Lehrstuhl für Automatisierung und Informationssysteme
Prof. Dr.-Ing. B. Vogel-Heuser

Vorname:	
Nachname:	
Matrikelnummer:	

Prüfung – Informationstechnik

Sommersemester 2011

26. August 2011

Bitte legen Sie Ihren Lichtbildausweis bereit.

Sie haben für die Bearbeitung der Klausur 120 Minuten Zeit.

Diese Prüfung enthält **22** nummerierte Seiten inkl. Deckblatt.
Bitte prüfen Sie die Vollständigkeit Ihres Exemplars!

Bitte nicht mit rot oder grün schreibenden Stiften oder Bleistift ausfüllen!

Diesen Teil nicht ausfüllen.

Aufgabe	ZS	MO	BS	RK	DB	Σ	**Note:**
erreichte Punkte							
erzielbare Punkte	**50**	**66**	**60**	**36**	**28**	**240**	

Vorname, Name _____ Matrikelnummer _____

Punkte

Aufgabe ZS: Zahlensysteme und logische Schaltungen

Aufgabe ZS: 50 Punkte

a) Überführen Sie die unten angegebenen Zahlen in die jeweils anderen Zahlensysteme. *Wichtig:* Achten Sie genau auf die jeweils angegebene *Basis*!

1 (100101)$_2$ = (**37**)$_{10}$ = (**1101**)$_3$

2 (33,875)$_{10}$ = (**100001,111**)$_2$

b) Stellen Sie die in IEEE754 kodierte Single-Zahl im Dezimalsystem dar.

1	**1000 0110**	**1001 1001 0000 0000 0000 000**
V	**biased Exponent e (8 Bits)**	**Mantisse (23 Bits)**

$$Z = (-1)^V * M * 2^E$$

1. **Schritt: Mantisse M=1,10011001**
2. **Schritt: Bias berechnen : $B = 2^{(8-1)} - 1 = 127$**
3. **Schritt: Exponent E berechnen:**
   a) $e = (1000\ 0110)_2 \Rightarrow ()_{10} = 134$
   b) $E = e - B = 7$
4. **Schritt: Einsetzen:**
   $Z = (-1)^{-1} * (1,10011001)_2 * 2^7$
   $Z = -(1100\ 1100,1)_2 = -204,5$

Vorname, Name _____ Matrikelnummer _____

Punkte

c) Um eine Schaltung möglichst kostengünstig umzusetzen, soll mit Hilfe eines KV-Diagramms die Minimalrealisierung dieser Schaltung berechnet werden.
Stellen Sie das KV-Diagramm daher anhand der folgenden DNF auf. Beachten Sie dabei das bereits vorgegebene und mit „X“ gekennzeichnete „Don‘t-care“-Bit.

$$Y_{DNF} = (\bar{A} \wedge \bar{B} \wedge C) \vee (\bar{A} \wedge B \wedge C) \vee (A \wedge \bar{B} \wedge \bar{C}) \vee (A \wedge \bar{B} \wedge C)$$

A	$\bar{A}$	$\bar{A}$	A	
0	1	1	1	C
0	0	X	1	$\bar{C}$
B	B	$\bar{B}$	$\bar{B}$	

$$Y_{DNF,min} = \bar{B} \vee (\bar{A} \wedge C)$$

Vorname, Name _____ Matrikelnummer _____

Punkte

d) Vervollständigen Sie den Mehrfachverzweigungsblock des folgenden C-Programms, das Zahlen in verschiedenen Zahlensystemen ausgibt.

```
#include <stdio.h>

typedef enum{ DEZIMAL, OKTAL, HEXADEZIMAL} Zahlensysteme;

int main (void)
{
    int Zahl;
    Zahlensysteme Zahlensystem;

    printf("Geben Sie eine Dezimalzahl ein: ");
    scanf("%d",&Zahl);

    printf("Geben Sie das Zahlensystem ein\n");
    printf("0:Dezimal, 1:Oktal, 2:Hexadezimal ");
    scanf("%d",&Zahlensystem);


    switch(Zahlensystem)
    {
      case DEZIMAL:          //im Fall von DEZIMAL

          //Ausgabe der Zahl in Dezimaldarstellung
          printf("Die Dezimaldarstellung ist: %d\n",Zahl);

          break;    //Die Bearbeitung an dieser Stelle abbrechen


     case OKTAL:          //im Fall von OKTAL

          //Ausgabe der Zahl in Oktaldarstellung
          printf("Die Oktaldarstellung ist: %o\n",Zahl);

          break;    //Die Bearbeitung an dieser Stelle abbrechen


      case HEXADEZIMAL:  //im Fall von HEXADEZIMAL

          //Ausgabe der Zahl in Hexadezimaldarstellung
          printf("Die Hexadezimaldarstellung ist: %x\n",Zahl);


          break;    //Die Bearbeitung an dieser Stelle abbrechen
    }

    return 0;
}
```

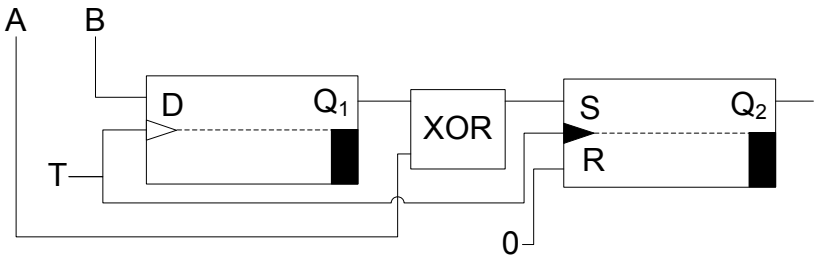
Vorname, Name

Matrikelnummer

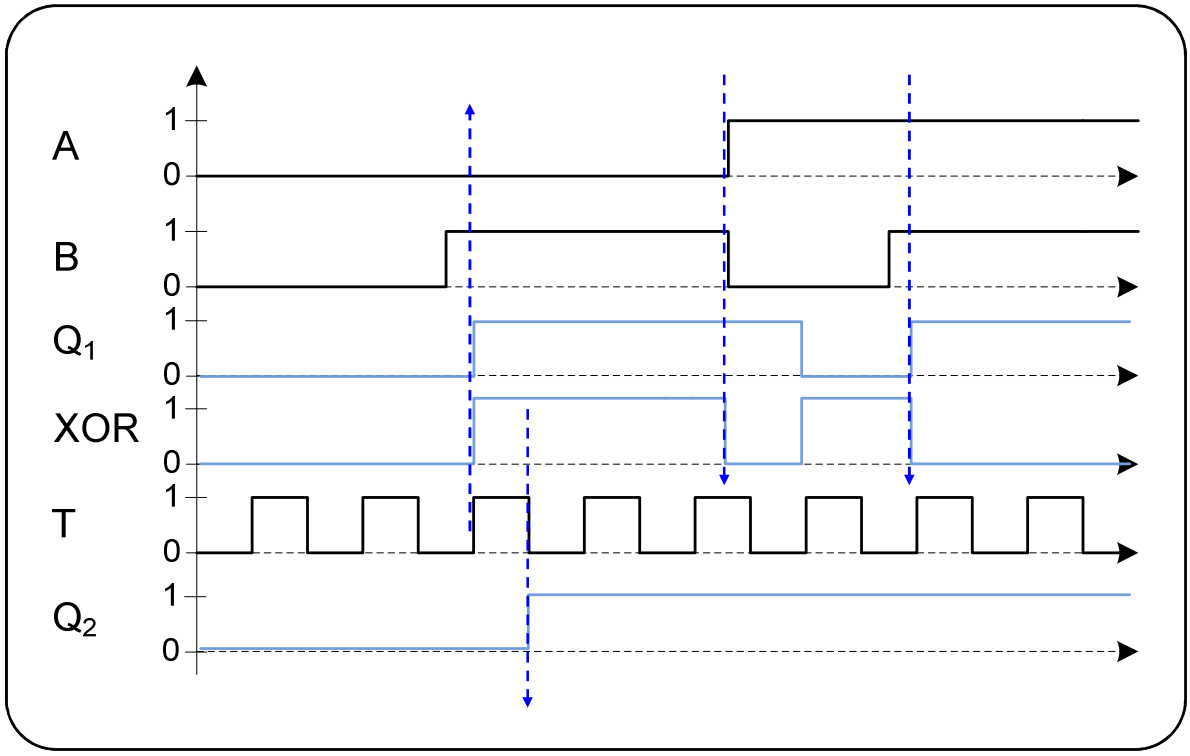
Punkte

e) Gegeben ist die unten dargestellte FlipFlop-Schaltung, bestehend aus zwei Taktflankengesteuerten FlipFlops. Erstellen Sie aus dem gegebenen Zeitdiagramm die Ausgänge, Q1 und Q2 unter Berücksichtigung des Triggersignals T und der zwei Eingänge A, B. Nehmen Sie als Ausgangssituation an, dass Q1 und Q2 alle auf Low-Level (Signal am Ausgang ist 0) sind. Beachten Sie, dass am Eingang R, des zweiten FlipFlops, dauerhaft das Low-Level Signal (Signal ist 0) anliegt. Die Funktionsweise der XOR-Schaltung ist rechts in Form einer Wahrheitstabelle angegeben.

Hinweis: Innerhalb eines Q-Diagramms gibt es Punktabzug für einen falschen Ausschlag. Negative Punkte werden nicht in die nächste Signal-Zeile mitgenommen.



A	B	XOR
0	0	0
0	1	1
1	0	1
1	1	0



Vorname, Name

Matrikelnummer

Punkte

f) Wie sieht die Konsolenausgabe des folgenden C-Programms in der Konsole aus?

```
#include <stdio.h>

int main (void)
{
     int iA = 3;
     int iB = 26;
     float fC = 5.789;
     char cD = 0;

     iB = iB >> 1;
     printf("%d %x\n", iB, iB);
     printf("%.2f %d\n", fC, (int)fC);
     printf("%d %d\n", cD || iA, cD | iA);
     iB = iA++ * 0xA;
     printf("%d %d\n", iA, iB);

     return 0;
}
```

```
13 d
5.79 5
1 3
4 30
```

Vorname, Name _____ Matrikelnummer _____

Punkte

Aufgabe 2: MO Modellierung

Aufgabe 2: 66Punkte

Die folgende Aufgabenbeschreibung benötigen Sie zur Bearbeitung der Teilaufgaben a) und b):

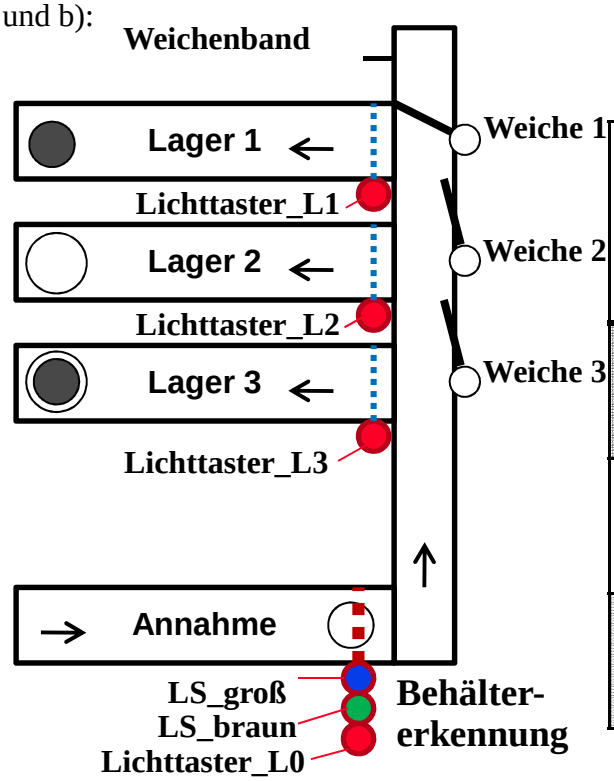


Tabelle Flaschensortierungslogik

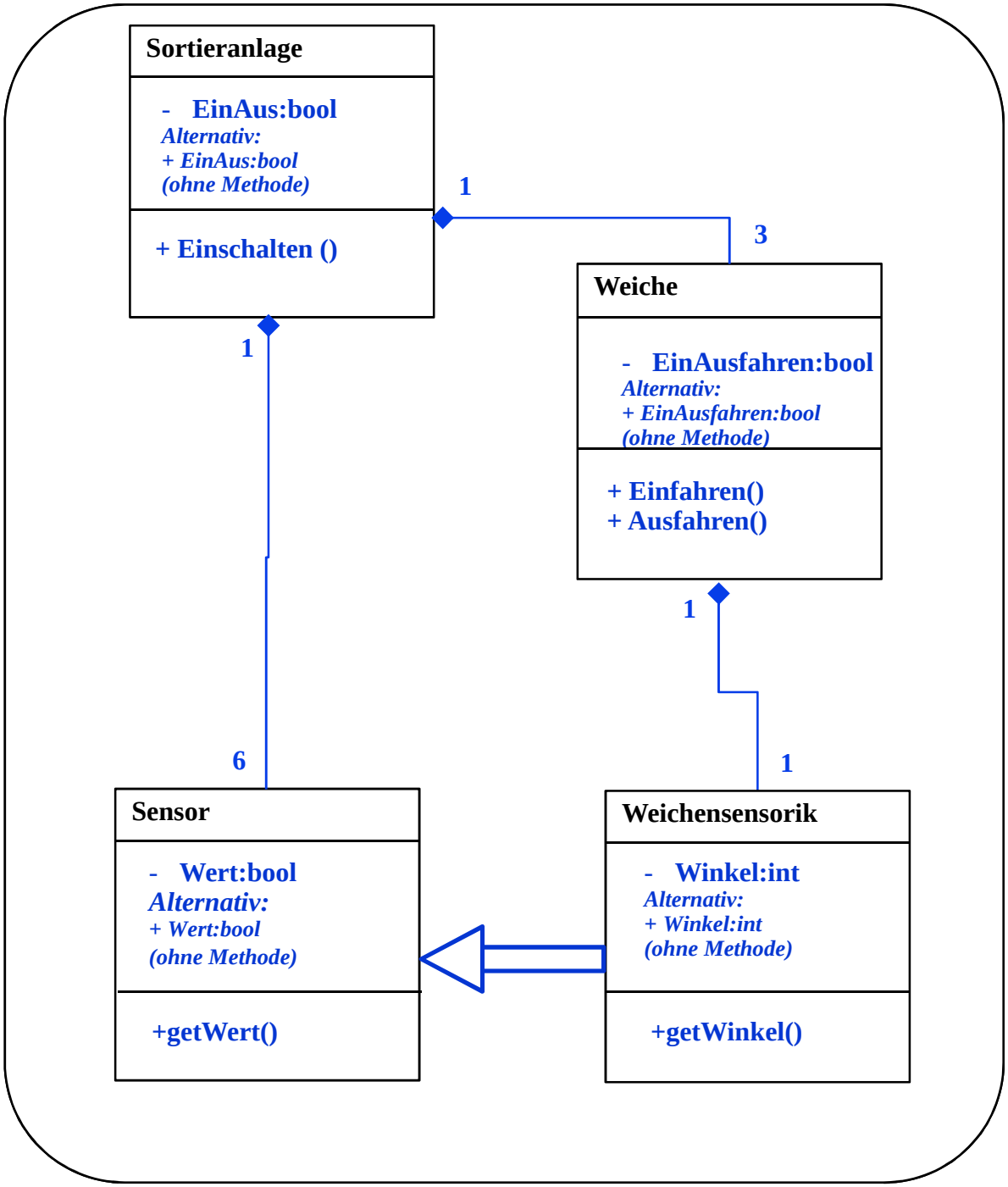
Flaschen-typ	Signal: LS_groß	Signal: LS_braun	Signal: Lichttaster_L0	Ziellager
Klein, braun	0	1	1	**1**
Groß, weiß	1	0	1	**2**
Groß, braun	1	1	1	**3**

Mit der aus zwei *Lichtschranken (LS_groß, LS_braun)* sowie einem *Lichttaster (Lichttaster_L0)* bestehenden Behältererkennung des Annahmebandes kann die Flaschensorte erkannt werden.
Alle Weichen besitzen eine *Weichensensorik*, die die Endlagen der Weichen (*1=Weiche ist Ausgefahren* und leitet Flasche in Lager um; *0=Weiche ist Eingefahren* und lässt Flasche passieren) ermitteln. Zusätzlich übermittelt die *Weichensensorik* den aktuellen *Stellwinkel (0 – 90 Grad in ganzen Zahlen).*
Ablauf der Sortierung:
Zu Beginn muss die Sortieranlage einmal *eingeschaltet* werden. Somit laufen alle Förderbänder an und sollen dann nicht mehr ausgeschalten werden.
Die Flaschensortierung beginnt am Annahmeband. Nach der Erkennung, um welchen Flaschentyp es sich handelt, sollen die Flaschen per *Aus- und Einfahren* der Weichen dem richtigen Lager zugeführt werden. Die Flaschensortierungslogik durch die Sensoren *LS_groß, LS_braun und den Lichttaster_L0* ergibt sich wie in der obigen Tabelle dargestellt und läuft wie folgt ab:
Für die Einsortierung kommt immer nur eine Flasche nach der anderen, sobald die jeweils vorhergehende Flasche einsortiert wurde. Verwenden Sie die Zustandssensoren in den Weichen, um zu prüfen, ob die entsprechende Weiche wirklich ausgefahren wurde.
Ob eine Flasche im Lager angekommen ist, wird durch die *jeweiligen Lagersensoren (Lichttaster_L1 – L3)* angezeigt. Stellen Sie nach dem Einsortieren sicher, dass die Weichen wieder eingefahren werden, bevor eine neue Flasche ankommt.

Vorname, Name _____ Matrikelnummer _____

Punkte

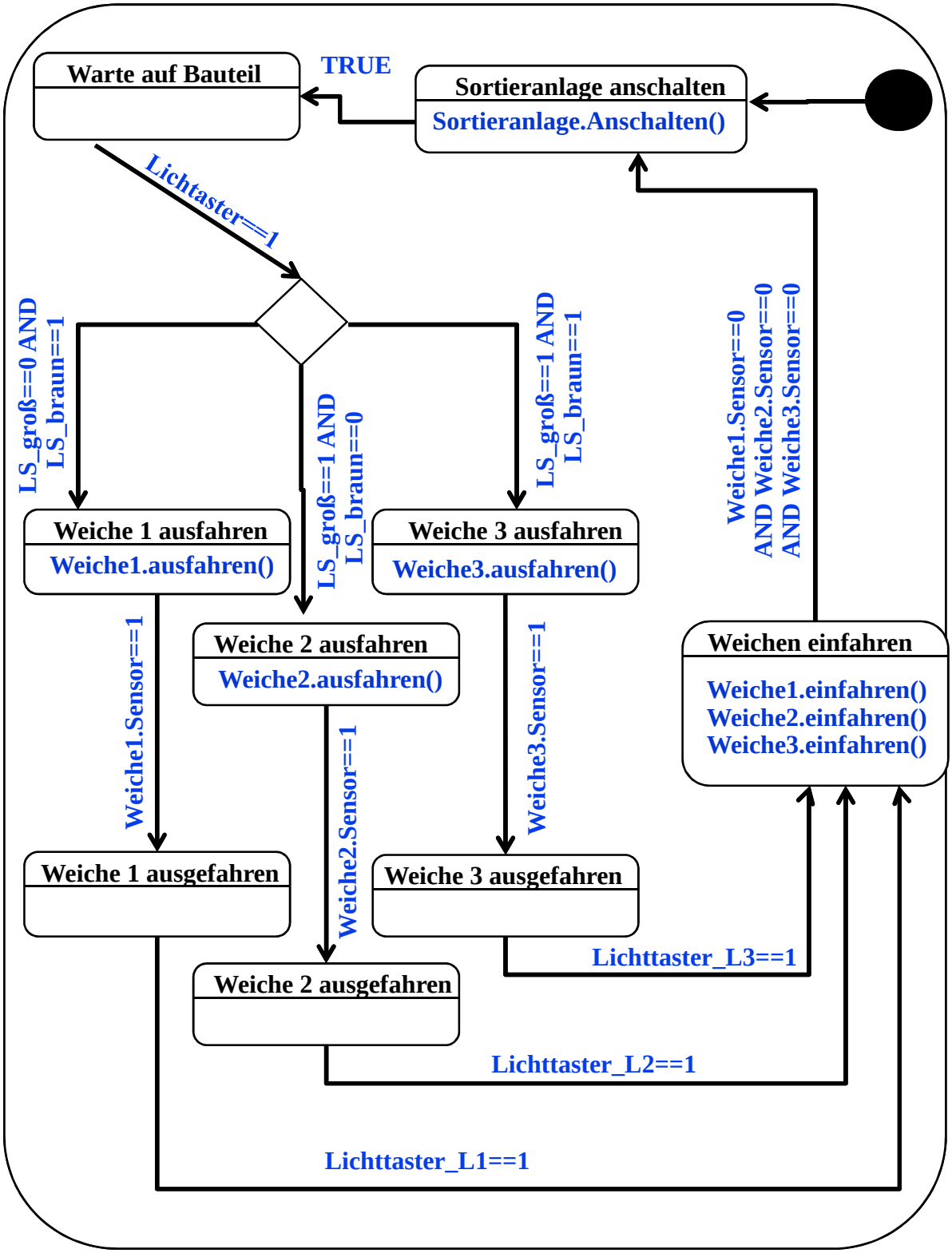
a) Ergänzen Sie das *gegebene Klassendiagramm.* Bilden Sie dazu die Struktur der Anlage durch Einzeichnen von Aggregationen, Vererbungen und Kardinalitäten ab. Ergänzen Sie die vorhandenen Klassen um nötige Attribute, Methoden sowie deren Datentyp und deren Zugriffsmodifikatoren. Definieren Sie keine weiteren Klassen.



Vorname, Name _____ Matrikelnummer _____

Punkte

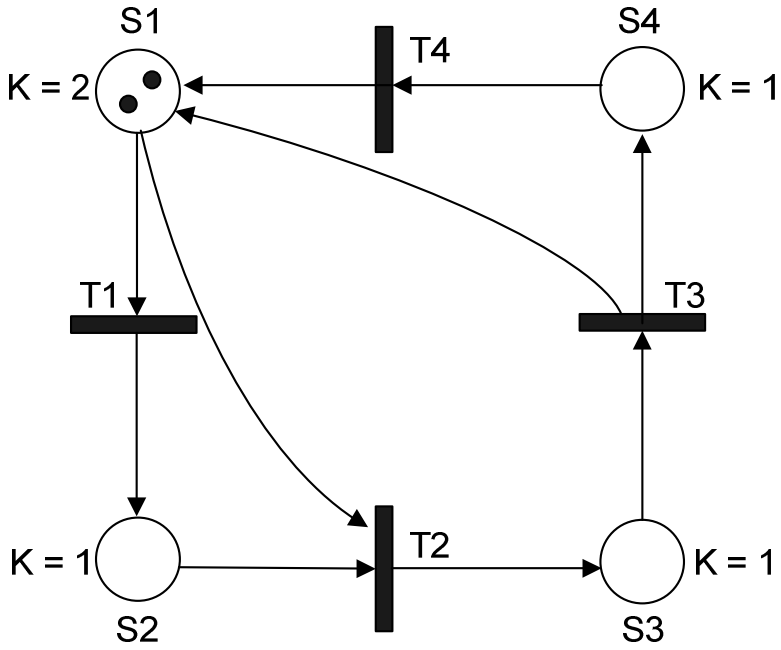
a) Ergänzen Sie das *gegebene Zustandsdiagramm*, um *alle beschriebenen Funktionen* der Anlage gemäß dem Aufgabentext abzubilden. Bilden Sie das Verhalten der Anlage mit den gegebenen Zuständen ab und ergänzen Sie nötige Übergangsbedingungen und Methodenaufrufe. Definieren Sie keine neuen Zustände.



Vorname, Name _____ Matrikelnummer _____

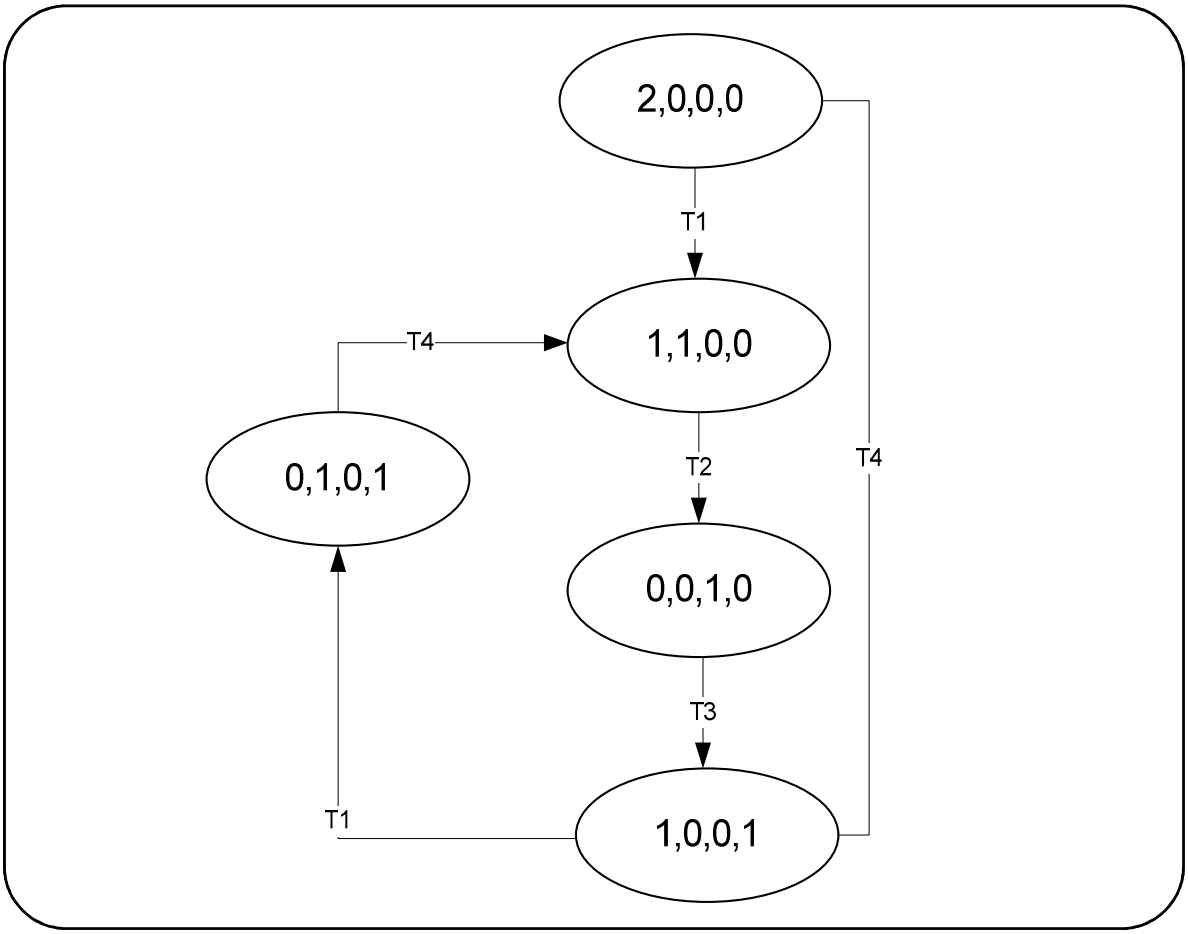
Punkte

Das unten abgebildete Petrinetz ist die Grundlage für die Aufgaben c und d.
Das Petrinetz hat die Startmarkierung: $M_0 = [2,0,0,0]^T$



$$\vec{K} = \begin{pmatrix} 2 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

c) Zeichnen Sie den Erreichbarkeitsgraphen des Petrinetzes.



Vorname, Name _____ Matrikelnummer _____

Punkte

d) Prüfen Sie rechnerisch, ob ungültige Zustände entstehen, unter der Bedingung dass folgende Sequenz schaltet: T1 → T2 → T1
Begründen Sie kurz (Stichwörter) ihre Antwort.

$$M0 = \begin{pmatrix}2\\0\\0\\0\end{pmatrix}$$

T1 schaltet

$$M0 + T1 = \begin{pmatrix}2\\0\\0\\0\end{pmatrix} + \begin{pmatrix}-1\\1\\0\\0\end{pmatrix} = \begin{pmatrix}1\\1\\0\\0\end{pmatrix}$$

T2 schaltet

$$\begin{pmatrix}1\\1\\0\\0\end{pmatrix} + \begin{pmatrix}-1\\-1\\1\\0\end{pmatrix} = \begin{pmatrix}0\\0\\1\\0\end{pmatrix}$$

T1 schaltet

$$\begin{pmatrix}0\\0\\1\\0\end{pmatrix} + \begin{pmatrix}-1\\1\\0\\0\end{pmatrix} = \begin{pmatrix}-1\\1\\1\\0\end{pmatrix}$$

Negative Zahl im Vektor, bedeutet dass eine Markierung entfernt wurde, die nicht vorhanden war.

Vorname, Name _____ Matrikelnummer _____

Punkte

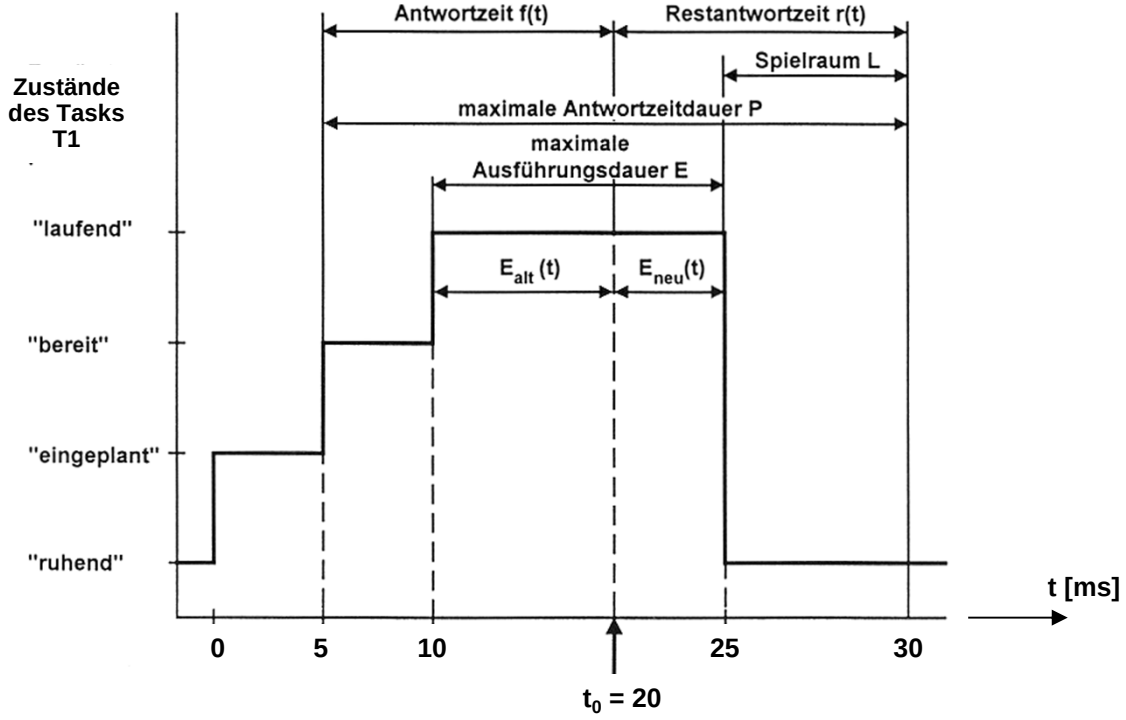
Aufgabe BS: Betriebssysteme

Aufgabe BS: 60Punkte

Hinweis: Gegeben sei ein Einprozessorsystem auf dem alle nachfolgend angegebenen Tasks ablaufen.

a) Betrachten Sie das gegebene Zeitdiagramm, welches die Zustände des Tasks T1 über der Zeit darstellt. Zum Zeitpunkt t_0 wird ein weiterer Task T2 aktiv (Wechsel von „bereit“ → „laufend“), der eine höhere Priorität und eine Ausführungszeit von 12 ms besitzt.

1) Kann der Task T1 bei einem präemptiven Scheduling noch vor seiner Deadline beendet werden? Begründen Sie Ihre Antwort.
2) Würde der Task T1 bei einem Earliest-Deadline-First Scheduling (EDF) von Task T2 unterbrochen werden unter der Annahme, dass Task T2 einen Spielraum L von 10 ms besitzt? Begründen Sie Ihre Antwort.
3) Bei welchen Taskeigenschaften wird das Round-Robin-Verfahren bei Echtzeitbetriebssystemen eingesetzt?



1) **Nein, da nach der Unterbrechung durch Task T2 nur nicht mehr genug Zeit zur restlichen Bearbeitung Verfügung stehen würde**

2) **Nein, da die Deadline von Task T2 später als die Deadline von Task T1 liegt.**

3) **Bei Tasks mit gleichen Prioritäten**

Vorname, Name _____ Matrikelnummer _____

Punkte

b) Gegeben ist die Anordnung von Semaphor-Operationen am *Anfang und am Ende der Tasks A,B,C*. Ermitteln Sie für die Fälle I, II ,III *ob und in welcher Reihenfolge* diese Tasks bei der angegebenen Initialisierung der Semaphor-Variablen ablaufen. Geben Sie zusätzlich an, ob es sich bei der Taskreihenfolge um eine Wiederholungsreihenfolge handelt, oder ob ein Deadlock entstanden ist.
Hinweis: Sind mehrere Tasks ablauffähig gilt die Priorität A>B>C. Geben Sie die Reihenfolge der ablaufenden Tasks an z.B. ABCABB an. P(Si) senkt Si um 1 V(Si) erhöht Si um 1.

Task	A	B	C
	P(SA)	P(SB)	P(SC)
	P(SA)	P(SB)	P(SC)
		P(SB)	
	…	…	…
		V(SA)	
	V(SB)	V(SA)	V(SA)
t ↓	V(SC)	V(SA)	V(SB)

Fall	SA	SB	SC
I	3	0	1
II	4	0	1
III	3	0	2

Fall I: **ACAB -> Wiederholung**

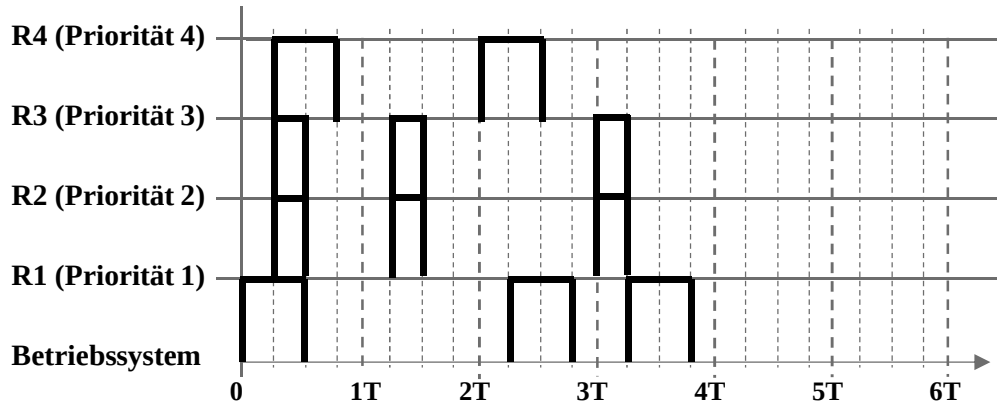
Fall II: **AACB -> Wiederholung**

Fall III: **ACAB -> Wiederholung**

Vorname, Name _____ Matrikelnummer _____

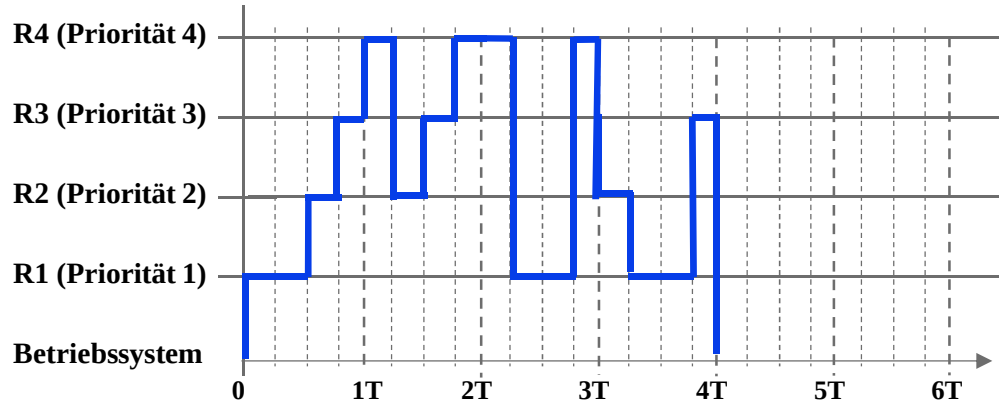
Punkte

c) Stellen Sie das Ist-Systemverhalten der Programmierart *asynchron-präemptiv* in dem angegebenen Diagramm dar.
Hinweis: Priorität 1 ist die höchste Priorität (BS>R1>R2>R3>R4).

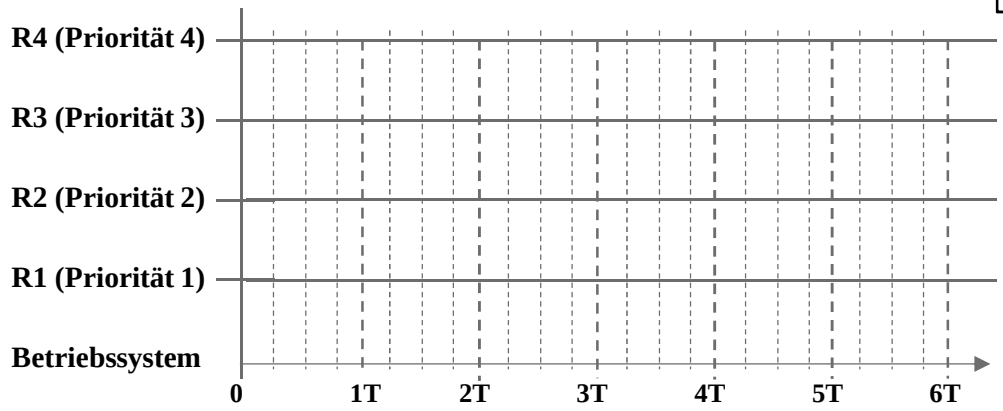


Es kann das Lösungsfeld 1 und das Reservelösungsfeld zum Bearbeiten der Aufgabe genutzt werden. Sollten Sie das Reservelösungsfeld verwenden müssen, markieren Sie dies durch ein „X“ im Kästchen des Reservelösungsfelds. Das Lösungsfeld 1 wird nicht gewertet, sobald ein „X“ im Kästchen des Reservelösungsfelds angegeben ist!

Asynchron-präemptive Programmierung: (Lösungsfeld 1)



Asynchron-präemptive Programmierung: Reservelösungsfeld werten ☐



Vorname, Name _____ Matrikelnummer _____

Punkte

d) Kreuzen Sie im Lösungsfeld die Bedeutung für „call-by-value“ in der Programmiersprache C an.

☒	eine Kopie der übergebenen Variable wird erstellt und an die aufgerufene Funktion übergeben (keine Änderung der ursprünglich übergebenen Variable).
☐	ein Zeiger auf eine Variable wird übergeben, über den die Variable ausgelesen bzw. verändert werden kann.

e) Sie möchten im Bereich der Scheduler-Entwicklung einen Datentyp für die Prozessverwaltung anlegen. Definieren Sie einen neuen Datentyp `PROCESS` als Struktur mit folgenden Elementen: `iUID` (Typ: Integer), `iPriority` (Typ: unsigned Integer) und `szName` (Array mit 20 Elementen vom Typ Char).

```
typedef struct
{       int iUID;
        unsigned int iPriority;
        char szName[20];
} PROCESS;
```

f) In der Programmiersprache C soll ein Programm entwickelt werden, welches Prozesse (Datentyp aus e) verwenden, in einem Array verwaltet und diese nach ihrer Priorität sortiert.

(1) Gegeben seien die rechts abgebildeten Prozesse. Legen Sie ein Array mit dem Namen `aProcesses` an und initialisieren Sie es mit den Werten des ersten Prozesses in der Tabelle.

iUID	4711	42	…
iPriority	5	1	…
szName	Prozess 1	Prozess 2	…

```
#include <string.h>
void main(){
     //Deklaration des Arrays aProcesses(Größe: 5 Elemente)
      PROCESS aProcesses[5];

     // Initialisierung der Werte von Prozess 1
     // als erstes Element des Arrays
      aProcesses[0].iUID = 4711;
      aProcesses[0].iPriority = 5;
      strcpy(aProcesses[0].szName, "Prozess 1");

     // Hier werden die restlichen Prozesse
     // analog zum Prozess 1 initialisiert
     // ...
     sortProcesses(aProcesses, 5); // Sortierung
}
```

Vorname, Name _____ Matrikelnummer _____

Punkte

(2) Erstellen Sie eine Funktion, welche die Prozesse anhand ihrer Priorität absteigend sortiert (Je höher die Priorität des Prozesses, desto geringer der Zahlenwert von `iPriority`). Der Funktionsprototyp soll kompatibel zu dem Funktionsaufruf aus (1) sein.

```
void sortProcesses(PROCESS* processes, int size)
{
  int i, j;

  // Durchlaufen des Arrays
  for(i = size-1 ; i >= 1 ; i--  )
  {
    for(j = 0; j < i; j++)
    {
      if(processes[j].iPriority > processes[j+1].iPriority)
      {
          // Tauschen der benachbarten Elemente
          PROCESS tmp = processes[j];
          processes[j] = processes[j+1];
          processes[j+1] = tmp;
      }
    }
  }
}
```

g) Innerhalb eines Programms wird u.a. ein Zeiger `piArray` und ein Array `aiInput` verwendet. Der Zeiger `piArray` zeigt auf das erste Element des Arrays `aiInput`.

(1) Nennen Sie die zwei Möglichkeiten, wie Sie auf das 7. Element des Arrays zugreifen können. Benutzen Sie dabei einmal den Zeiger und einmal direkt das Array.

(2) Sind die folgenden beiden Ausdrücke äquivalent? Erläutern Sie kurz ihre Antwort.

`aiInput` und `&aiInput[0]`

i) ***(piArray +6)**
aiInput [6]

ii) **Ja, da aiInput die Adresse auf das erste Element darstellt. aiInput[0] stellt das erste Element dar und der Adressoperator & liefert wiederum die Adresse des ersten Elements**

Vorname, Name

Matrikelnummer

Punkte

Aufgabe RK: Rechnerkommunikation

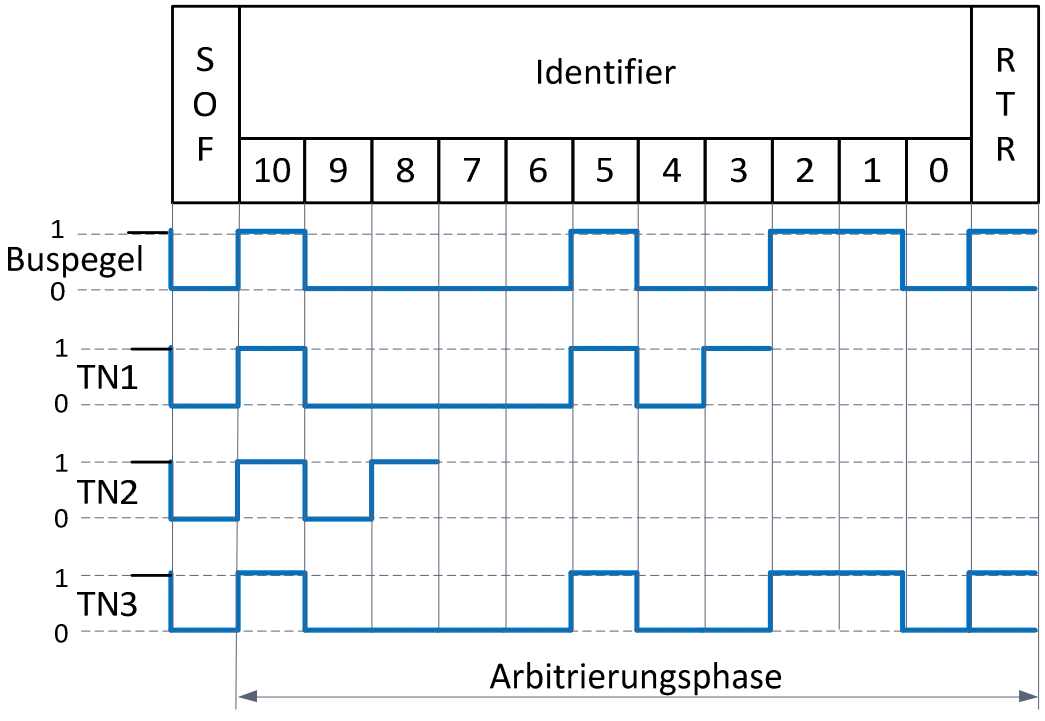
Aufgabe RK: 36 Punkte

a) Drei Teilnehmer sind an einem CAN Bus angeschlossen und wollen zum gleichen Zeitpunkt senden. In der Arbitrierungsphase senden alle Teilnehmer die im folgenden angegebenen Identifier:
   a) Teilnehmer 1: 42E (hex)
   b) Teilnehmer 2: 567 (hex)
   c) Teilnehmer 3: 426 (hex)

   Stellen Sie im folgenden Diagramm den Arbitrierungsvorgang dar und geben Sie an, welcher Teilnehmer zu welchem Takt aus dem Arbitrierungsprozess ausscheidet. Stellen Sie weiterhin den resultierenden Buspegel des CAN-Busses im Diagramm dar.

TN1 – 42E – 100 0010 1110
TN2 – 567 – 101 0110 0111
TN3 - 426 - 100 0010 0110

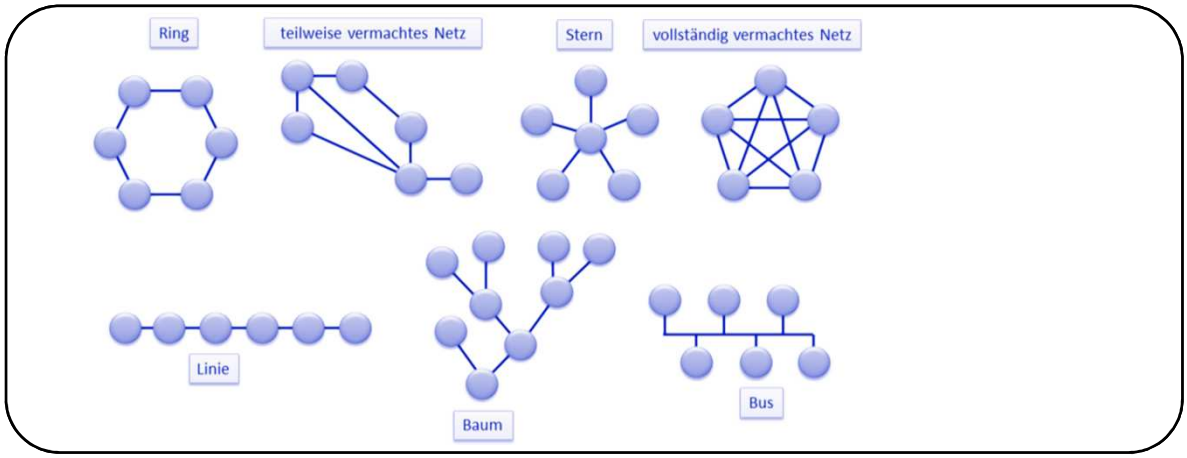
TN1 – Takt 9,
TN2 – Takt 4,
TN3 - sendet



Vorname, Name _____ Matrikelnummer _____

Punkte

b) Benennen oder skizzieren Sie drei Netzwerktopologien.

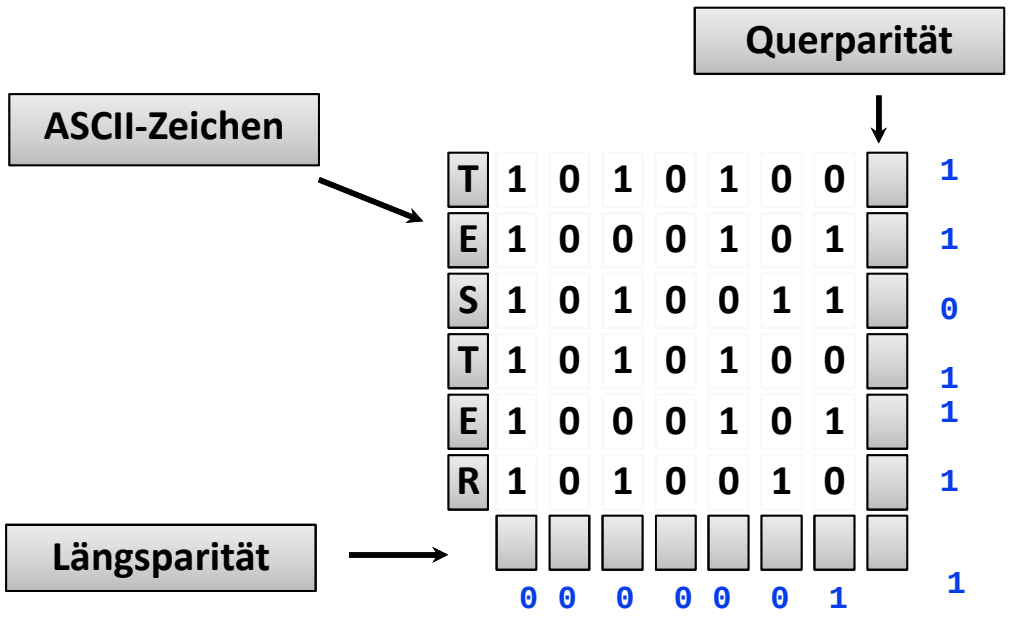


c) Markieren Sie die kollisionsfreien Buszugriffsverfahren mit einem X.

ALOHA	
CSMA/CD	
Token-Passing	X
TDMA	X

Master/Slave	X
CSMA/CA	

d) Befüllen Sie die folgende Tabelle mit Längs- und Querparitäten (gerade Parität).



Vorname, Name _____ Matrikelnummer _____

Punkte

e) Vervollständigen Sie die Implementierung der Funktion `test_primzahl` in der Programmiersprache C entsprechend der Kommentare.

```
#include <stdio.h>

void test_primzahl(int Zahl)
{
   int Testzahl = Zahl-1;

   //Prüfung ob die Zahl 1 oder 0 ist
   //(Das sind per Definition keine Primzahlen)

       if(Zahl==0 || Zahl==1)
   {
      //Auf der Konsole soll ausgegeben werden
      //"Die Zahl ... ist keine Primzahl"
      //Dabei sollen statt den ... die geprüfte Zahl stehen.
      //Danach soll ein Zeilenumbruch ausgegeben werden

          printf("Die Zahl %i ist keine Primzahl\n",Zahl);
      //Die Funktion soll an dieser Stelle beendet werden

          return;
   }

   while(Testzahl>1)
   {
      //Prüfung ob der Rest der Division der Zahl durch
      //die Testzahl gleich 0 ist

          if(Zahl % Testzahl == 0)
      {
          //Auf der Konsole soll ausgegeben werden
          //"Die Zahl ... ist keine Primzahl"
          //Dabei sollen statt den ... die geprüfte Zahl stehen.
          //Danach soll ein Zeilenumbruch ausgegeben werden

             printf("Die Zahl %i ist keine Primzahl\n",Zahl);

          //Die Funktion soll an dieser Stelle beendet werden

            return;
      }

      Testzahl--;
   }

   //Auf der Konsole soll ausgegeben werden
   //"Die Zahl ... ist eine Primzahl"
   //Dabei sollen statt den ... die geprüfte Zahl stehen.
   //Danach soll ein Zeilenumbruch ausgegeben werden

       printf("Die Zahl %i ist eine Primzahl\n",Zahl);
}
```

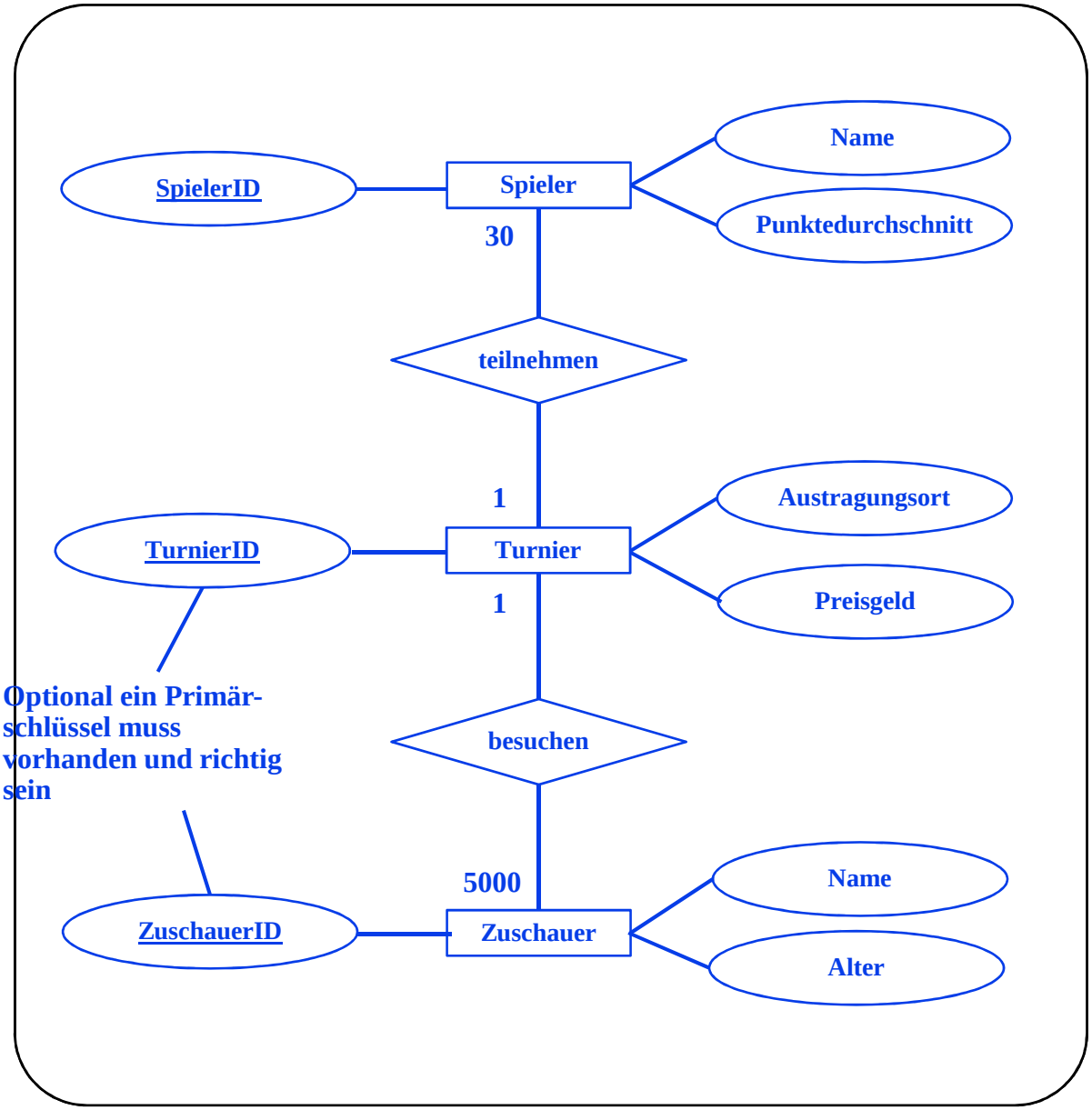
Vorname, Name _____ Matrikelnummer _____

Punkte

Aufgabe DB: Datenbanken

Aufgabe DB: 28 Punkte

a) ER Modellierung:
Der englische Dartverband will für die bevorstehende Saison eine Datenbank anlegen, in der alle Angaben über Spieler, Turniere und Zuschauer gesammelt werden sollen.
Folgende Randbedingungen gelten: Jeder *Spieler* wird mit seinem Namen und einem Punktedurchschnitt beschrieben. An einem *Turnier* nehmen genau 30 *Spieler,* die dem Dartverband angehören, teil. Für alle *Turniere* wird ein Veranstaltungsort und ein Preisgeld festgelegt. Ein *Turnier* wird von 5000 *Zuschauern* besucht. Dabei wird von jedem *Zuschauer* der Name und das Alter gespeichert.
Zeichen Sie ein Entity-Relationship-Diagramm. Ergänzen Sie sinnvolle Primärschlüssel. Geben Sie die entsprechenden Kardinalitäten an.



Vorname, Name _____ Matrikelnummer _____

Punkte

Für alle weiteren Teilaufgaben ist folgender unvollständiger Datenbankausschnitt gegeben. Der gegebene Datenbankausschnitt steht in keinen Zusammenhang mit der Teilaufgabe a).

Spieler	
Name	**Punktedurch-schnitt**
Taylor	100
Lewis	97
Wade	98
Anderson	102
Whitlock	92
Barneveld	89

Zuschauer		
Vorname	**Nachname**	**Alter**
Holger	Meier	47
Paul	Roberts	19
Wayne	Giggs	32

Ticket			
TicketID	**Reihe**	**Platz**	**Preis**
5	*12*	*C15*	*29,40*

Turnier			
Austragungsort	**Preisgeld**	**Termin**	**Sieger**
London	200000	Mai	Lewis
Dublin	30000	Juli	Taylor
Newcastle	95000	August	Whitlock

b) Legen Sie mit einem SQL-Befehl die Tabelle *Ticket* an.

```
CREATE TABLE Ticket (TicketID INT PRIMARY KEY,
                    Reihe INT,
                    Platz VARCHAR(50),
                    Preis FLOAT);
```

Vorname, Name | Matrikelnummer

Punkte

c) Befüllen Sie die Tabelle *Ticket* mit einem SQL-Befehl mit folgenden Daten:
- TicketID: *5*
- Reihe: *12*
- Platz: *C15*
- Preis*: 29,40*

INSERT INTO Ticket (TicketID, Reihe, Platz, Preis)

VALUES (5, 12, "C15", 29.40);

d) Geben Sie einen SQL-Befehl an, mit dem nur der *Punktedurchschnitt* des Spielers, der das Turnier am Austragungsort *Dublin* gewonnen hat, ausgegeben wird.

SELECT Punktedurchschnitt FROM Spieler WHERE Name =

(SELECT Sieger FROM Turnier WHERE Austragungsort = "Dublin");