac{t}{40ms} \rceil \cdot 10ms + \lceil \frac{t}{80ms} \rceil \cdot 20ms + \lceil \frac{t}{100ms} \rceil \cdot 20ms + \lceil \frac{t}{200ms} \rceil \cdot 50ms$$

h)

$$t_{c,4}^{(1)} = 100ms$$

$$t_{c,4}^{(2)} = \lceil \frac{100ms}{40ms} \rceil \cdot 10ms + \lceil \frac{100ms}{80ms} \rceil \cdot 20ms + \lceil \frac{100ms}{100ms} \rceil \cdot 20ms + \lceil \frac{100}{200ms} \rceil \cdot 50ms = 140ms$$

$$t_{c,4}^{(3)} = \lceil \frac{140ms}{40ms} \rceil \cdot 10ms + \lceil \frac{140ms}{80ms} \rceil \cdot 20ms + \lceil \frac{140ms}{100ms} \rceil \cdot 20ms + \lceil \frac{140}{200ms} \rceil \cdot 50ms = 170ms$$

$$t_{c,4}^{(4)} = \lceil \frac{170ms}{40ms} \rceil \cdot 10ms + \lceil \frac{170ms}{80ms} \rceil \cdot 20ms + \lceil \frac{170ms}{100ms} \rceil \cdot 20ms + \lceil \frac{170}{200ms} \rceil \cdot 50ms = 200ms$$

$$t_{c,4}^{(5)} = \lceil \frac{200ms}{40ms} \rceil \cdot 10ms + \lceil \frac{200ms}{80ms} \rceil \cdot 20ms + \lceil \frac{200ms}{100ms} \rceil \cdot 20ms + \lceil \frac{200}{200ms} \rceil \cdot 50ms = 200ms$$

$$t_{Rmax,4} = 200ms$$

i)

A: $20ms \leq 20ms \leq 30ms \leq 80ms$ Ja, diese Bedingung ist erfüllt.

B: $0ms \leq 30ms \leq 200ms \leq 140ms$ Nein, diese Bedingung ist nicht erfüllt.

C: $0ms \leq 10ms \leq 10ms \leq 40ms$ Ja, diese Bedingung ist erfüllt.

D: $0ms \leq 20ms \leq 60ms \leq 100ms$ Ja, diese Bedingung ist erfüllt.

Schritthaltende Verarbeitung ist nicht möglich!

6.

a)

$$t_{C,ges}(I) = \sum \lfloor \frac{I + t_{Pmin,i} - t_{Dmax,i} - t_{A,i}}{t_{P,i}} \rfloor \cdot t_{Emax,i}$$

$$t_{C,ges}(I) = \lfloor \frac{I - 80ms + 80ms + 0ms}{80ms} \rfloor \cdot 20ms + \lfloor \frac{I - 140ms + 200ms}{200ms} \rfloor \cdot 50ms + \lfloor \frac{I - 40ms + 40ms}{40ms} \rfloor \cdot 10ms + \lfloor \frac{I - 100ms + 100ms}{100ms} \rfloor \cdot 20ms$$

$$t_{C,ges}(I) = \lfloor \frac{I}{80ms} \rfloor \cdot 20ms + \lfloor \frac{I + 60ms}{200ms} \rfloor \cdot 50ms + \lfloor \frac{I}{40ms} \rfloor \cdot 10ms + \lfloor \frac{I}{100ms} \rfloor \cdot 20ms$$

b)

I ist zu untersuchen im Bereich von $0 < I < kgV(80ms, 200ms, 40ms, 100ms) = 400ms$; (da $t_A = 0$)

I_A: 80ms, 160ms, 240ms, 320ms

I_B: 140ms, 340ms

I_C: 40ms, 80ms, 120ms, 160ms, 200ms, 240ms, 280ms, 320ms, 360ms

I_D: 100ms, 200ms, 300ms

I: 40ms, 80ms, 100ms, 120ms, 140ms, 160ms, 200ms, 240ms, 280ms, 300ms, 320ms, 340ms, 360ms

7.

c)

$$t_{C,ges}(I) = \lfloor \frac{I}{80ms} \rfloor \cdot 20ms + \lfloor \frac{I + 60}{200ms} \rfloor \cdot 50ms + \lfloor \frac{I}{40ms} \rfloor \cdot 10ms + \lfloor \frac{I}{100ms} \rfloor \cdot 20ms$$

$$t_{C,ges}(40ms) = \lfloor \frac{40ms}{80ms} \rfloor \cdot 20ms + \lfloor \frac{100ms}{200ms} \rfloor \cdot 50ms + \lfloor \frac{40ms}{40ms} \rfloor \cdot 10ms + \lfloor \frac{40ms}{100ms} \rfloor \cdot 20ms = 10ms$$

$$t_{C,ges}(80ms) = \lfloor \frac{80ms}{80ms} \rfloor \cdot 20ms + \lfloor \frac{140ms}{200ms} \rfloor \cdot 50ms + \lfloor \frac{80ms}{40ms} \rfloor \cdot 10ms + \lfloor \frac{80ms}{100ms} \rfloor \cdot 20ms = 40ms$$

$$t_{C,ges}(100ms) = \lfloor \frac{100ms}{80ms} \rfloor \cdot 20ms + \lfloor \frac{160ms}{200ms} \rfloor \cdot 50ms + \lfloor \frac{100ms}{40ms} \rfloor \cdot 10ms + \lfloor \frac{100ms}{100ms} \rfloor \cdot 20ms = 60ms$$

$$t_{C,ges}(120ms) = \lfloor \frac{120ms}{80ms} \rfloor \cdot 20ms + \lfloor \frac{180ms}{200ms} \rfloor \cdot 50ms + \lfloor \frac{120ms}{40ms} \rfloor \cdot 10ms + \lfloor \frac{120ms}{100ms} \rfloor \cdot 20ms = 70ms$$

$$t_{C,ges}(140ms) = \lfloor \frac{140ms}{80ms} \rfloor \cdot 20ms + \lfloor \frac{200ms}{200ms} \rfloor \cdot 50ms + \lfloor \frac{140ms}{40ms} \rfloor \cdot 10ms + \lfloor \frac{140ms}{100ms} \rfloor \cdot 20ms = 120ms$$

$$t_{C,ges}(160ms) = \lfloor \frac{160ms}{80ms} \rfloor \cdot 20ms + \lfloor \frac{220ms}{200ms} \rfloor \cdot 50ms + \lfloor \frac{160ms}{40ms} \rfloor \cdot 10ms + \lfloor \frac{160ms}{100ms} \rfloor \cdot 20ms = 150ms$$

$$t_{C,ges}(200ms) = \lfloor \frac{200ms}{80ms} \rfloor \cdot 20ms + \lfloor \frac{260ms}{200ms} \rfloor \cdot 50ms + \lfloor \frac{200ms}{40ms} \rfloor \cdot 10ms + \lfloor \frac{200ms}{100ms} \rfloor \cdot 20ms = 180ms$$

$$t_{C,ges}(240ms) = \lfloor \frac{240ms}{80ms} \rfloor \cdot 20ms + \lfloor \frac{300ms}{200ms} \rfloor \cdot 50ms + \lfloor \frac{240ms}{40ms} \rfloor \cdot 10ms + \lfloor \frac{240ms}{100ms} \rfloor \cdot 20ms = 210ms$$

$$t_{C,ges}(280ms) = \lfloor \frac{280ms}{80ms} \rfloor \cdot 20ms + \lfloor \frac{340ms}{200ms} \rfloor \cdot 50ms + \lfloor \frac{280ms}{40ms} \rfloor \cdot 10ms + \lfloor \frac{280ms}{100ms} \rfloor \cdot 20ms = 220ms$$

$$t_{C,ges}(300ms) = \lfloor \frac{300ms}{80ms} \rfloor \cdot 20ms + \lfloor \frac{360ms}{200ms} \rfloor \cdot 50ms + \lfloor \frac{300ms}{40ms} \rfloor \cdot 10ms + \lfloor \frac{300ms}{100ms} \rfloor \cdot 20ms = 240ms$$

$$t_{C,ges}(320ms) = \lfloor \frac{320ms}{80ms} \rfloor \cdot 20ms + \lfloor \frac{380ms}{200ms} \rfloor \cdot 50ms + \lfloor \frac{320ms}{40ms} \rfloor \cdot 10ms + \lfloor \frac{320ms}{100ms} \rfloor \cdot 20ms = 270ms$$

$$t_{C,ges}(340ms) = \lfloor \frac{340ms}{80ms} \rfloor \cdot 20ms + \lfloor \frac{400ms}{200ms} \rfloor \cdot 50ms + \lfloor \frac{340ms}{40ms} \rfloor \cdot 10ms + \lfloor \frac{340ms}{100ms} \rfloor \cdot 20ms = 320ms$$

$$t_{C,ges}(360ms) = \lfloor \frac{360ms}{80ms} \rfloor \cdot 20ms + \lfloor \frac{420ms}{200ms} \rfloor \cdot 50ms + \lfloor \frac{360ms}{40ms} \rfloor \cdot 10ms + \lfloor \frac{360ms}{100ms} \rfloor \cdot 20ms = 330ms$$

Da die Bedingung $t_{C,ges}(I) \leq I$ für alle zu untersuchenden I erfüllt ist, ist schritthaltende Verarbeitung möglich.

5.3.2. Aufgabe 2

Die Funktion `gettimeofday()` liefert die Zeit in Sekunden und Mikrosekunden über die folgende (vereinfacht dargestellte) Datenstruktur:

```
struct timeval {
    unsigned int tv_sec;
    unsigned int tv_usec;
};
```

Mit Hilfe der Funktion soll eine Differenzzeitmessung durchgeführt werden, die auch für größere Zeitabstände ausgelegt ist. Das Ergebnis soll wieder in Form der Datenstruktur `struct timeval` vorliegen (also in Sekunden und in Mikrosekunden). Hinweis: Der Datentyp `unsigned int` ist 32 Bit breit und kann damit einen Wert von 0 bis 4294967295 annehmen.

1. Welcher Zeitbereich lässt sich in einer derartigen Datenstruktur unterbringen? (2P)
2. Im Praktikum wurde die per `gettimeofday()` zurückgelieferte Zeit in eine einzelne Variable (Typ Integer) umgerechnet. Welchen Vorteil bringt das mit sich? Was sind die Nachteile? (2P)
3. Erläutern Sie das Prinzip der Differenzzeitmessung. (2P)
4. Geben Sie für die folgenden drei Paare von Zeitstempeln die jeweilige Differenz in Sekunden und Mikrosekunden an. (6P)

Vorher	tv_sec=12345	tv_usec=309111
Nachher	tv_sec=12347	tv_usec=309156

Vorher	tv_sec=12345	tv_usec=1300

Nachher	tv_sec=12347	tv_usec=1156

Vorher	tv_sec=4294967295	tv_usec=200
Nachher	tv_sec=1	tv_usec=100

5. Skizzieren Sie in Form eines **Struktogramms** einen Algorithmus, der die Differenzzeit berechnet und in die Variablen `diff_seconds` und `diff_useconds` ablegt. Der Algorithmus soll für alle angegebenen Zeitwerte das jeweils richtige Ergebnis liefern. (10P)

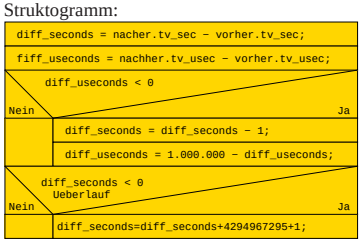
Lösung

1. Zeitbereich: 4294967295 Sekunden und 999999 Mikrosekunden.
2. Vorteil: Die Differenz zwischen zwei Zeitstempeln ist leicht durch eine einfache Subtraktion zu bestimmen.

   Nachteil: Das Verfahren berücksichtigt keine Zählerüberläufe. Die zu berechnenden Zeitdifferenz schöpfen nicht den möglichen Wertevorrat aus.
3. Bei der Differenzzeitmessung wird vor und nach dem auszumessenden Ereignis jeweils ein Zeitstempel genommen. Die beiden Zeitstempel werden voneinander subrahiert und spiegeln die bis dahin verstrichene Zeitdauer wider.
4. 2s, 45us [12347s-12345s=2s; 309156us-309111us=45us].

   1s, 999856us [12347s-12345s=2s; 1156us-1300us=-144us; da der Mikrosekundenanteil negativ ist, wird der Sekundenanteil um eine Sekunde reduziet und der Mikrosekundenanteil ergibt sich zu 1.000.000us-144us=999856us].

   1s, 999900us [1s-4294967295s=-4294967294s; 100us-200us=-100us; da der Sekundenanteil negativ ist, hat es einen Überlauf gegeben. Daher wird auf das Ergebnis der Wertebereich 4294967296 aufaddiert: 4294967296s+(-4294967294s)=2s. Da der Mikrosekundenanteil negativ ist, wird der Sekundenanteil dekrementiert und der Mikrosekundenanteil zu 1.000.000us-100us berechnet: 1.000.000us-100us=999.900us].
5. Struktogramm:



5.3.3. Aufgabe 3

In einem Steuerungssystem greifen zwei Threads A und B auf ein Betriebsmittel F1 zu. Folgende Abläufe sind bekannt:

Thread A:


```
- T1: Wenn der Thread aktiviert wird, belegt er das Betriebsmittel F1.
- T2: Der Thread gibt das Betriebsmittel wieder frei und ist danach
      wieder startklar.
```

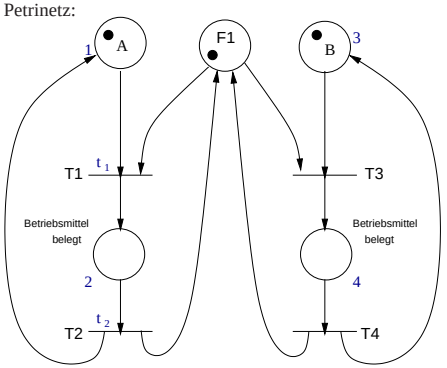
Thread B:

```
- T3: Wenn der Thread aktiviert wird, belegt er das Betriebsmittel F1.
- T4: Der Thread gibt das Betriebsmittel wieder frei und ist danach
      wieder startklar.
```

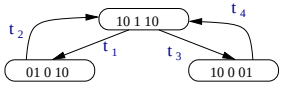
1. Skizzieren Sie den Zugriff der beiden Threads auf das Betriebsmittel F1 mit Hilfe eines Petrinetzes. Geben Sie eine sinnvolle Anfangsbedingung an. Beschriften Sie Stellen und Transitionen! (5P)
2. Stellen Sie den zugehörigen Erreichbarkeitsgraphen auf. Kommt es zu einer Verklemmung? Begründen Sie Ihre Antwort! (4P)

Lösung

1. Petrinetz:



2. Es kann zu keiner Verklemmung kommen, da es im Erreichbarkeitsgraphen keinen Knoten ohne abgehende Kanten gibt.



5.3.4. Aufgabe 4

Ein Elektrofahrzeug besteht unter anderem aus zwei Frontantrieben, zwei Heckantrieben, einem CAN-Bus und einer Steuerungseinheit. In der Spezifikation wird folgendes festgelegt:

„Das Fahrzeug ist genau dann funktionsfähig, wenn die Steuerungseinheit, beide Heckantriebe oder beide Frontantriebe und der CAN-Bus funktionsfähig sind. “

Für die einzelnen Komponenten am CAN-Bus sind die folgenden Dauerverfügbarkeiten bekannt:

Einheiten	**Dauerverfügbarkeit p**
Frontantrieb (F)	je $p_F = 99.9\%$

Einheiten	Dauerverfügbarkeit p
Heckantrieb (H)	je p_H=98.2%
Steuerungseinheit (S)	p_S=97%
CAN-Bus (C)	p_C=99%

1. Zeichnen Sie das zugehörige Zuverlässigkeitsersatzschaltbild. (2P)
2. Stellen Sie die Gleichung für die Dauerverfügbarkeit des Systems p_{sys} in Abhängigkeit von p_F, p_H, p_S, und p_C auf. Wie groß ist p_{sys}? (4P)

In der Praxis sind die Komponenten mit Leistungselektronik stärkeren Belastungen ausgesetzt, als die Komponenten mit Mikroelektronik. Daher soll im folgenden geprüft werden welche Maßnahmen ergriffen werden können, um die Leistungskomponenten verfügbarer zu machen. Dafür wird die Leistungselektronik (Heckantrieb) näher betrachtet, für die vom Hersteller eine MTTR von 15h angegeben ist. Die Leistungselektronik besteht unter anderem aus einer Kondensatorbatterie, einem Spannungsteiler und einem Frequenzsteller. Alle drei Komponenten müssen verfügbar sein, damit die Leistungselektronik verfügbar ist.

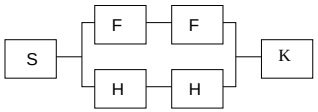
3. Welche Ausfallrate λ_{LS} ergibt sich für die Leistungselektronik, wenn die Verfügbarkeit derselben p_{LS}=0.95 ist? (3P)

Die Ausfallrate des Spannungsteilers λ_{Sp} ist 1.5 mal so groß wie die Ausfallrate der Kondensatorbaterie λ_K und die Ausfallrate λ_{Fr} des Frequenzteilers ist 1.3 mal so groß wie λ_K.

4. Bestimmen Sie λ_{SP}, λ_{FR} und λ_K. (4P)

Lösung

1.



2. Die Dauerverfügbarkeit des Systems (Fahrzeug) ergibt sich nach der oben gegebenen Spezifikation:

   $p_{sys} = p_S \cdot p_K \cdot (1-[(1-p^2_F) \cdot (1-p^2_H)] = 96.02\%$

3. $p_{LS} = MTBF_{LS} / (MTBF_{LS} + MTTR_{LS}) \longrightarrow MTBF_{LS} = (p_{LS} \cdot MTTR_{LS}) / (1-p_{LS})$

   mit p_{LS}=0.95 $\longrightarrow$ $MTBF_{LS}$=285h

   $\longrightarrow \lambda_{LS}$=1/$MTBF_{LS}$ = 0.0035 = 3.5 · 10^{-3} 1/h

4. Die Verfügbarkeit der Leistungselektronik ist von der Verfügbarkeit der Einzelkomponenten abhängig $\longrightarrow$ Serienschaltung!

   $\lambda_{ges} = \Sigma \lambda_i$

   $\lambda_{LS} = \lambda_K + \lambda_{Sp} + \lambda_{Fr} = 3.8 \cdot \lambda_K$

   $\longrightarrow \lambda_K$ = 0.000921 1/h= 9.21 · 10^{-4} 1/h

   $\longrightarrow \lambda_{Sp} = 1.5 \cdot \lambda_K$ = 0.00138 1/h 1.38 · 10^{-3} 1/h

   $\longrightarrow \lambda_{Fr} = 1.3 \cdot \lambda_K$ = 0.00119 1/h 1.119 · 10^{-3} 1/h