DHBW
Duale Hochschule
Baden-Württemberg
Stuttgart

Übungen

zur Vorlesung

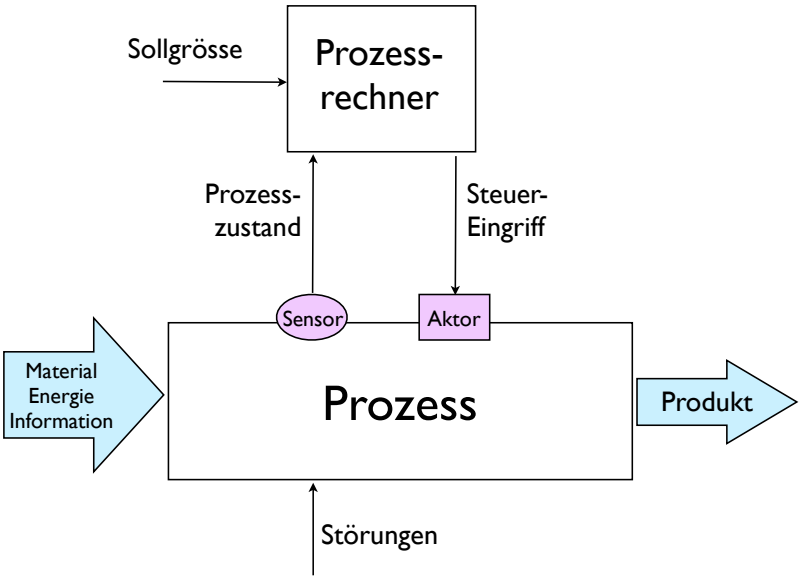
Realzeitsysteme

(Programmierung von Echtzeitsystemen)

Edition 0.5, 10.12.2012
Autor: Stephan Rupp

1. Technischer Prozess

Ein technischer Prozess transformiert Eingangsgrößen wie Material, Energie und Information zu einem Ausgangsprodukt, wie in der Abbildung unten gezeigt. Zur Steuerung wird ein Prozessrechner verwendet. Der Prozessrechner erhält durch Sensoren Informationen über den Zustand des Prozesses und kann über Aktoren in den Prozess eingreifen. Der Prozess kann auf einen Sollzustand eingestellt werden, wobei Störungen ausgeglichen werden.



Frage 1: Benennen Sie die oben definierten Begriffe für folgende konkrete Beispiele: Drucker, Automobil, Kaffeeautomat.

Frage 2: Beschreiben Sie mit den oben eingeführten Begriffen eine einfache Füllstandsregelung (inkl. Regelalgorithmus).

Frage 3: Vom Prozessrechner aus betrachtet gibt es Eingangsgrößen, aus denen der Prozesszustand abgeleitet werden kann, und Ausgangsgrößen, mit denen die Aktoren zum Eingriff in der Prozess gesteuert werden. Der Zustand des Prozesses kann hierbei innerhalb des Prozessrechners als Prozessmodell geführt werden. Wenn man die Eingangsgrößen als Menge X bezeichnet, die Ausgangsgrößen als Menge Y und die Zustände als Menge Z, welche Bedeutung haben dann folgende Begriffe:

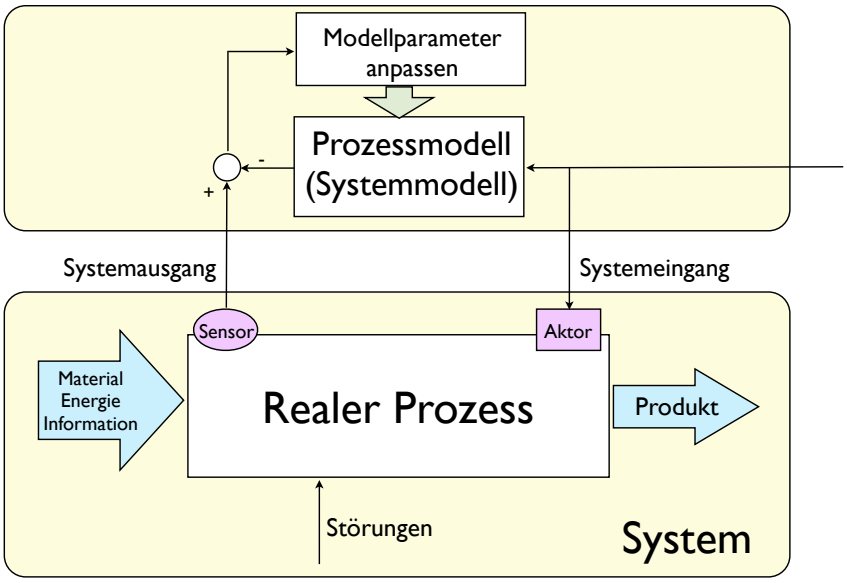
- Überführungsfunktion f: Z x X -> Z
- Ausgabefunktion g: Z x X -> Y

Frage 4: Prozessmodelle können bzgl. des Zeitverlaufs als diskontinuierliche (zeitdiskrete) Systeme bzw. als kontinuierliche Systeme abgebildet werden, wobei lineare Systeme einen wichtigen Spezialfall darstellen. Diskutieren Sie die Unterschiede.

2. Prozessmodell

Betrachtet man den in Aufgabe 1 gezeigten realen Prozess als System, so stellen die Messpunkte für die Eingangsgrößen des Prozessrechners den Systemausgang dar, und die Eingriffspunkte für die Aktoren den Systemeingang. Bei dieser Betrachtung wird das System aus der Perspektive des Prozessrechners bzw. der Regelung betrachtet.

Die folgende Abbildung zeigt, wie sich durch Stimulation des realen Systems Informationen zur Bestimmung der Parameter des Prozessmodells gewinnen lassen.



Frage 1: Erläutern Sie die gezeigte Anordnung und ihre Funktionsweise.

Frage 2: Auf welche Weise lässt sich das Prozessmodell mathematisch beschreiben?

Frage 3: Welchen Einfluss haben Störungen auf die Abschätzung der Modellparameter?

Frage 4: Welcher Zusammenhang zwischen den Messpunkten und den Zustandsgrößen im Prozessmodell wäre wünschenswert, um des Messaufwand zu reduzieren?

3. Prozessablauf

Die folgende Abbildung zeigt den zeitlichen Ablauf eines Prozesses.

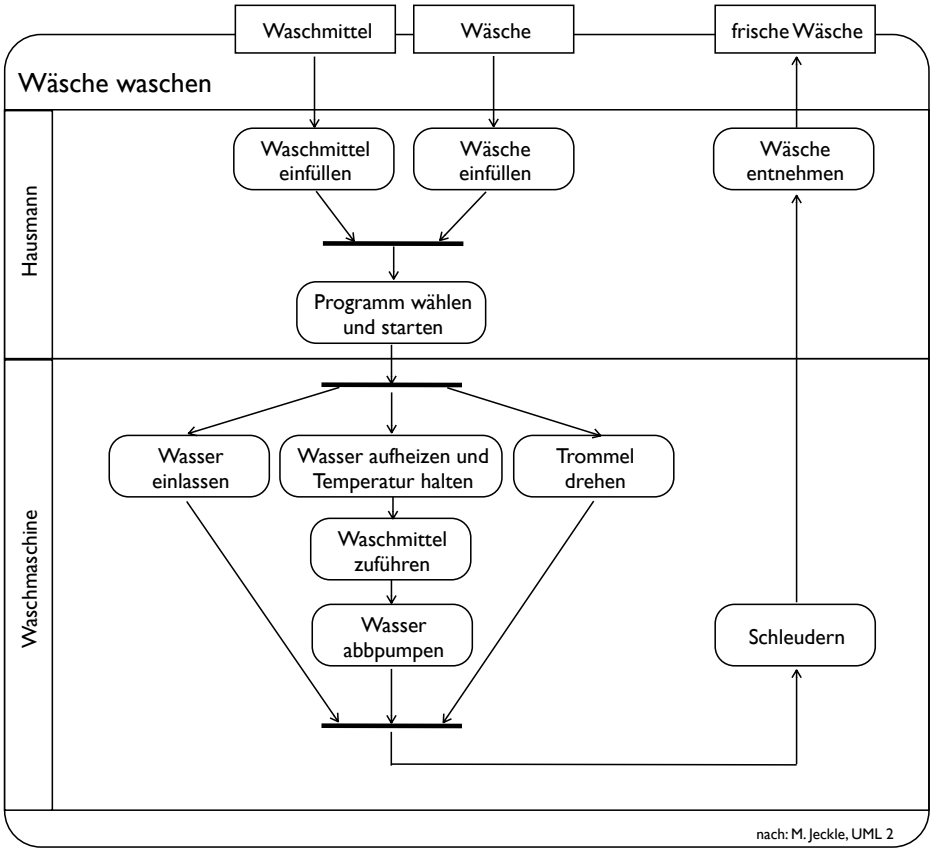
Frage 1: Um welche Aktivität handelt es sich? Was sind Anfangsprodukte und Endprodukte des Prozesses? Welches sind die beteiligten Akteure? Welches sind die beteiligten Objekte? Erläutern Sie den Ablauf des Prozesses.

Frage 2: Erläutern Sie die Komponenten der Darstellung: Wie sind einzelne Schritte im Ablauf (Aktionen) abgebildet? Wo gibt es Verzweigungen in parallele (nebenläufige) Aktivitäten? Wo werden nebenläufige Aktivitäten synchronisiert?

Frage 3: Einzelne Aktionen wie z.B. „Schleudern“ könnte man auch weiter detaillieren. Wie beispielsweise?

Frage 4: Ein weiterer Arbeitsgang „Spülen“ liesse sich ergänzen. Wie beispielsweise?

Frage 5: Versuchen Sie eine sinngemässe Darstellung für den Prozessverlauf bei einem Kaffeautomaten. Falls Sie die Auswahl von Abläufen abhängig von Entscheidungen vorsehen, führen Sie ein passendes grafisches Symbol ein (Entscheidungsknoten inklusive der Bedingungen für die Entscheidungen).



4. Prozesszustände

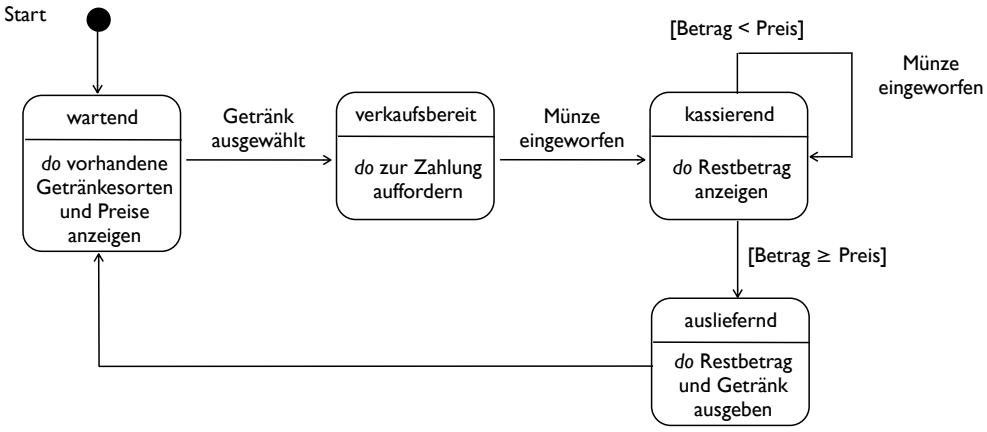
Eine andere Sicht auf den Prozess ist sein jeweiliger Zustand. Hierbei wird angenommen, dass das Verhalten des Prozesses je nach Zustand unterschiedlich ist. Zwar sind technische Prozesse oder Automaten nicht je nach Tagesform gut oder schlecht aufgelegt, jedoch reagiert beispielsweise ein Getränkeautomat oder Fahrkartenautomat in aller Regel anders, nachdem eine passende Summe Geld eingeworfen wurde.

Frage 1: Bei der Waschmaschine wären die Zustände beispielsweise „Wartend“, „Vorwäsche“, „Hauptwäsche“, „Spülen“, sowie „Schleudern“. Geben Sie die Zustände und Zustandsübergänge in einem Diagramm wieder.

Frage 2: Die folgende Abbildung zeigt die Zustände und Zustandsübergänge für einen Getränkeautomaten. Erläutern Sie die Abbildung.

Frage 3: Identifizieren Sie die grafischen Komponenten: Wie sind Zustände gekennzeichnet? Wie sind Zustandsübergänge (Transitionen) gekennzeichnet? Welche Aktivitäten werden innerhalb der Zustände ausgeführt? Wann erfolgen Zustandswechsel?

Frage 4: Das Diagramm ist in der dargestellten Form wenig praxistauglich: Der Getränke-automat verbleibt bei einer unzureichenden Zahlung für unbegrenzte Zeit im Zustand kassierend. Auch fehlt eine Abbruchmöglichkeit durch den Benutzer. Ergänzen Sie das Diagramm.



5. Echtzeitverhalten

Bei Betrachtung der in Aufgabe 1 gezeigten Abbildung muss der Prozessrechner zeitlich angemessen in das Prozessgeschehen eingreifen, wenn er Wirkung zeigen soll.

Frage 1: Wie würden Sie den Begriff „Echtzeitverhalten“ als Anforderung an den Prozess-rechner definieren?

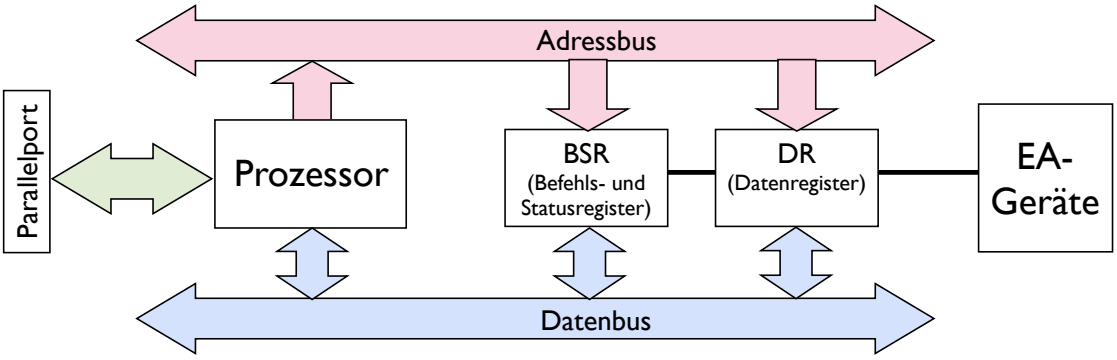
Frage 2: Was würden Sie unter dem Begriff „definierte Antwortzeiten“ verstehen?

Frage 3: Welche Anforderungen an Antwortzeiten ergeben sich für folgende Beispiele: phasengenaue Drehzahlregelung, Steuerung eines Fahrzeugs, Steuerung der Bewegung einer Fräsmaschine (Werkstück gegen Werkzeug)?

Frage 4: Diskutieren Sie die technische Umsetzung im Prozessrechner bzgl. Prozessabbild, Anwendung, Betriebssystem, Aktualisierung des Prozessabbildes, Steuereingriffe. Welches sind die kritischen Stellen?

6. Geräteschnittstellen

Folgende Abbildung zeigt eine Übersicht über die im Manuskript (siehe Literatur (1)) in den Abschnitten 1.5.1 und 1.5.2 diskutierten Geräteschnittstellen, die entweder über Geräte-register realisiert werden, bzw. über Parallelports.



Frage 1: Erläutern Sie die in der Abbildung gezeigten Komponenten und ihr Zusammenspiel.

Frage 2: Erläutern Sie folgende Konzepte: (1) spezielle EA-Register mit eigenem Adressbereich bzw. (2) EA-Register im Adressraum des Arbeitsspeichers (memory mapped I/O)

Frage 3: Erläutern Sie folgende Konzepte: (1) programmgesteuertes Ein- und Auslesen der EA-Register bzw. (2) gerätegesteuertes Ein- und Auslesen der EA-Register (DMA: Direct Memory Access).

Frage 4: Eine Alternative zu den EA-Registern ist der Anschluss von Geräten über Parallelports (sofern der Prozessor über General Purpose IO Ports verfügt). Erläutern Sie das Funktionsprinzip im Vergleich zu den bisher diskutierten Verfahren (Fragen 2 und 3).

Frage 5: Was ist in allen bisher diskutierten Fällen (Fragen 2, 3 und 4) Ergebnis der Interaktionen mit EA-Geräten aus Sicht des Prozessors (Stichwort: Arbeitsspeicher)?

7. Unterbrechungen durch Alarme

Geräte kommunizieren Statusänderungen als Alarme (engl. Interrupt) an den Prozessor. Je nach Priorität des Alarms unterbricht der Prozessor seine laufende Tätigkeit und fügt eine Routine zur Alarmbehandlung aus. Bild 1.21 in Abschnitt 1.5.3 des Vorlesungsmanuskript (siehe Literatur (1)) zeigt den Zeitverlauf einer solchen Unterbrechung.

Frage 1: Erläutern Sie die in Bild 1.21 gezeigten Zusammenhänge.

Frage 2: Um aus der laufenden Tätigkeit in eine andere Routine zu wechseln (bzw. von einem Kontext in einen anderen Kontext zu wechseln), schreibt der Prozessor den Inhalt seiner Register und eine Referenz auf den Stand seiner Arbeit (Stapelzeiger) in den Arbeitsspeicher und liest den Kontext der neuen Routine vom Arbeitsspeicher. Warum?

Frage 3: Welchem Einfluss auf das Echtzeitverhalten hat die Kontextumschaltung?

Frage 4: Wie kann man sicherstellen, dass kritische Teile nicht unterbrochen werden? Was sollte mit den in dieser Zeit anfallenden Alarmen aus Sicht des Prozesses geschehen?

8. Analog-Digital-Wandler

Für die Messung analoger Prozessgrößen werden Analog-Digital-Wandler eingesetzt. Die benötige Wortbreite hängt hierbei von der gewünschten Messgenauigkeit ab.

Frage 1: Welche Messgenauigkeit ermöglicht ein 12-Bit AD-Wandler?

Frage 2: Der Dynamikbereich der traditionell PCM kodierten Digital Audio Disks (CDs) wird mit 96dB angegeben. Welche Auflösung (Wortbreite) benötigt die AD-Wandlung?

Frage 3: Aus Sicht des Prozessrechners sind folgende Größen von Interesse : (1) die Dauer der Wandlung, (2) die Abtastrate bzw. der Zyklus der Abfrage der analogen Messgrößen. Diskutieren Sie mögliche Realisierungen zur Abfrage analoger Prozessgrößen aus Anwendungssicht.

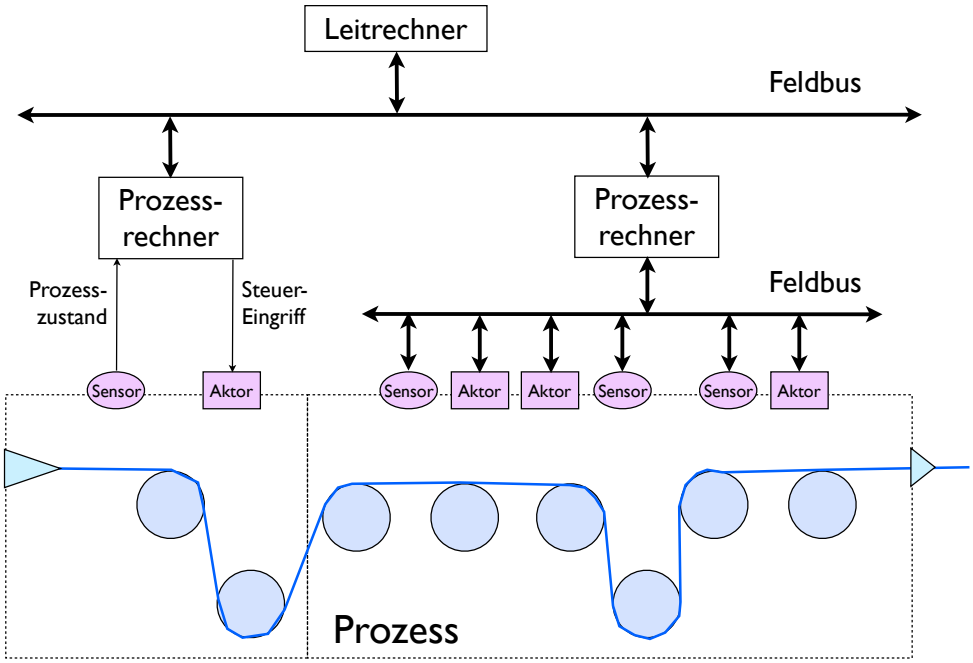
Frage 4: Nebenläufigkeit und Synchronisation bei der Bedienung des AD-Wandlers: Erläutern Sie das Aktivitätsdiagramm im Manuskript (siehe Literatur (1)), Bild 1.23

9. Feldbusse

Feldbusse werde eingesetzt, um mehrere Prozessrechner im Feld miteinander zu verbinden, bzw. auch zur Anbindung intelligenter Sensoren und Aktoren (wobei unter Intelligenz hier eine Feldbusschnittstelle am Sensor oder Aktor verstanden wird). Die folgende Abbildung zeigt eine Übersicht, in der ausserdem ein Leitrechner als übergeordneter Prozessrechner dargestellt ist.

Serielle Feldbusse: Traditionell wurden Feldbusse als serielle Feldbusse ausgeführt. Beim seriellen Feldbus wird eine Leitung als Kommunikationsmedium zwischen den angeschlossenen Geräten benutzt. Serielle Feldbusse haben folgende Eigenschaften:

- Obwohl die Busleitung an den Enden reflexionsfrei abgeschlossen ist, werden serielle Feldbusse mit relativ niedrigen Frequenzen betrieben (elektrisch kurze Leitung, quasistationärer Betrieb), typischerweise mit 1 Mbit/s.
- Der Feldbus ist gemeinsames Medium. Es gibt Mechanismen zur Erkennung der Kollisionen von Nachrichten. Ein Busmaster organisiert als Moderator die Kommunikation, z.B. durch zyklische Abfrage aller beteiligter Geräte: Das abgefragte Gerät antwortet mit seinen Prozessdaten, alle angeschlossenen Geräte hören mit. Die Geräte verhalten sich also wie Teilnehmer einer Besprechung um einen gemeinsamen Tisch.
- Im Vergleich zu den parallelen Busplatinen früherer Gerätegenerationen funktioniert der serielle Feldbus ähnlich, jedoch eben seriell, d.h. Anfragen und Antworten werden als Nachrichten verschickt mit Nachrichtenkopf (engl. Header, für Nachrichtentyp, Geräteadressen, Prüfsummen etc.) und Nachrichtenkörper für die Nutzdaten.



Netzwerkbasierte Feldbusse: Zunehmend wird Ethernet als Medium für Feldbusse eingesetzt. Die wesentlichen Unterschiede zum seriellen Feldbus bestehen darin, dass

- in der Regel statt eines gemeinsamen Mediums nun mit Leitungspaaren in Senderichtung und Empfangsrichtung gearbeitet wird,
- Ethernet Switches zum Anschluss der Geräte und zur Verbindung von Netz-segmenten eingesetzt werden,

- die Übertragungsgeschwindigkeit sich erhöht (typischerweise 100 Mbit/s), sowie
- der Nachrichtenaustausch ähnlich oder auch ganz anders organisiert werden kann.

Frage 1: Welche Zykluszeiten lassen sich mit einem serieller Feldbus erzielen, wenn 10 Geräte angeschlossen sind, die Übertragungsrate 1 Mbit/s beträgt, die Länge einer Nachricht 64 Bytes beträgt. Hierbei wird angenommen, dass der Busmaster alle Geräte individuell abfragt. (Hinweis: Wie lange dauert die Übertragung einer Nachricht? Wie lange dauert es, bis alle Geräte abgefragt wurden?)

Frage 2: Was ändert sich, wenn statt des seriellen Feldbusses Ethernet mit einer Übertragungsrate von 100 Mbit/s eingesetzt wird, die Organisation der Nachrichten aber beibehalten wird?

Frage 3: Über den Feldbus fragt der Busmaster zyklisch z.B. alle 100 ms alle angeschlossenen Geräte ab. Auf diese Weise kann der Prozessrechner das Prozessabbild alle 100 ms aktualisieren. Eine Zustandsänderung unmittelbar nach der Abfrage wird auf diese Weise erst im nächsten Zyklus erfasst. Eine technische Alternative wäre, dass die Geräte spontan Zustandsmeldungen über dass Netz kommunizieren, sobald sich ein Zustand geändert hat. Diskutieren Sie Vorteile und Nachteile beider Verfahren bzgl. der Steuerbarkeit des Prozesses.

Frage 4: Ein praktischer Vorteil der Netzwerktechnik besteht darin, dass neben den Prozessdaten über das gleiche Netz auch regulärer Ethernet-Verkehr übertragen werden kann, beispielsweise von Benutzer-PCs bzw. von Videokameras. Welcher Konflikt ergibt sich hieraus? Welcher Einfluss auf das Echtzeitverhalten ergibt sich hierdurch?

10. Rechenprozesse (Tasks)

Auf dem Prozessrechner wird ein Programm abgearbeitet, das den industriellen Prozess steuert. In aller Regel ist das Programm als Anwendung auf dem Betriebssystem des Rechners realisiert. Innerhalb des Betriebssystems läuft das Programm als Rechenprozess. Zur Abgrenzung gegenüber dem zu steuernden industriellen Prozess wird im Folgenden für das laufende Programm anstelle des Begriffs „Prozess“ im Sinne von „Rechenprozess“ die Bezeichnung „Task“ verwendet. Hiermit soll ausgedrückt werden, dass die Anwendung eine der Abläufe (Tasks) darstellt, die das Betriebssystem verwaltet. Sinngemäß wird unter „Multi-Tasking“ die quasi-parallele Ausführung der einzelnen Tasks durch das Betriebssystem verstanden.

Frage 1: Kontextumschaltung. Wenn das Betriebssystem zwischen Tasks wechselt, muss die unterbrochene Task die Registerinhalte des Prozessors und den Stand der Abarbeitung ihres Programms (Stapelzeiger, engl. stack pointer) im Arbeitsspeicher ablegen (den sogenannten Kontext der Task). Die aktivierte Task muss ihre Registerinhalte und ihren Stapelzeiger aus dem Arbeitsspeicher laden. Mit dem Wechsel ist also eine sogenannte Kontextumschaltung (engl. context switch) verbunden. Erläutern Sie das Abarbeiten eines Programms mit Hilfe des Stapelspeichers (engl. stack) und Stapelzeigers (Beispiele: Aufruf einer Funktion, Rekursion, Sprung in ein Unterprogramm). Was befindet sich im Prozessor? Was befindet sich im Arbeitsspeicher? Welcher Zeitaufwand ist mit der Kontextumschaltung verbunden?

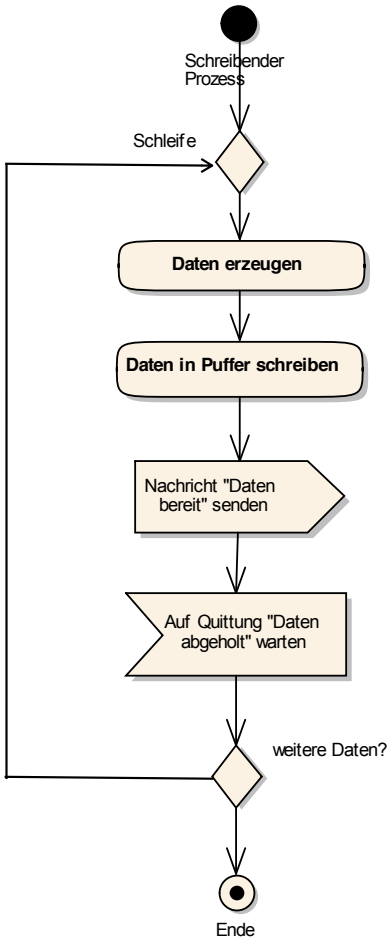
Frage 2: Adressraum. Eine Task innerhalb des Betriebssystem hat einen eigenen Adressraum im Arbeitsspeicher. Welches sind die Vorteile und Nachteile eines eigenen Adressraums?

Frage 3: Unterprozesse. Innerhalb einer Task lässt sich ein Unterprozess bzw. Prozessfaden (engl. thread) starten. Der Unterprozess besitzt den gleichen Adressraum wie die übergeordnete Task. Welches sind die Vorteile und Nachteile?

Frage 4: Inter-Task-Kommunikation (Inter-Prozess-Kommunikation). Wie können Rechen-prozesse miteinander kommunizieren, um beispielsweise Statusmeldungen oder Daten untereinander auszutauschen? Was ist die Aufgabe des Betriebssystems?

11. Datenpuffer

Zum Austausch von Daten zwischen zwei Tasks wird vom Betriebssystem als Datenpuffer ein Speicherbereich zur Verfügung gestellt, auf den beide Tasks schreibend oder lesend zugreifen können. Die folgende Abbildung zeigt den Ablauf innerhalb Task 1.



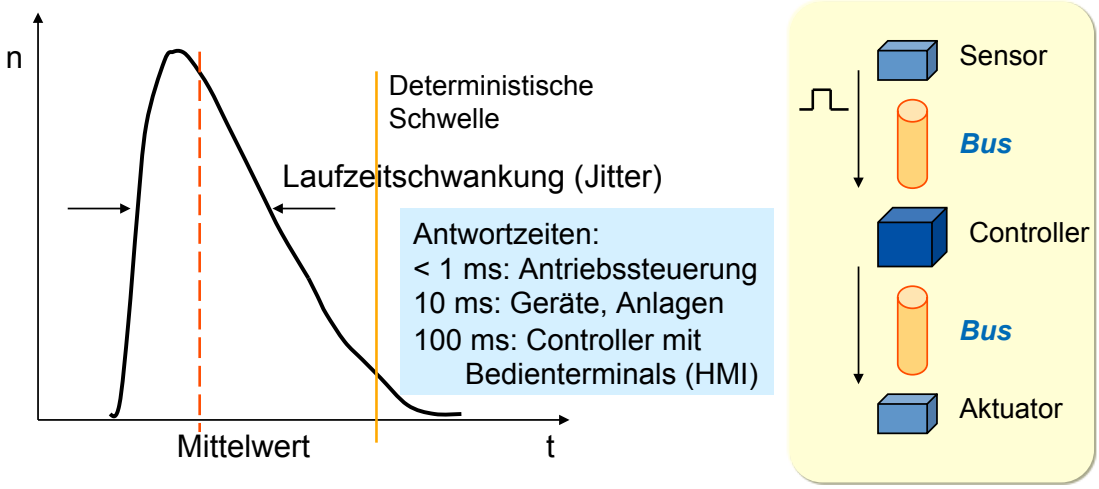
Frage 1: Ergänzen Sie das Diagramm in der Abbildung um den zugehörigen Ablauf in Task 2.

Frage 2: Wodurch werden beide Prozesse synchronisiert? Welche Rolle spielt das Betriebs-system hierbei?

12. Antwortzeiten

Antwortzeiten realer Systeme unterliegen statistischen Schwankungen. Möchte man beispielsweise die Zeit messen, die Kunden in einem Supermarkt verbringen, so kann man beispielsweise Minutenintervalle definieren und erfassen, wie viele Minuten Kunden jeweils im Markt verbringen. Man wird feststellen, dass es zwar eine mittlere Verweildauer gibt, einige Kunden aber

auch schneller wieder auftauchen, und viele auch deutlich länger benötigen. Trägt man die Häufigkeit der Verweildauer über der Zeit auf, so erhält man eine Häufigkeitsverteilung (Histogramm) wie in der Abbildung unten gezeigt.



Frage 1: Die Verteilung kann man durch folgende Kenngrößen definieren: Mittelwert (Erwartungswert) und Varianz. Was genau beschreiben beide Begriffe?

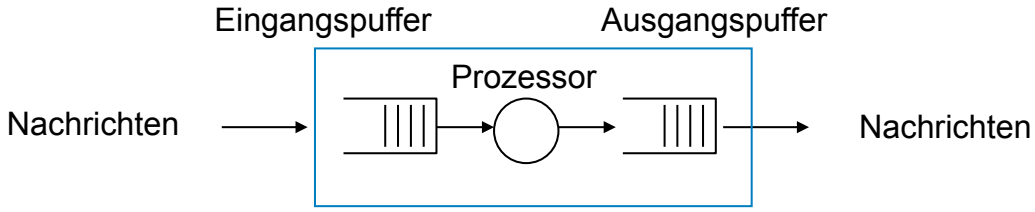
Frage 2: Welchen Einfluss haben konstante Verzögerungen und Laufzeitschwankungen der Messwerte und Steuereingriffe auf eine Steuerung?

Frage 3: Welche Ursachen gibt es für Laufzeitschwankungen auf (1) im Prozessor, (2) im Feldbus? Wie lassen sich Laufzeitschwankungen verringern? Lassen sich Laufzeit-schwankungen komplett abstellen?

Frage 4: Für ein Echtzeitsystem werden definierte Antwortzeiten gefordert (deterministische Schwelle in der Abbildung). Hierfür werden beispielsweise Sensoren vom Prozessrechner in einem definierten Zeitintervall zyklisch abgefragt. Was geschieht, wenn die Antwort eines Gerätes nicht innerhalb des geforderten Intervalls eintrifft?

13. Systemauslastung

Einen Prozessor kann man wie in der folgenden Abbildung darstellen als Maschine, die Nachrichten empfängt, verarbeitet und weiterleitet. Hierbei sind folgende Begriffe definiert: (1) die Ankunftsrate der Nachrichten, (2) die Verarbeitungsrate (Service Rate) des Prozessors. Das Verhältnis aus Ankunftsrate und Service-Rate wird (3) als Systemauslastung definiert. Hierbei wird angenommen, dass die Nachrichten mit einer Verteilung wie in der letzten Aufgabe um die mittlere Ankunftsrate schwanken.



Frage 1: Stellen Sie sich den Prozessor als Kasse in einem Supermarkt vor. Die Kassiererin kann 1 Kunden pro Minute abfertigen (Service Rate). Der Filialleiter kennt die mittlere Ankunftsrate der

Kunden zur Haupteinkaufszeit und kann hinreichend viele Kassen aufstellen. Der Filialleiter möchte wenigstens 100% Systemauslastung erzielen. Was ist die Konsequenz? Warum?

Frage 2: Wodurch kann (1) bei einem Prozessrechner und (2) in einem Feldbus Engpässe (im Sinne von langen Wartezeiten) vermeiden? Lassen sich sporadische Engpässe komplett ausschliessen?

14. Einplanung und Einlastung (Scheduling und Dispatching)

Das Betriebssystem teilt einzelne Tasks zur Laufzeit zur Abarbeitung ein (siehe Manuskript Abschnitt 3.3).

Frage 1: Erläutern Sie die Begriffe „Single-Tasking“ und „Multi-Tasking“ im Zusammenhang mit einem Betriebssystem.

Frage 2: Was versteht man unter präemptiver Einplanung (engl. pre-emptive scheduling)?

Frage 3: Welchen Einfluss haben Ereignisse (Alarme, Timer) auf die Einteilung? Welche Rolle spielen Prioritäten?

Frage 4: Wozu sind periodische Tasks geeignet? Haben periodische Tasks eine Priorität? Wie werden periodische Tasks angestossen?

15. Taskzustände und Task Control Block

Damit eine Task eingeplant werden kann, muss sie im Zustand „ablaufbereit“ sein. Tasks, die beispielsweise auf Eingabe- oder Ausgabeoperationen warten, sind nicht ablaufbereit (siehe Manuskript Abschnitt 3.3.3).

Frage 1: Erläutern Sie die Zustände „ruhend“, „laufend“, „wartend“ und „ablaufbereit“.

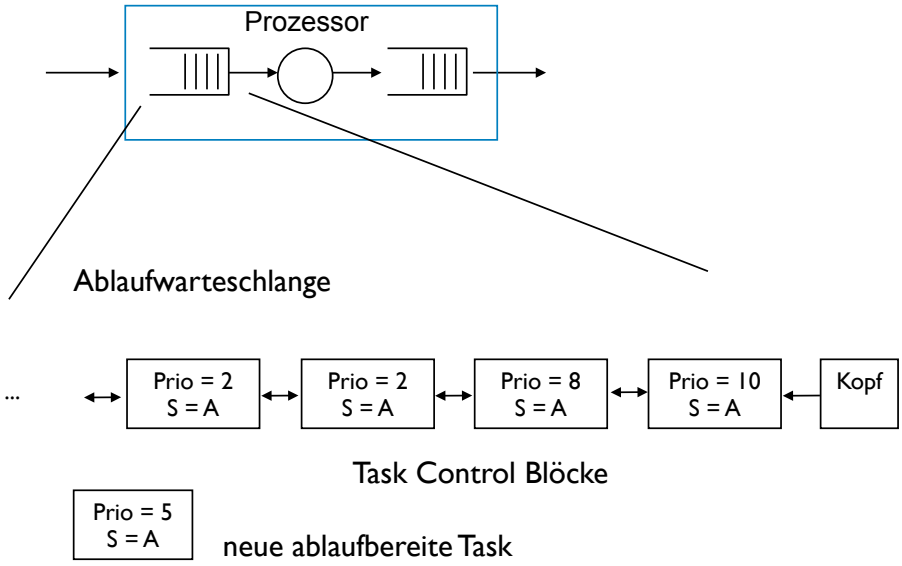
Frage 2: In welchem Zustand muss eine Task sein, damit sie in einer Warteschlange zum Ablauf eingeplant werden kann?

Frage 3: Welche weitere Information außer dem Ablaufzustand der Task muss das Betriebs-system haben, damit die Task in der Warteschlange richtig eingeplant werden kann?

Frage 4: Welche weitere Information ist zum Start der Task erforderlich?

16. Ablaufwarteschlange

Die folgende Abbildung zeigt eine Ablaufwarteschlange zu dem Zeitpunkt, als durch ein Ereignis eine weitere Task ablaufbereit geworden ist.



Frage 1: Beschreiben Sie, wie das Betriebssystem die neue Task in die Warteschlange einreiht.

Frage 2: Nach welchen Kriterien lassen sich Prioritäten für periodische Tasks vergeben?

17. Nebenläufige Prozesse

Aufgaben im Prozess, die parallel ausführbar sind, lassen sich durch nebenläufige Tasks steuern. Idealerweise sind die nebenläufigen Tasks völlig voneinander unabhängig. In der Realität konkurrieren Tasks allerdings um gemeinsame Betriebsmittel, bzw. sie tauschen Daten untereinander aus. Daher sind in der Praxis Mechanismen zur Kooperation erforderlich. Aufgabe 11 zeigte bereits eine Kooperation von Tasks zum Austausch von Daten.

Frage 1: Zeigen Sie, wie sich der im Manuskript in Bild 4.9 gezeigte Konflikt mit Hilfe eines Tickets bzw. einer Marke (Token) lösen lässt. Hierbei symbolisiert der Besitz des Tokens das Fahrtrecht bzw. Recht auf den Zugriff auf das gemeinsame Betriebsmittel. Erläutern Sie die Begriffe „kritischer Abschnitt“ und „gegenseitiger Ausschluss“.

Frage 2: Im Strassenverkehr ist das Zugriffsrecht auf die Kreuzung durch die Regel „rechts vor links“ gelöst. Beschreiben Sie die Funktion dieser Regel in verschiedenen Situationen mit Hilfe eines Tokens.

18. Petri-Netze

Petri-Netze verwenden Marken als Token zur Darstellung des Zeitverhaltens neben-läufiger Aktivitäten. Die Marke kennzeichnet hierbei den Stand innerhalb eines Kontrollflusses. Die Darstellung verwendet folgende Symbole: Stellen (Ovale), Transitionen (Rechtecke), sowie Pfeile (für Übergänge zwischen Stelle und Transition bzw. Transition und Stelle). Dabei gelten folgende Spielregeln:

- Zustandsänderungen erfolgen durch Schalten von Transitionen.
- Eine Transition kann schalten, wenn sich an allen eintreffenden Pfeilen wenigstens eine Marke befindet.
- Durch das Schalten wird jeweils eine Marke pro eintreffender Pfeil entfernt und jeweils eine Marke zu den ausgehenden Pfeilen hinzugefügt.

Frage 1: Führen Sie ein Markenspiel für das Netz in Bild 4.18 des Manuskriptes aus.

Frage 2: Welche Unterschiede ergeben sich in den beiden Darstellungen des Bildes 4.18?

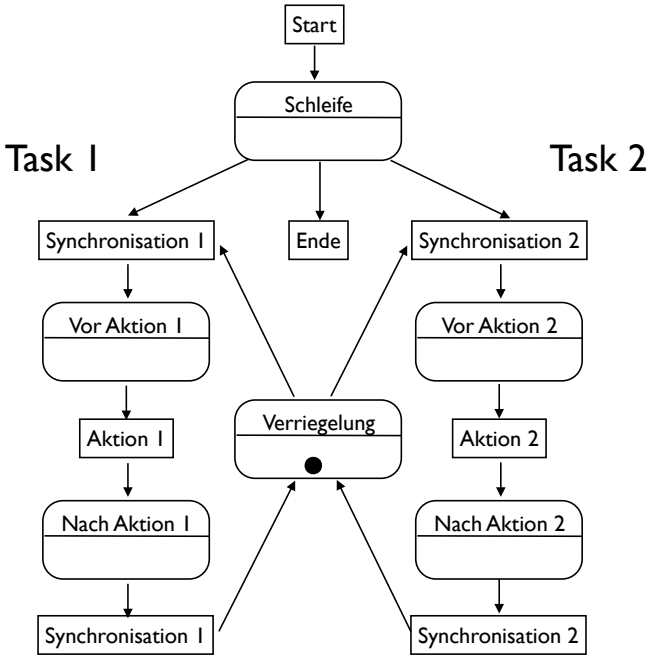
19. Konflikte in Petri-Netzen

Transitionen stehen im Konflikt, wenn sie mindestens eine Stelle teilen und beide Transitionen schalten können. Zur Lösung von Konflikten müssen gesonderte Regeln vereinbart werden, wie z.B. die Vergabe von Prioritäten.

Frage 1: Erläutern Sie den im Manuskript in Bild 4.19 gezeigten Konflikt durch ein Markenspiel.

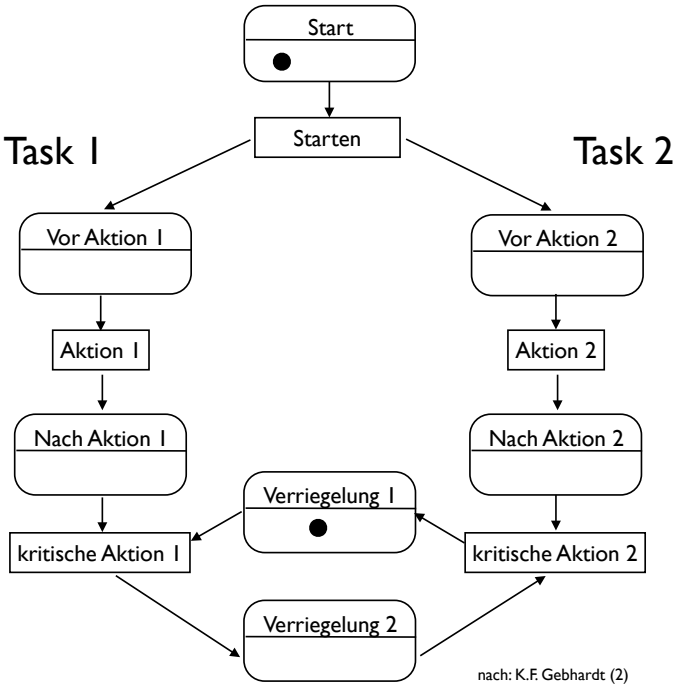
Frage 2: An welcher Stelle findet sich im Manuskript in Bild 4.20 ein Konflikt? Erläutern Sie die im Manuskript in Bild 4.20 gezeigte Lösung des Konflikts durch ein Markenspiel. Erläutern Sie die Begriffe „kritischer Abschnitt“ und „gegenseitiger Ausschluss“.

Frage 3: Erläutern Sie das in der folgenden Abbildung gezeigte Netz. Bemerkungen: (1) sie Stellen wurden als Zustände dargestellt und benannt, (2) eine Transition ohne Eingang (hier: Start) erzeugt Marken, (3) eine Transition ohne Ausgang (hier: Ende) vernichtet Marken. Wozu dient diese Konstruktion in einem Programm? (Hinweis: Am Ende der Übungen finden sich leere Spielfelder für das Markenspiel).



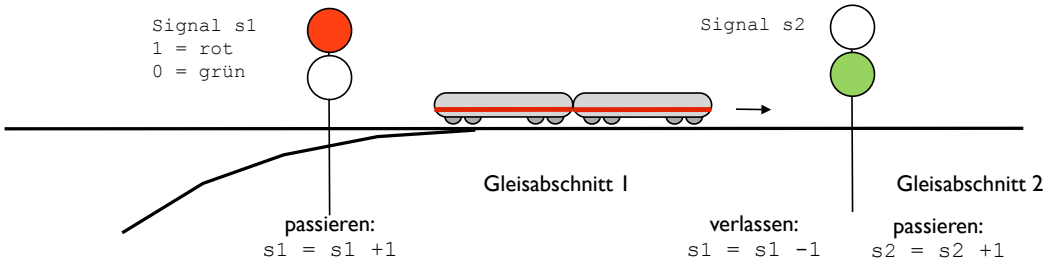
nach: K.F. Gebhardt (2)

Frage 4: Erläutern Sie das in der folgenden Abbildung gezeigte Netz. Worin liegen die Unterschiede zu dem Diagramm aus Frage 3?



20. Stellwerk

Im Schienenverkehr sorgt eine Verriegelung dafür, dass in einen bereits befahrenen Gleisabschnitt kein weiterer Zug einfahren kann. Hierzu wird beim Passieren eines Zugs das Eingangssignal des Gleisabschnitts auf rot geschaltet, beim Verlassen des Gleisabschnitts wird das Eingangssignal wieder auf Grün geschaltet. Die folgende Abbildung illustriert das Prinzip.



Frage 1: Wodurch wird der Zustand des jeweiligen Gleisabschnittes abgebildet? Wie erzeugt ein Zug als Task eine Zustandsänderung? Worauf wartet ein weiterer Zug, bevor er in einen Gleisabschnitt einfahren kann?

Frage 2: Wie liesse sich das Verfahren für von verschiedenen Tasks benutze gemeinsame Betriebsmittel verwenden?

21. Multi-Tasking

Bild 5.1 im Manuskript zeigt die Anordnung aus Frage 9 aus Sicht der beteiligten Tasks: die Steuerung oder Regelung erfolgt durch verteilte, miteinander gekoppelte Prozesse.

Frage 1: Wie erfolgt die Kommunikation zwischen Tasks auf dem gleichen Rechner?

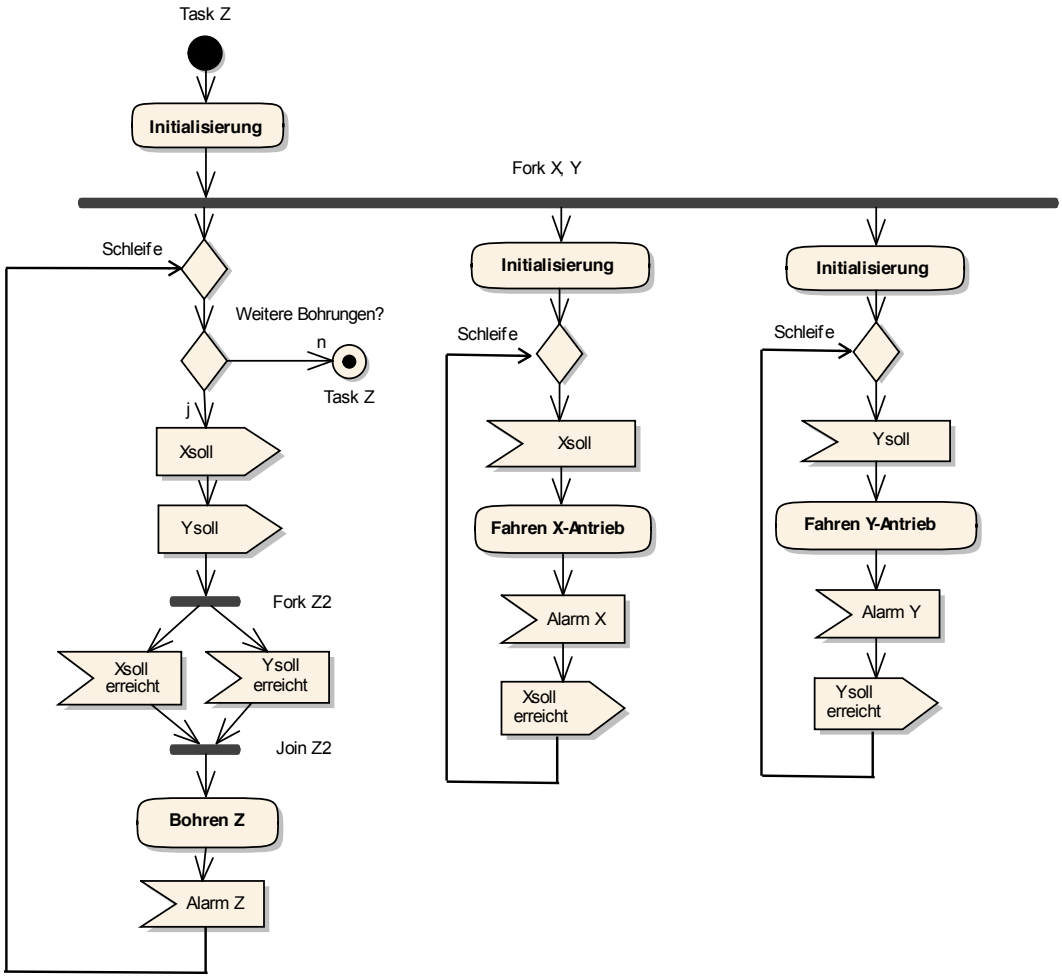
Frage 2: Wie erfolgt die Kommunikation zwischen Tasks auf unterschiedlichen Rechnern?

Frage 3: Wie lassen sich die Synchronisationspunkte zwischen Tasks in einem Aktivitäts-diagramm darstellen?

Frage 4: Welche Rolle spielt das Betriebssystem bei der Kommunikation zwischen Tasks? Welche Rolle spielt das Betriebssystem bei der Kommunikation zwischen Tasks und Betriebsmitteln?

22. Bohrautomat

Folgendes Aktivitätsdiagramm zeigt den Ablauf des Bohrautomaten aus Abschnitt 5.2.2 des Manuskripts.



Frage 1: Erläutern Sie den grundsätzlichen Ablauf.

Frage 2: An welchen Stellen finden sich Synchronisationspunkte zwischen nebenläufigen Tasks? Wo kommen die Nachrichten für die Alarme her?

Frage 3: Erläutern Sie die Unterschiede zwischen Verzweigungen (Fork) bzw. Zusammen-führungen (Join) und Synchronisation durch Nachrichten.

Frage 4: Welche Unterschiede bestehen zur Darstellung in Bild 5.4 des Manuskripts?

23. Systemaufrufe (Service Calls)

Die Kommunikation zwischen Tasks auf einem Rechner wird durch Systemaufrufe ermöglicht, die das Betriebssystem hierfür bereit stellt. Hierbei spielt das Betriebssystem die Rolle einer übergeordneten Task. Bild 5.9 im Manuskript zeigt eine Übersicht über ablaufbereite Anwender-Tasks, sowie die Task des Betriebssystems.

Frage 1: Bei einem Multi-Tasking system konkurrieren alle Tasks um die CPU. Erläutern Sie, wie die in der Abbildung gezeigten Tasks quasi-parallel abgearbeitet werden.

Frage 2: Was geschieht beim Eintreffen eines Alarms?

24. Ereignisse und Zustandswechsel

Bild 5.12 im Manuskript zeigt ein Zustandsdiagramm für Tasks in einem Multi-Tasking System.

Frage 1: Welcher Task-Zustand ist charakteristisch für ein Multi-Tasking System?

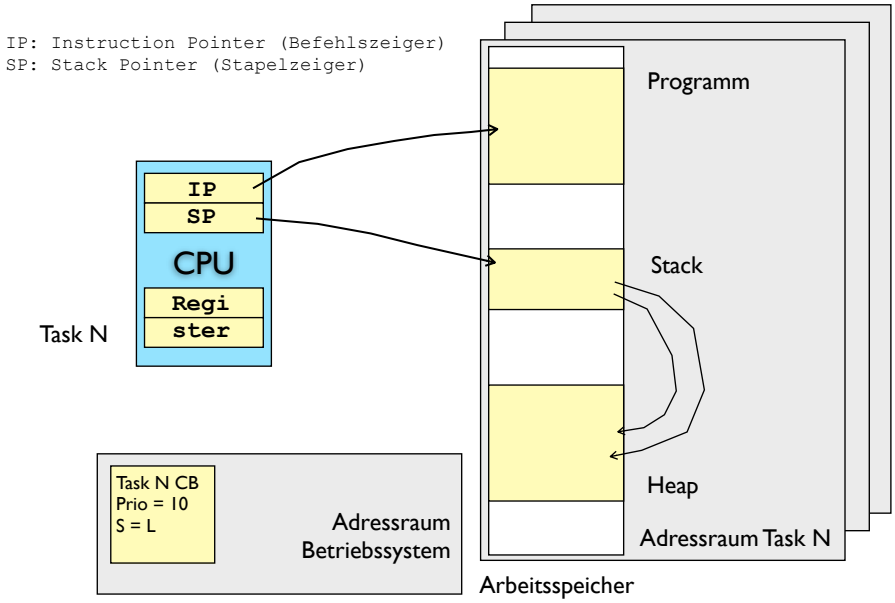
Frage 2: Nach welchen Kriterien schaltet das Betriebssystem eine Task vom Zustand „laufend“ in den Zustand „ablaufbereit“ und zurück?

Frage 3: Wann finden generell Zustandswechsel statt? Wann finden die im Diagramm gezeigten jeweiligen Zustandswechsel statt?

Frage 4: Wie wird eine Task erzeugt bzw. gelöscht?

25. Task Kontext

Eine laufende Task hat taskspezifische Informationen auf der CPU gespeichert (wie z.B. den Befehlszeiger, den Stapelzeiger, sowie CPU-Register). Beim Wechsel aus dem laufenden Zustand muss dieser Task-spezifische Kontext in den Arbeitsspeicher gerettet werden. Umgekehrt wird bei einer Kontextumschaltung der Kontext der in den laufenden Zustand wechselnden Task in die CPU geladen. Folgende Abbildung illustriert den Kontext der Task.



Frage 1: Erläutern Sie die in der Abbildung gezeigten Komponenten und die Zusammenhänge (siehe auch Manuskript Abschnitt 5.4.2).

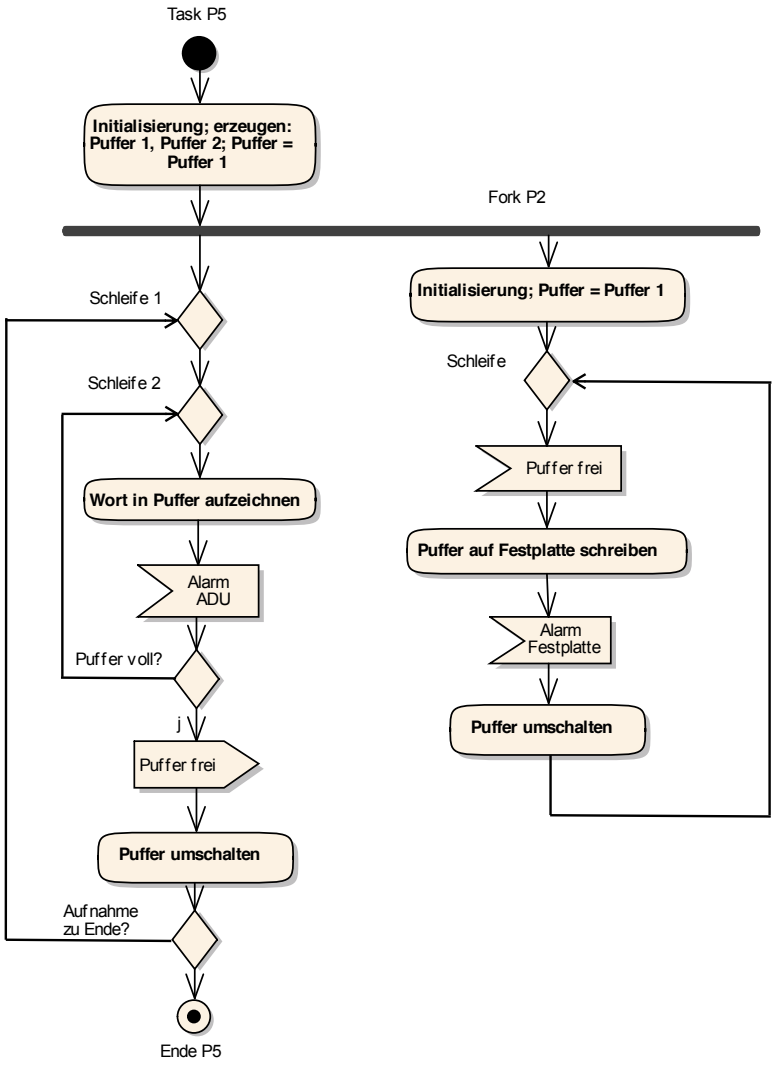
Frage 2: Zeigen Sie den Zustand nach Auslagerung (Kontext-Sicherung) des Task-spezifischen Kontextes. Wohin wird der Task-spezifische Kontext aus der CPU verlagert? Welche Informationen erhält das Betriebssystem für den Task Control Block (TCB)?

Frage 3: Wie wird der Kontext der neu anlaufenden Task aktiviert (Kontext-Restaurierung)? Zeigen Sie die Zustände der ablaufbereiten und der laufenden neuen Task.

Frage 4: Wie kann das Betriebssystem Tasks identifizieren? Wie kann das Betriebssystem Tasks erzeugen und löschen?

26. Wechselpuffer

In Abschnitt 5.4.4 des Manuskripts erfolgt eine Sprachaufzeichnung mithilfe eines Wechselpuffers, der von einer schreibenden Task befüllt wird, während eine lesende Task die Aufzeichnung auf Festplatte speichert. Da die lesende Task P2 schneller arbeitet als die schreibende Task P5, sollte die in Bild 5.20 gezeigte Anordnung funktionieren. Die folgende Abbildung zeigt ein Aktivitätsdiagramm der beiden Tasks.



Frage 1: Erläutern Sie den im Diagramm gezeigten Ablauf.

Frage 2: Wodurch werden beide Prozesse synchronisiert?

Frage 3: Versuchen Sie, mit den im Manuskript genannten Annahmen (5.3) für die Geschwin-digkeit beider Prozesse das Zeit-Balken-Diagramm in Bild 5.23 im Manuskript nach zu vollziehen.

Frage 4: Diskutieren Sie die unter Bild 5.23 im Manuskript genannten Maßnahmen zur Lösung der beiden Echtzeitanforderungen.

27. Warten bei Systemaufrufen

In Abschnitt 5.5 im Manuskript sind einige Alternativen für das Verhalten einer Task beschrieben, die auf ein Ereignis wartet. Für die Interaktion mit Ereignissen stellt das Betriebssystem Funktionen (Service Calls) für das Anwenderprogramm (Task) zur Verfügung.

Frage 1: In welchen Zustand versetzt das Betriebssystem eine Task, die auf ein Ereignis wartet?

Frage 2: Was geschieht, wenn das Ereignis schließlich eintrifft?

Frage 3: Was kann man für den Fall vorsehen, dass das Ereignis ausbleibt?

Frage 4: Geben Sie folgenden Ablauf durch ein Aktivitätsdiagramm wieder: Task 1 arbeitet in einer Schleife, um Nachrichten aus einer vom Betriebssystem bereit gestellten Mailbox zu lesen und in einer Datei zu protokollieren.

28. Rolle des Betriebssystems

Die Abschnitte 5.6 und 5.7 im Manuskript beschreiben Dienstleistungen des Betriebs-systems für die Verwaltung der Tasks, die Verwaltung der Betriebsmittel, sowie für die Kommunikation zwischen den Tasks.

Frage 1: Was leistet das Betriebssystem für die Verwaltung der Tasks?

Frage 2: Was leistet das Betriebssystem für die Verwaltung der Betriebsmittel?

Frage 3: Was leitet das Betriebssystem für die Kommunikation zwischen den Tasks?

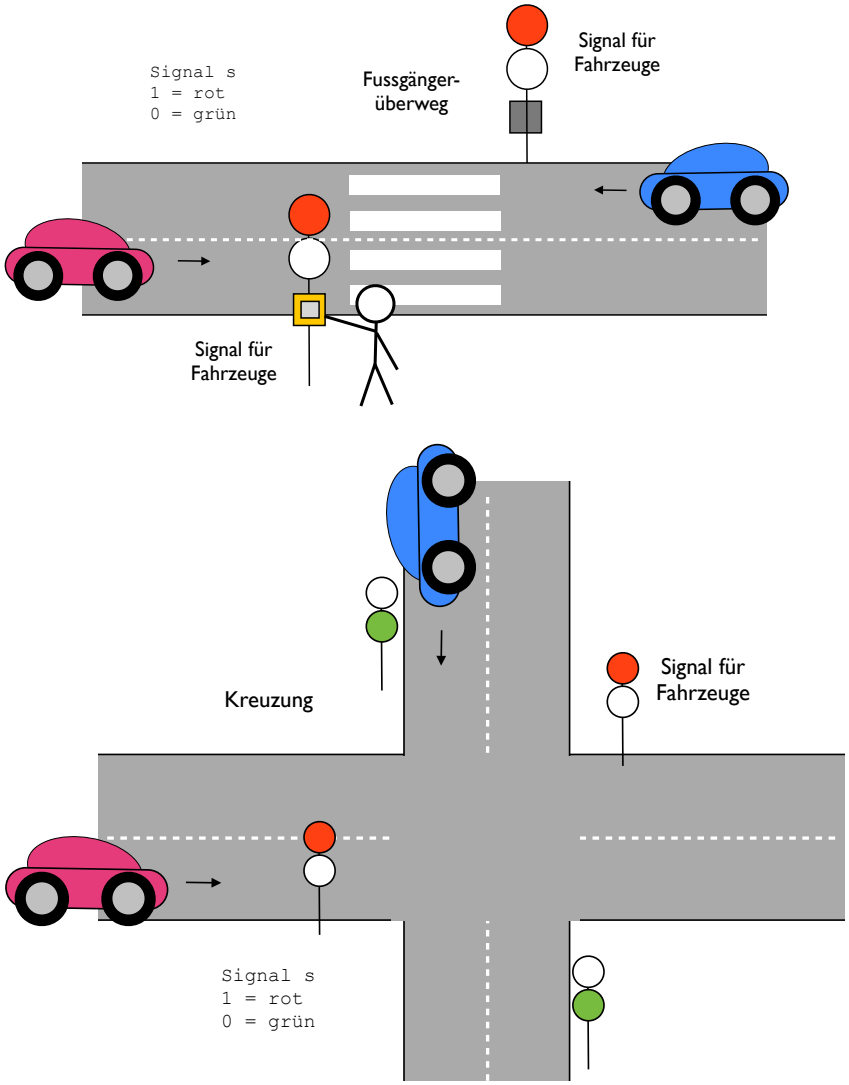
Frage 4: Was wird unter Kernobjekten verstanden?

29. Semaphore

Wenn Tasks um Betriebsmittel konkurrieren, sind Mechanismen erforderlich, um Konflikte zu lösen. Als Beispiel sei ein Drucker genannt, den mehrere Tasks benutzen möchten. Eine Möglichkeit besteht darin, den Belegungszustand des Betriebsmittels durch ein Signal zu kennzeichnen. Dieses Signal (bzw. dieses Semaphor) wird vom Betriebssystem verwaltet. Der Zustand des Signals wird durch Systemaufrufe an die Tasks kommuniziert. Hierbei lässt sich der Belegungszustand abfragen, sowie das Signal und das Betriebsmittel durch die Tasks belegen und wieder freigeben. Die Methode entspricht den Signalen im Strassenverkehr bzw. Schienenverkehr für gemeinsam genutzte Betriebsmittel wie beispielsweise Füßgänger-übergänge, Kreuzungen, Zufahrten oder Gleisabschnitte. Im Manuskript finden sich Semaphore in Abschnitt 5.8.

Frage 1: Was genau kennzeichnet ein Semaphor? Wie ist das Semaphor mit dem Betriebsmittel verbunden?

Frage 2: Beschreiben Sie den Einsatz eines Signals für die in der folgenden Abbildung gezeigte Konfliktsituationen.

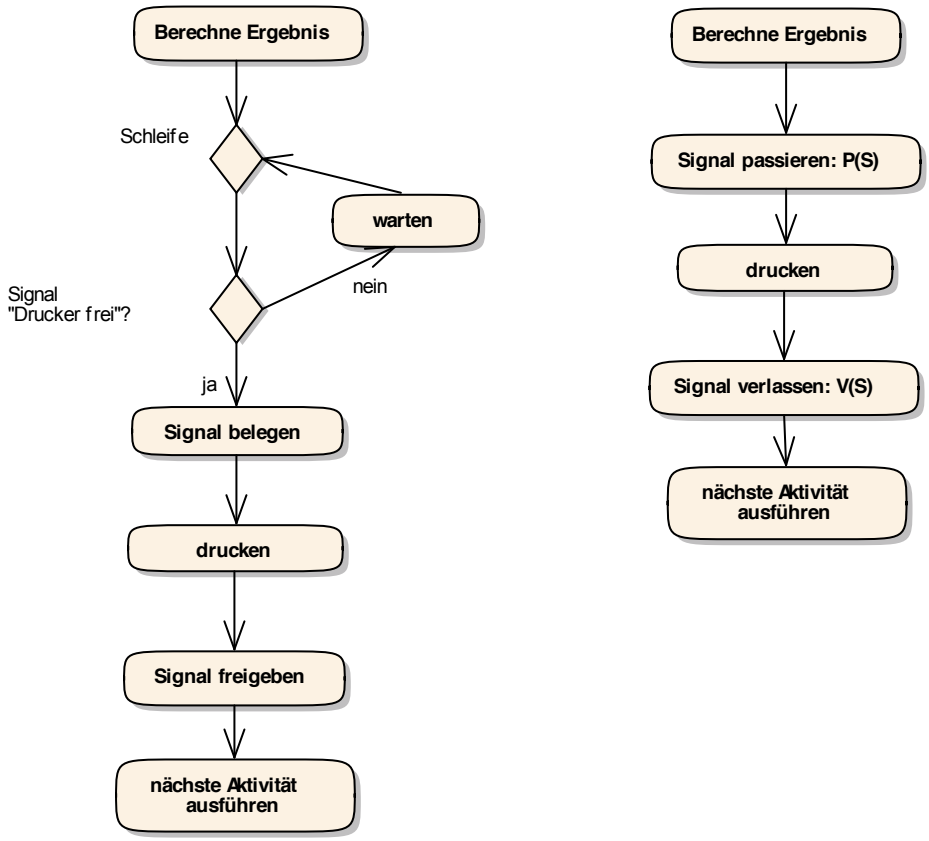


Frage 3: Wie kann in der oben abgebildeten Situation der Konflikt vermieden werden, dass beide Fahrzeuge (bzw. Tasks) gleichzeitig das Betriebsmittel Kreuzung beanspruchen? Übersetzen Sie die Situation für folgenden Ablauf innerhalb der Tasks: (1) prüfen, ob Betriebsmittel frei ist, (2) Signal auf belegt schalten (wenn Betriebsmittel frei), (3) Betriebsmittel beanspruchen, (4) nach Freigabe des Betriebsmittels das Signal wieder auf frei schalten. Welche Abläufe dürfen nicht durch das Multi-Tasking unterbrochen werden?

Frage 4: Erläutern Sie folgende Begriffe: (1) atomare Abläufe, (2) kritischer Bereich, (3) gegenseitiger Ausschluss.

30. Systemaufrufe (Service Calls) für Semaphore

Folgende Abbildung zeigt den Ablauf für die Nutzung eines Betriebsmittels aus Sicht der Task (siehe auch Bild 5.31 im Manuskript). Die linke Seite der Abbildung zeigt den Ablauf unter Nutzung eines Signals. Die rechte Seite zeigt eine Implementierung mit Systemaufrufen.



Frage 1: Erläutern Sie den auf der linken Seite des Diagramms gezeigten Ablauf.

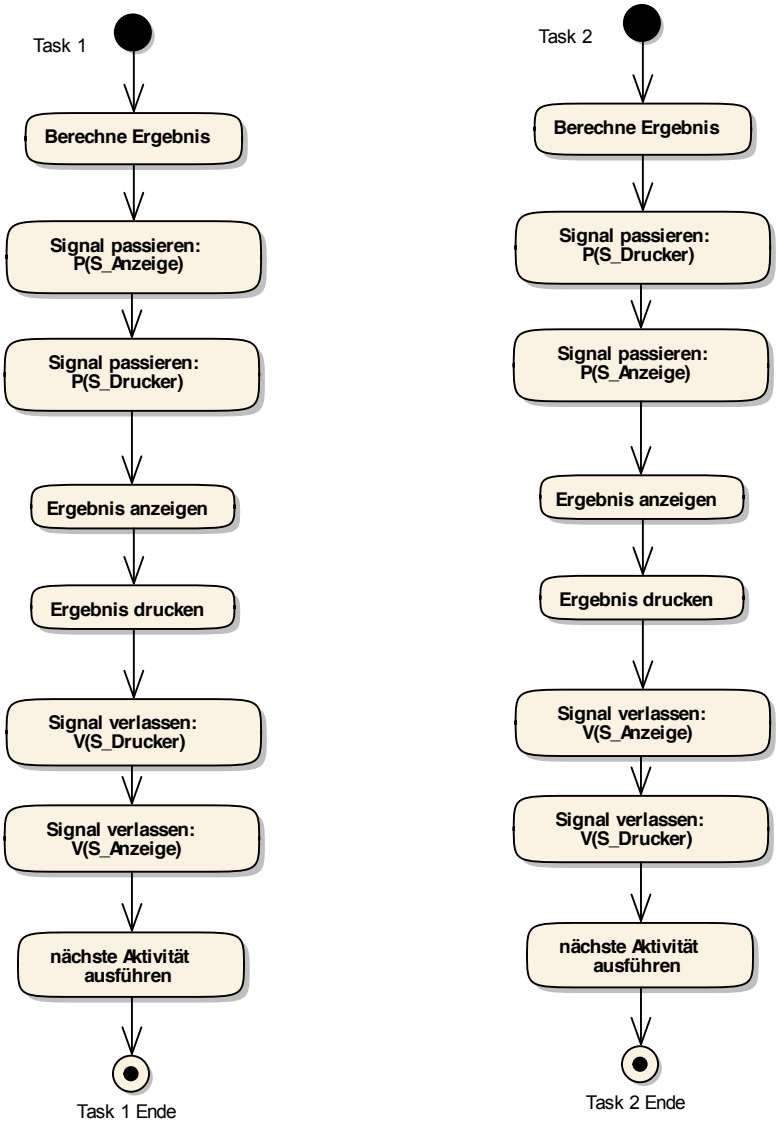
Frage 2: Welche Abschnitte dieses Ablaufs dürfen nicht durch Multitasking unterbrochen werden? Warum nicht? Hinweis: Stellen Sie sich eine weitere Task mit dem gleichen Ablauf vor.

Frage 3: Erläutern Sie die Inhalte der beiden Systemaufrufe P(S) und V(S) auf der rechten Seite der Abbildung. Was ist die Rolle des Betriebssystems?

Frage 4: Welcher Systemaufruf enthält einen unteilbaren Ablauf? Welche Möglichkeiten hat das Betriebssystem, diesen Ablauf unterbrechungsfrei zu realisieren?

31. Verklemmung (Deadlock)

Folgendes Diagramm zeigt den Ablauf zweier Tasks. Hierbei treten sogenannte zirkulare Wartebedingungen auf, die zur gegenseitigen Behinderung (Verklemmung) der Tasks führen können.



Frage 1: Erläutern Sie den Ablauf jeder einzelnen Task.

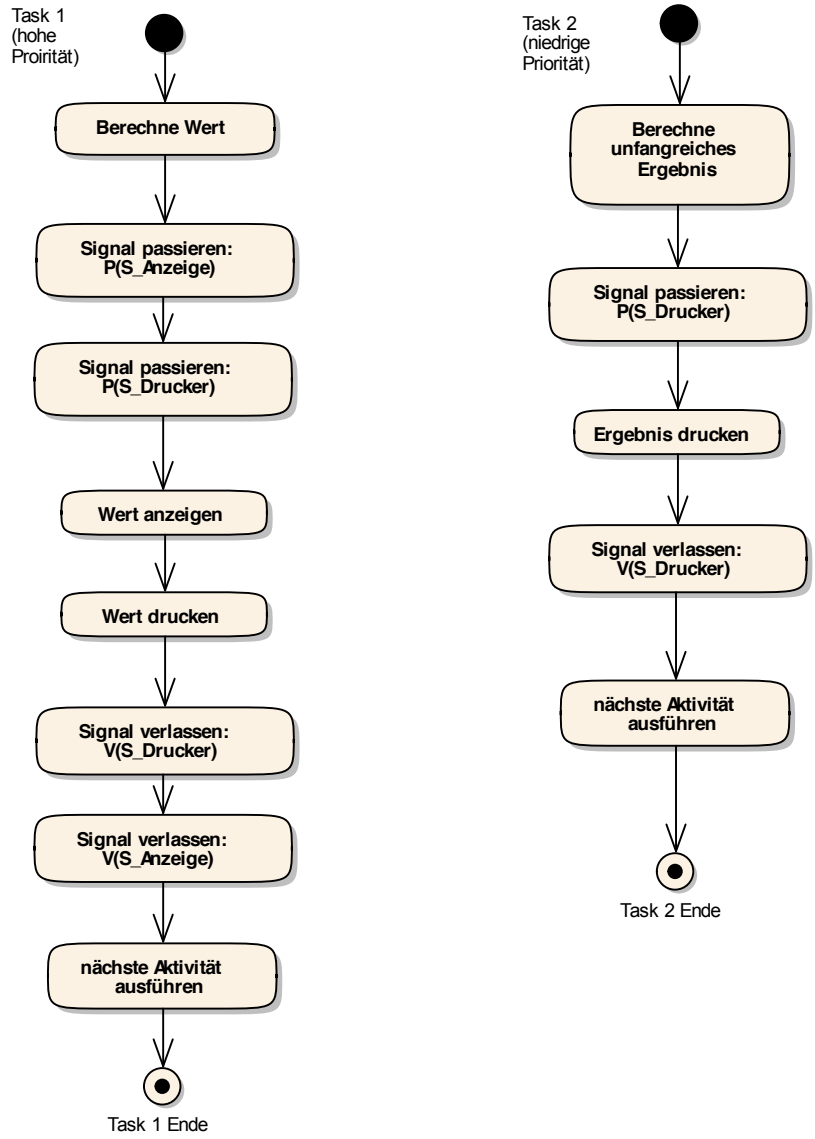
Frage 2: Was kann in einem ungünstigen Fall geschehen, wenn beide Tasks in einem Multi-Tasking System ablaufen? Von welchen Bedingungen hängt der ungünstige Fall ab?

Frage 3: Mit welchen Methoden kann der Anwendungsprogrammierer diesen Fall vermeiden? Hinweis: siehe auch Abschnitt 5.8.4 im Manuskript.

Frage 4: Die Tasks werden in einem Multi-Prozessor System auf unterschiedlichen Prozessor-kernen wirklich parallel ausgeführt. Kann der beschriebene Fall ebenfalls eintreten?

32. Umkehr der Prioritäten (Livelock)

Folgende Abbildung zeigt den Ablauf zweier Tasks in einem Multi-Tasking Betriebs-system. Beide Tasks benutzen einen Drucker, auf dem der Ablauf protokolliert wird. Unter Umständen tritt der Fall ein, dass eine hoch priorisierte Task durch niedriger priorisierte Tasks behindert wird.



Frage 1: Kann es beim Ablauf beider Tasks zu einer Verklemmung kommen?

Frage 2: Kann die niedrig priorisierte Task 2 die höher priorisierte Task 1 behindern?

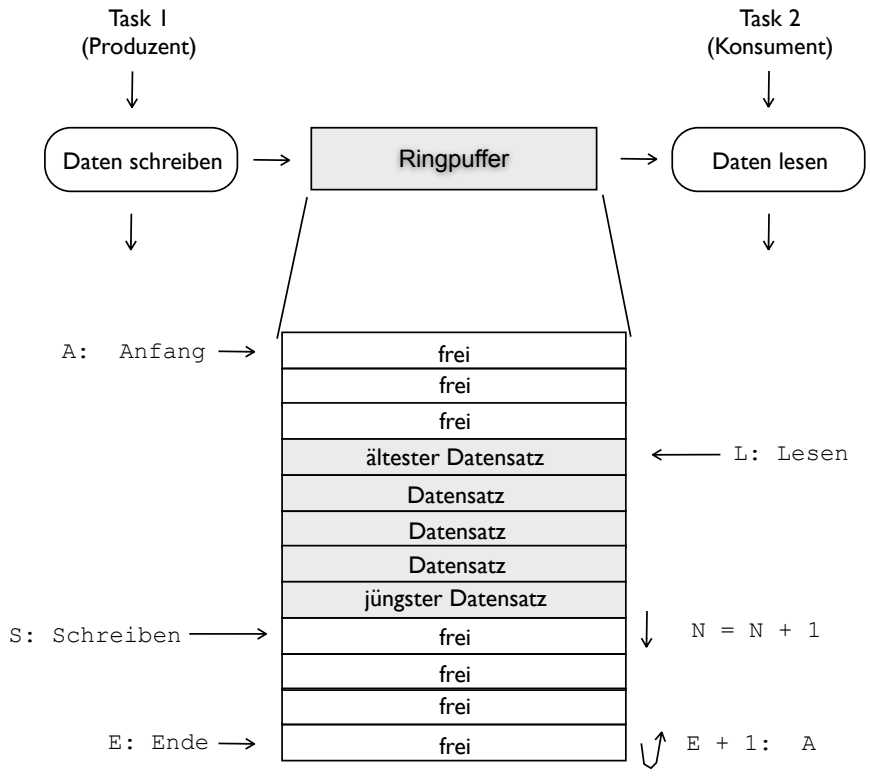
Frage 3: Kann die niedrig priorisierte Task 2 ihrerseits von anderen höher priorisierten Tasks behindert werden? Auch von anderen Tasks mit geringerer Priorität als Task 1? Wie lange wartet Task 1 im ungünstigsten Fall?

Frage 4: Was können Sie als Anwendungsprogrammierer tun, um die Situation zu verbessern?

33. Ringpuffer zur zeitlichen Entkopplung

Task 1 auf einem Multi-Tasking System erzeugt Datensätze (z.B. Koordinaten (x,y,z) für eine Steuerung), die durch eine weitere Task 2 (Steuerung) gelesen werden. Hierbei kann der Konflikt entstehen, dass Daten bereits überschrieben werden, die noch nicht vollständig ausgelesen wurden. Durch einen Ringpuffer soll das Schreiben und Lesen der Daten durch die beiden Tasks zeitlich entkoppelt werden.

Hierbei ist der Ringpuffer, wie in der folgenden Abbildung gezeigt, so realisiert, dass die schreibende Task auf das nächste freie Feld schreibt und den Schreibindex S um eine Position erhöht. Die lesende Task liest den ältesten Datensatz an Position L und erhöht diesen Leseindex L ebenfalls. Von der Endposition E des Ringpuffers aus wird in die Anfangsposition A weiter gezählt. Die Anzahl N der belegten Speicherplätze wird bei jedem Schreibzugriff erhöht und bei jedem lesenden Zugriff herab gezählt.



Frage 1: Unter welcher Bedingung kann Task 1 den nächsten Wert in den Ringpuffer schreiben? Wann muss Task 1 warten?

Frage 2: Unter welcher Bedingung kann Task 2 den nächsten Wert aus dem Ringpuffer lesen? Wann muss Task 2 warten?

Frage 3: Auf welche Weise werden die beiden Tasks durch den Ringpuffer zeitlich entkoppelt? Wie wäre die Größe des Ringpuffer zu bemessen?

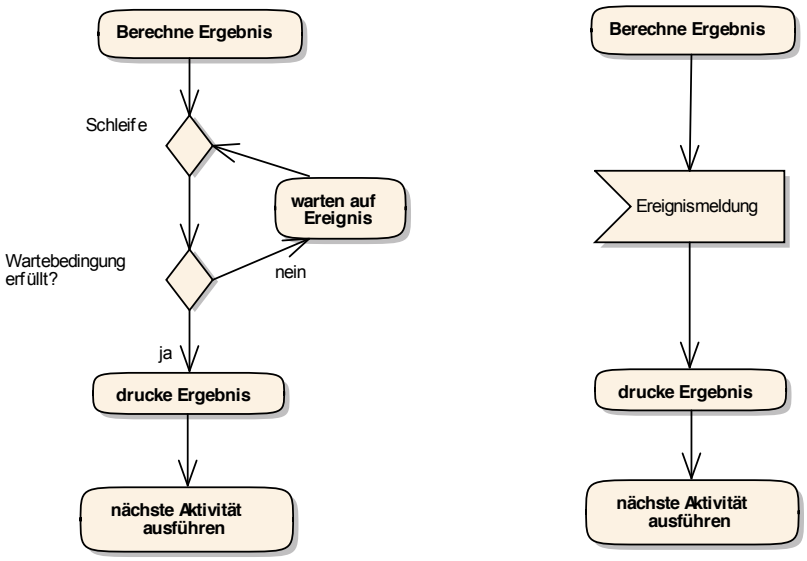
Frage 4: In Bild 5.42 im Manuskript wurde die Synchronisation beider Tasks mit Hilfe zählender Semaphore realisiert, wobei das Semaphor LEER die Anzahl der leeren Felder zählt (bei LEER=0 ist der Ringpuffer voll), und das Semaphor VOLL die Anzahl der vollen Felder zählt (bei VOLL=0 ist der Ringpuffer leer). Es gilt LEER + VOLL = M, der Anzahl der insgesamt vorhandenen Plätze im Ringpuffer. Beschreiben Sie den Ablauf beider Tasks. Hinweis: Was bewirken die Operationen P(S) und V(S) auf dem Semaphor S?

34. Ereignismeldungen (Eventflags)

Zustandsänderungen, Alarme und Ereignisse im Prozess lassen sich mit Hilfe von Ereignismeldungen vom Betriebssystem zu des Tasks kommunizieren. Den Ereignismeldungen

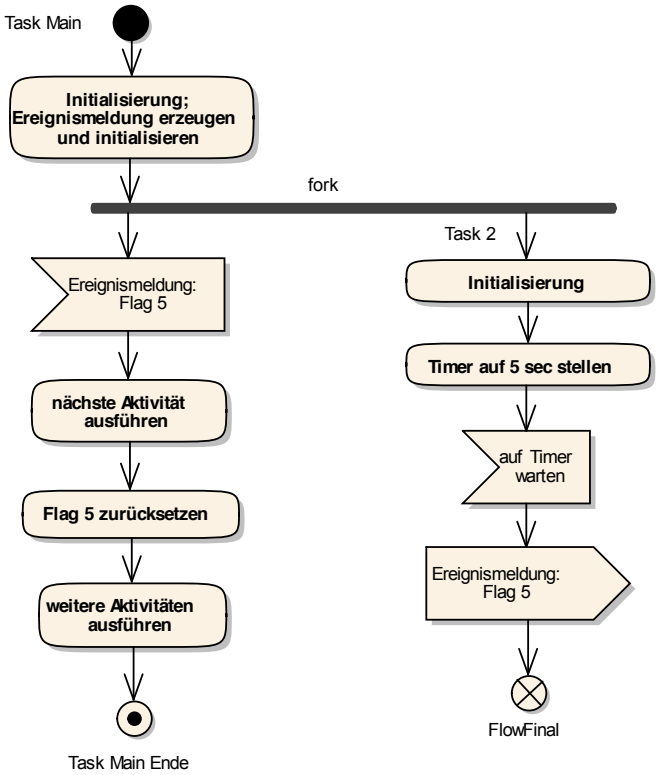
werden hierzu Bits in Ereignisgruppen zugeordnet. Ereignismeldungen lassen sich so auch logisch miteinander verknüpfen (siehe Manuskript Abschnitt 5.9).

Frage 1: Folgender Ablauf beschreibt das Warten auf eine Ereignismeldung (siehe auch Bild 5.53 im Manuskript). Erläutern Sie den Ablauf.



Frage 2: Vergleichen Sie den Ablauf aus Frage 1 mit dem Ablauf in Aufgabe 30 (Semaphor). Worin besteht der Unterschied zwischen Semaphor und Ereignismeldung?

Frage 3: Die folgende Abbildung zeigt Ereignismeldungen zur Synchronisation von Prozessen (siehe auch Bild 5.54 im Manuskript). Erläutern Sie den Ablauf.



Frage 4: Welche Rolle spielen die Prioritäten beider Tasks? Wozu wird die Ereignismeldung wieder zurück gesetzt? Was kennzeichnet der Begriff „einseitige Synchronisation“? Wie lassen sich Ereignismeldungen zur Synchronisation mehrerer Tasks untereinander verwenden? Welche Arten der Synchronisation finden Sie im Manuskript?

35. Inter-Task-Kommunikation

Tasks sind in ihrem eigenem Adressraum gekapselt. Somit kann eine Task mit einer anderen Task nur mit Unterstützung des Betriebssystem korrespondieren, um Abläufe zu synchronisieren, bzw. um Daten auszutauschen. Abschnitt 5.10 im Manuskript beschreibt die wichtigsten Mechanismen der Inter-Task-Kommunikation.

Frage 1: Erläutern Sie die Funktionsweise von Ereignismeldungen (Eventflags) und nennen Sie einige Anwendungsfälle.

Frage 2: Erläutern Sie die Funktionsweise von Postfächern (Mailboxes) und nennen Sie einige Anwendungsfälle.

Frage 3: Erläutern Sie die Funktionsweise globaler Datenbereiche und nennen Sie einige Anwendungsfälle.

Frage 4: Im Fall von Frage 3, was bezeichnen die Begriffe global, statisch bzw. dynamisch?

36. Speicherverwaltung

Damit eine Task ablaufen kann, muss der Programm-Code in den Arbeitsspeicher geladen werden. Ausserdem benötigt die Task zu Arbeiten Arbeitsspeicher für die vorgegebenen und von der Task erzeugten Daten. Zur Verwaltung des Datenspeichers werden in Abschnitt 6 des Manuskripts einige Methoden genannt.

Frage 1: Was versteht man unter statischer Allokation des Datenspeichers?

Frage 2: Wozu dienen Stack und Heap einer Task? Wie werden Stack und Heap allokiert?

Frage 3: Welches sind die Vorteile und Nachteile der dynamischen Allokation?

Frage 4: Was versteht man unter globalen, statischen und lokalen Daten einer Task bzw. einer Funktion (Methode) innerhalb der Task? Was versteht man unter Konstanten und Variablen in einem Programm? Wo werden Konstante und Variable zur Laufzeit des Programms abgebildet?

37. Stapelüberlauf (Stack Overflow)

Frage 1: Was versteht man unter einem Überlauf des Stapels innerhalb einer Task?

Frage 2: Was sind die möglichen Konsequenzen? Hinweis: Verwenden Sie zur Diskussion das Bild aus Aufgabe 25. Hinweis: Die tatsächlichen Konsequenzen sind abhängig vom Prozessortyp und von der Organisation des Arbeitsspeichers (z.B. in welche Richtung der Stapel wächst und welche Daten in dieser Richtung abgelegt sind).

Frage 3: Wodurch kann ein Stapelüberlauf hervorgerufen werden?

Frage 4: In welchem Zusammenhang mit Stapelüberläufen stehen ungeprüfte Eingabefelder in Programmfunktionen (bzw. Bibliotheken)? Beispiel: Eine vom Benutzer geforderte Texteingabe

`get_s(s: string)` wird an eine Variable vorgegebener Länge übergeben, wobei die Länge des tatsächlich eingegebenen Textes nicht überprüft wird.

Frage 5: Erläutern Sie, was beim Aufruf einer rekursiv programmierten Funktion im Stapel geschieht (zum Beispiel: Berechnen von n-Fakultät (n!) durch Aufruf der Funktion `y = f (n)` für den Fall n=3 bzw. für den allgemeinen Fall).

Frage 6: Wie könnte man sich vor Stapelüberläufen schützen?

38. Verwaltung des Heaps

Der sogenannte Heap innerhalb einer Task stellt eine Halde dar für grössere Mengen von Daten. Objekte (bzw. Variablen), die in einer Task erzeugt und initialisiert werden, werden auf Halde gelegt. Hierfür muss für die Halde (den Heap) dynamisch Speicher allokiert werden. In der Regel führt die Task einen Zeiger auf das Datenobjekt (bzw. auf die Variable) als Objektreferenz mit. Die Objektreferenz kann beispielsweise als lokale Variable auf dem Stapel geführt werden.

Frage 1: Was sollte ein ordentlicher Programmierer tun, wenn das erzeugte Objekt (bzw. die Variable) nicht mehr benötigt wird?

Frage 2: Was geschieht, wenn innerhalb einer Task für nicht mehr benötigte Objekte (z.B. durch Nachlässigkeit des Programmierers) zwar die Objektreferenz (der Zeiger auf das Objekt) verschwindet, nicht jedoch das Objekt selbst? Welches sind mögliche Konsequenzen? Bemerkung: Für diesem Fall taucht in der Fachliteratur für unreferenzierte Objekte gelegentlich der Begriff „Object Heaven“ auf. Weitere Begriffe für diesen Fall sind auf Deutsch „Speicherfresser“ bzw. im Englischen „Memory Leak“.

Frage 3: Was kann man tun, um den in Frage 2 beschriebenen Zustand zu verhindern? Hinweis: In einigen Laufzeitumgebungen taucht in diesem Zusammenhang der Begriff „Garbage Collection“ (Müllsammlung) auf.

Frage 4: Was kann geschehen, wenn die Anweisung zur Löschung von Objekten (Destruktor) mehrfach aufgerufen wird? Hinweis: ein ähnlicher Effekt ergibt sich, wenn der allokierte Speicher zu früh freigegeben wird und eine andere Instanz noch auf diesen Bereich zugreift (siehe Abschnitt 6.3.6 im Manuskript). Bemerkung: In der Fachliteratur findet sich hierzu gelegentlich der Begriff „Object Hell“.

39. Unterbrechnungsbehandlung

In Bild 1.21 im Manuskript wurde der grundsätzliche Ablauf bei Auftreten eines Alarmes beschrieben: Der Alarm führt dazu, dass eine Unterbrechungsroutine (UBR) aufgerufen wird, die die laufende Task (PU) verdrängt. Diese Zusammenhänge werden in Abschnitt 7 des Manuskripts näher detailliert. Die Alarmbehandlung ist im Detail abhängig vom Prozessortyp und vom Betriebssystem. Für die Übungen sind nur die generellen Zusammenhänge von Interesse.

Frage 1: Erläutern Sie mit Blick auf Bild 1.21, warum es von Vorteil ist, wenn die Unterbrechungsroutine möglichst kurz ist?

Frage 2: Eine Möglichkeit, die Unterbrechungsroutine (UBR) kurz zu halten, besteht darin, alle Funktionen, die nicht dringend in der Unterbrechungsroutine erforderlich sind, in eine andere Task, die Alarmtask (PA), auszulagern. Beschreiben Sie Funktionen, die in die

Unterbrechungsroutine gehören. Beschreiben Sie Funktionen, die in die Alarmtask gehören. Nennen Sie ein Beispiel (z.B. für eine Task, die auf ein Ereignis aus dem E/A-System wartet).

Frage 3: Erläutern Sie den in Bild 7.2 gezeigten Ablauf. Welches ist die Aufgabe der Unterbrechungsroutine UBR?

Frage 4: Erläutern Sie den in Bild 7.5. gezeigten Ablauf. Welches ist die Aufgabe der Alarmtask PA? Wann ist eine gesonderte Alarmtask nicht erforderlich?

Frage 5: In der Regel führen auf einem Prozessrechner viele Ereignisse zu Alarmen. Wie werden mehrere Alarme behandelt?

Frage 6: Eine alternative zur alarmgetriebenen Behandlung von Ereignissen ist das zyklische Abfragen der Umgebung des Prozessrechners. Erläutern Sie Vorteile und Nachteile beider Verfahren. Nennen Sie einige Beispiele bzw. Einsatzgebiete.

Frage 7: Ein Spezialfall der zyklischen Abfrage wäre das aktive Warten (Abfrage eines Wertes in einer Programm-Schleife). Wie wäre diese Methode einzuordnen?

Frage 8: Beim Wechsel zwischen zwei Tasks erfolgt eine Kontextumschaltung. In Bild 7.15 ist die Kontextumschaltung für den Wechsel von der Task PU in die Unterbrechungsroutine UBR in einer spezielle Prozessorumgebung gezeigt. Erläutern Sie den Ablauf.

40. Mehrfachunterbrechung

Wird eine Unterbrechungsroutine unterbrochen, spricht man von einer Mehrfachunter-brechung. Im Manuskript sind Mehrfachunterbrechungen für eine spezielle Laufzeitumgebung in den Abschnitten 7.6.3 bis 7.6.5 gezeigt.

Frage 1: Welches sind die generellen Merkmale der mehrfachen Unterbrechung?

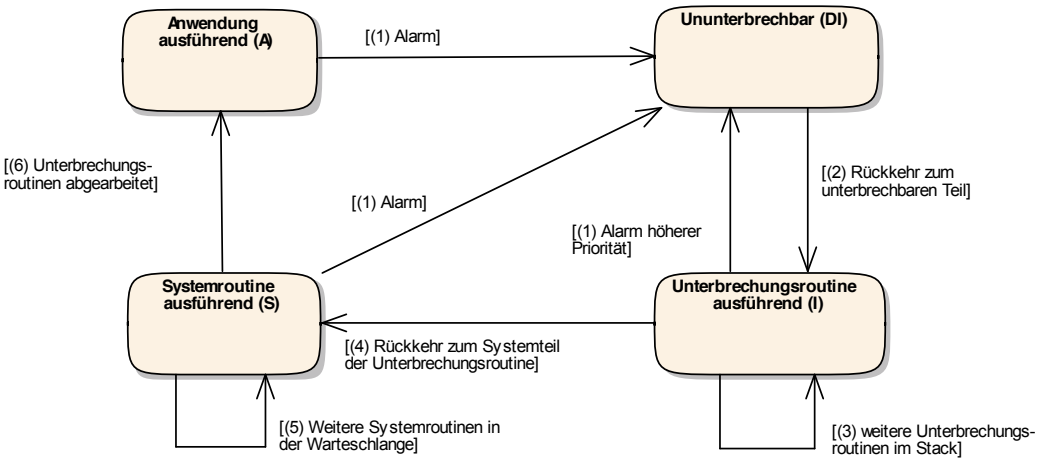
Frage 2: Wann finden Kontextumschaltungen statt?

Frage 3: Wann kann der jeweilige Kontext auf dem gleichen Stack gesichert werden? Welche Konsequenz ergibt sich hieraus für die Rückkehr in die diesbezüglichen Abläufe?

Frage 4: Erläutern Sie den in Bild 7.21 gezeigten speziellen Ablauf.

41. Zustandsdiagramm des Echtzeitbetriebssystems

Folgende Abbildung zeigt ein Zustandsdiagramm für die Unterbrechungszustände des im Manuskript beschriebenen Echtzeitbetriebssystems im Zusammenhang mit der Abarbeitung mehrfacher Unterbrechungen (siehe auch Manuskript Abschnitt 7.6.6 und Bild 7.23). Die einzelnen Zustände sind hierbei: Anwendung ausführend (A), Systemroutine ausführend (S), Ununterbrechbar (DI), sowie Unterbrechungsroutine ausführend (I).



Frage 1: Erläutern Sie das Diagramm und die möglichen Zustandswechsel. Welche Zustände lassen sich durch eintreffende Alarme unterbrechen?

Frage 2: Welche Zustände werden bei Eintreffen eines einfachen Alarms durchlaufen (Startpunkt: Ausführung einer Anwendung)?

Frage 3: Welche Zustände werden im Fall mehrfacher Alarme durchlaufen? Erläutern Sie das Abarbeiten mehrfacher Alarme für den in Bild 7.21 gezeigten Ablauf.

Frage 3: Welches ist für den Ablauf aus Frage 3 der jeweils aktive Stack? Welche Informationen werden für den Ablauf aus Frage 3 jeweils auf dem aktiven Stack abgelegt?

42. Gerätetreiber

Für Eingabegeräte und Ausgabegeräte sind Unterbrechungsroutinen Teil der Geräte-treiber. Der Hersteller der Geräte stellt die Treiber für das jeweils verwendete Betriebssystem zur Verfügung. Der Anwendungsprogrammierer kann somit auf die ihm bekannten Bibliotheken des Betriebssystems zurückgreifen. Die grundsätzliche Struktur der Einbindung in das Betriebssytem ist in Abschnitt 7.9 des Manuskripts gezeigt.

Frage 1: Welches sind die wesentlichen Aufgaben eines Gerätetreibers?

Frage 2: Welche Informationen werden dem Treiber beim Aufruf einer Bibliotheksfunktion über-geben? Was ist das Ergebnis des Aufrufs der Bibliotheksfunktion?

Frage 3: Was versteht man unter einem wiedereintrittsfähigen Programm (Treiber)? Nennen Sie einige Beispiele.

Frage 4: Was versteht man unter einer einheitlichen bzw. geräteunabhängigen Software für Eingabegeräte und Ausgabegeräte? Welche Kompromisse geht man hierbei ein?

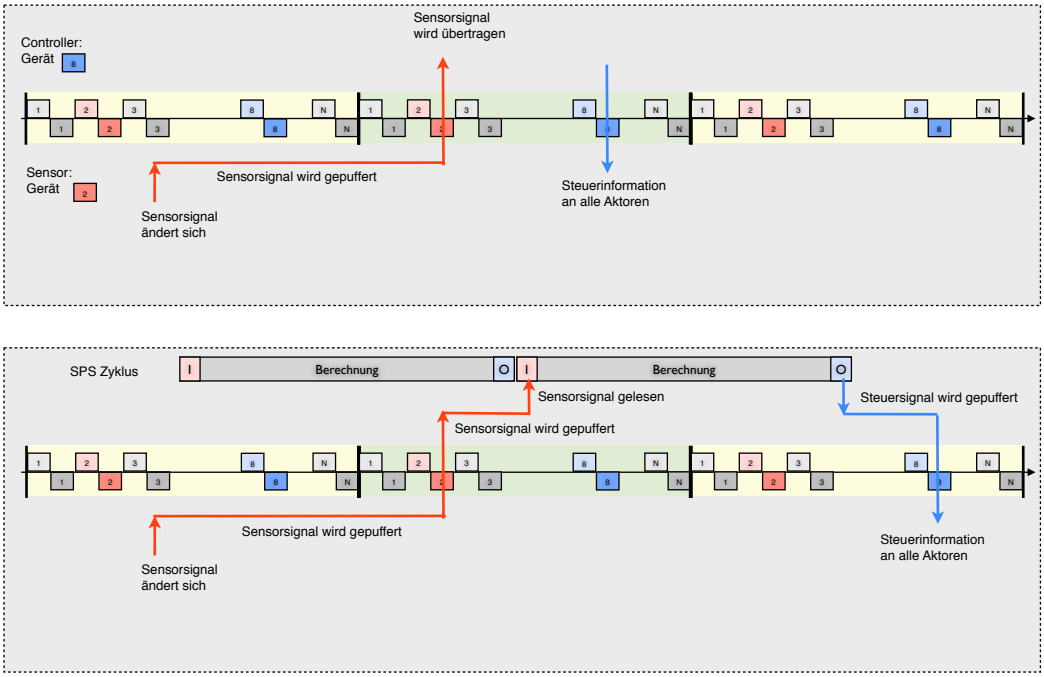
43. Speicherprogrammierbare Steuerung (SPS)

Eine Task innerhalb einer SPS arbeitet zyklisch nach folgendem Schema (1) Eingänge abfragen, (2) Ausgänge berechnen, (3) Ausgänge schalten. Das Abfragen der Eingänge geschieht

durch Empfang von Nachrichten von den Sensoren über einen Feldbus. Das Schalten der Ausgänge erfolgt durch Senden von Nachrichten an die Aktoren über den Feldbus.

Der Busmaster organisiert den Nachrichtenaustausch am Feldbus so, dass innerhalb eines Buszyklus Zeitpunkte für den Austausch von Meldungen zwischen den angeschlossenen Geräten vereinbart sind. Hierbei arbeitet der Feldbus also ebenfalls zyklisch, läuft jedoch nicht synchron mit dem Zyklus der SPS.

Die nachfolgende Abbildung zeigt den zeitlichen Ablauf.



Frage 1: Welchen Einfluss hat der Buszyklus auf die Weitergabe der Meldungen an die angeschlossenen Sender und Empfänger im Feld? Welche Verzögerung (gemessen in Bus-Zyklen) ergibt sich zwischen Senden einer Sensormeldung und Empfangen der Steuermeldung am Aktor aus Perspektive des Busses?

Frage 2: Erläutern Sie den prinzipiellen Ablauf für den Empfang und das Senden von Meldungen aus dem Feld aus der Perspektive der SPS.

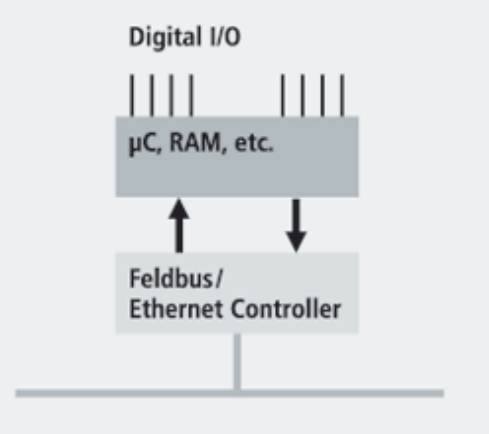
Frage 3: Welche Reaktionszeiten (gemessen in Bus-Zyklen) ergeben sich im günstigsten und ungünstigsten Fall aus Perspektive des Systems (Sensor, SPS, Aktor)?

Frage 4: Mit welchen Modifikationen am Feldbus liessen sich die Reaktionszeiten verkürzen?

44. Feldbus

Teil 1

Der Anschluss an einen Ethernet basierten Feldbus geschieht über einen Schnittstellenbaustein (NIC, Network Interface Controller), der Meldungen per DMA (Direct Memory Access) in den Arbeitsspeicher eines Mikrocomputers übertragen kann, wie in folgender Abbildung gezeigt. Der Microcomputer verfügt ausserdem über eigene digitale Eingänge und Ausgänge.

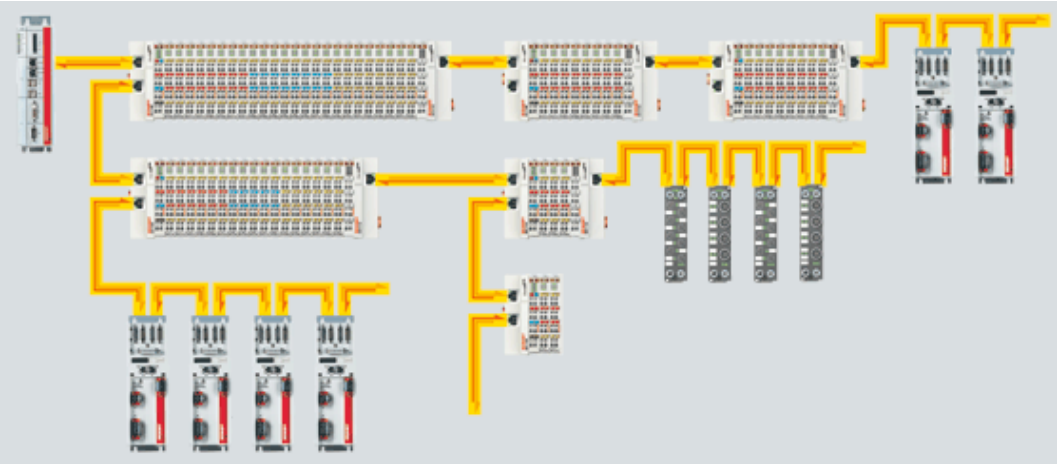


Frage 1: Erläutern Sie den Vorteil von DMA gegenüber einer einfachen Abbildung der Geräteregister in den Adressraum des Arbeitsspeichers (Memory Mapped IO)

Frage 2: Feldbusklemme: Es sei angenommen, dass die digitalen Eingänge und Ausgänge (Digital I/O) die einzigen Ports des Mikrocomputers sind. Die einzige Funktion des Mikrocomputers besteht darin, die vom Feldbus empfangenen Meldungen in Signale an den Ausgangsports abzubilden, sowie Signale an den Eingangsports als Meldungen über den Feldbus zu senden. Wie könnte man das Gerät vereinfachen (Alternativen zum Mikrocomputer)? Welche Vereinbarung ist hierfür erforderlich?

Teil 2

In der folgenden Abbildung sind Feldbusklemmen zu einem Netz verschaltet.



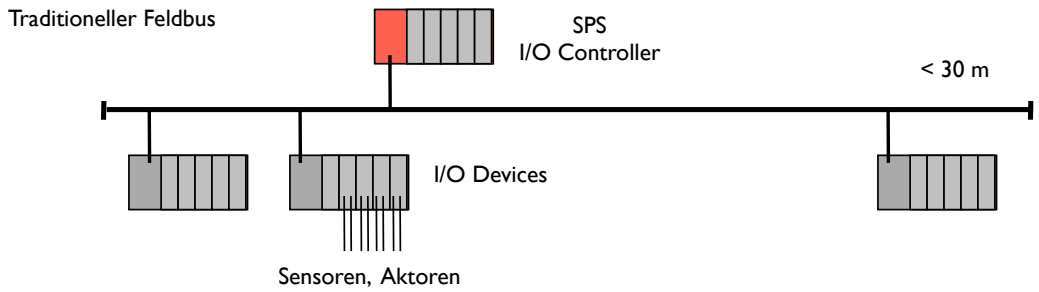
Frage 3: Beschreiben Sie die dargestellten Topologien. Hinweis: Wenn mehrere Ports pro Gerät vorhanden sind, ist im Gerät ein Ethernet-Switch enthalten.

Frage 4: Es werden Ethernet Rahmen der Größe 64 Bytes verwendet, die 20 Bytes Header und 44 Bytes Nutzinformation enthalten. Wie viele digitale I/Os lassen sich mit einem solchen Rahmen schalten? Wie lässt sich die Zahl der digitalen I/Os erhöhen?

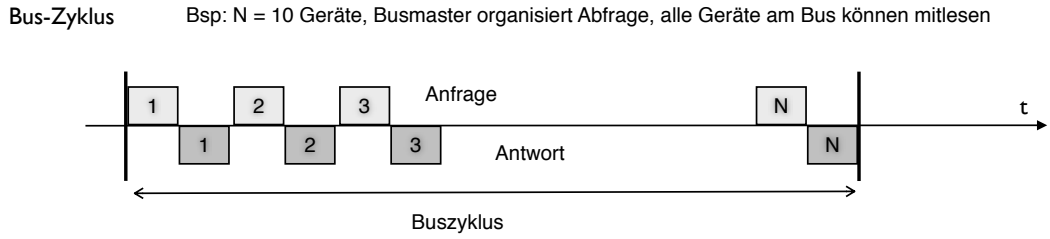
Frage 5: Als Übertragungsmedium wird Fast Ethernet mit 100 Mbit/s verwendet. Wie viel Zeit vergeht von der Sendung des Ethernet Rahmens aus Frage 4 vom Kopf des Netzes (oben links) bis zum Empfang der Meldung (Schalten der Ausgänge) über den längsten Pfad des dargestellten Netzes? Wenn alle Geräte in der längsten Kette individuell adressiert werden, welche Zeit ergibt sich für den Buszyklus?

Frage 6: Statt die dargestellten Geräte individuell mit Ethernet Rahmen zu adressieren, könnte man die Informationen für alle Geräte auf dem Pfad in den gleichen Ethernet Rahmen packen. Welcher Zeitvorteil liesse sich hierdurch erreichen? Welche Besonderheit weist ein Ethernet-Switch für eine solche Anwendung auf?

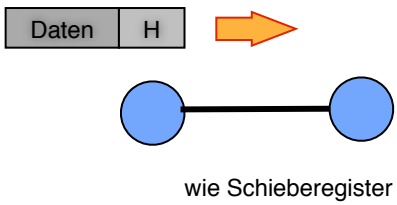
Einige Erläuterungen im Vergleich zu traditionellen Feldbussystemen:



- Laufzeit der Signale: 300 * 106 m/s ⇒ 300 m/µs. 30 m werden also in 100 ns durchlaufen.
- Übertragungsrate am Bus: 1 Mbit/s. 1 Bit dauert 1 µs, damit quaistationäre Verhältnisse auf der Leitung.
- Größe der Telegramme bzw. Nachrichten: z.B. 12 Bytes = ca 100 Bits
- Dauer eines Telegramms somit ca. 100 µs.



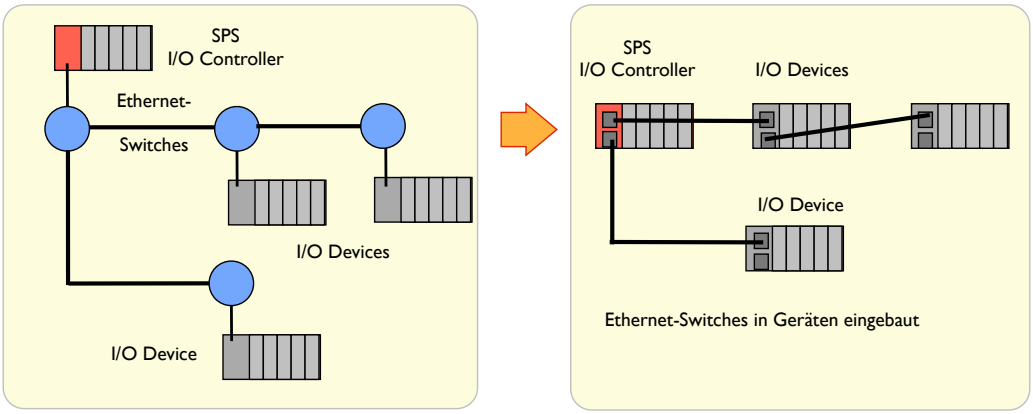
- Dauer: > 2 * N * Dauer einer Nachricht (für 10 Geräte und 100 µs: 2 ms)



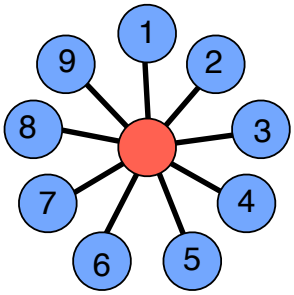
Ethernet basierter Feldbus:

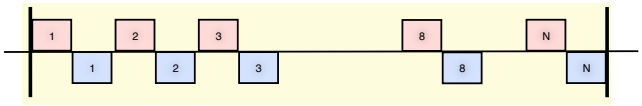
- Ethernet Rahmen: 64 Bytes (20 Bytes Header, 44 Bytes Nutzdaten)
- Übertragungsrate: 100 Mbit/s (Fast Ethernet, 1 Bit dauert 10 ns))
- 64 * 8 Bytes / 100 MBit/s ⇒ ca 5 µs Übertragungsdauer

- Ethernet-Switch: speichern und weiterleiten verursacht ca 5 µs Verzögerung pro Switch (auch bei längeren Ethernet Rahmen, da die Header-Information zum Auswerten der Zieladresse zum Weiterleiten genügt)



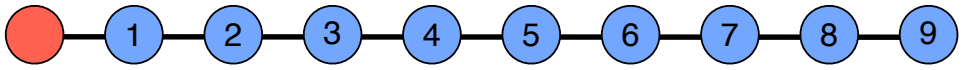
Netz-Topologien:





Traditioneller Buszyklus:

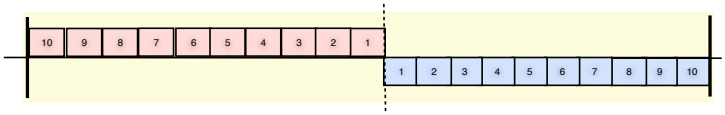
Dauer: > 2 * N * Dauer einer Nachricht (für 10 Geräte und 5 µs: **100 µs**)



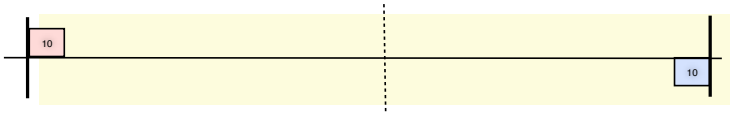
Bus-Zyklen:

A. Traditionell: maximale Laufzeit einkalkulieren: 10 * 5 µs = 50 µs Dauer einer Nachricht ⇒ **1 ms** (immer noch schneller)

B. Laufzeiten optimieren: 2 * 10 * 5 µs = 100 µs (wie in der Stern-Topologie)

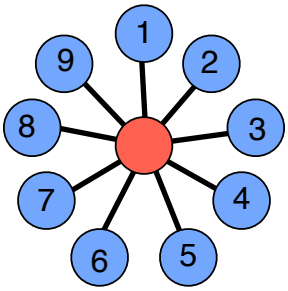


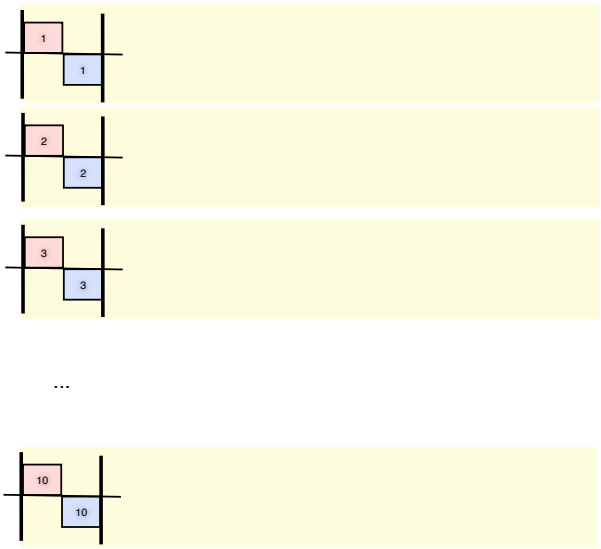
C. Sammeltelegramm: 2 * 10 * 5 µs = **100 µs** (wie in der Stern-Topologie)



D. Sammeltelegramm ohne MAC-Adressierung: 2 * 5 µs = **10 µs** (besser als Stern-Topologie?)

Stern-Topologie im Parallelbetrieb



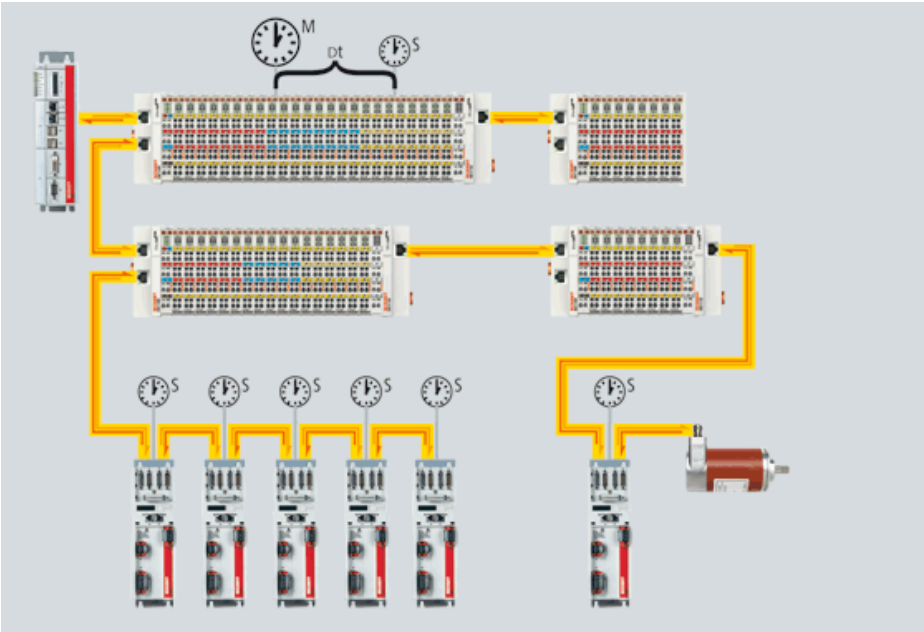


Optimierter Buszyklus:

Dauer: > 2 * N * Dauer einer Nachricht (für 10 Geräte und 5 µs: **10 µs**)

Teil 3

Zeitsynchrone Steuerung: Um Antriebe zu synchronisieren, werden verteilte Uhren eingesetzt, wie in der folgenden Abbildung gezeigt.



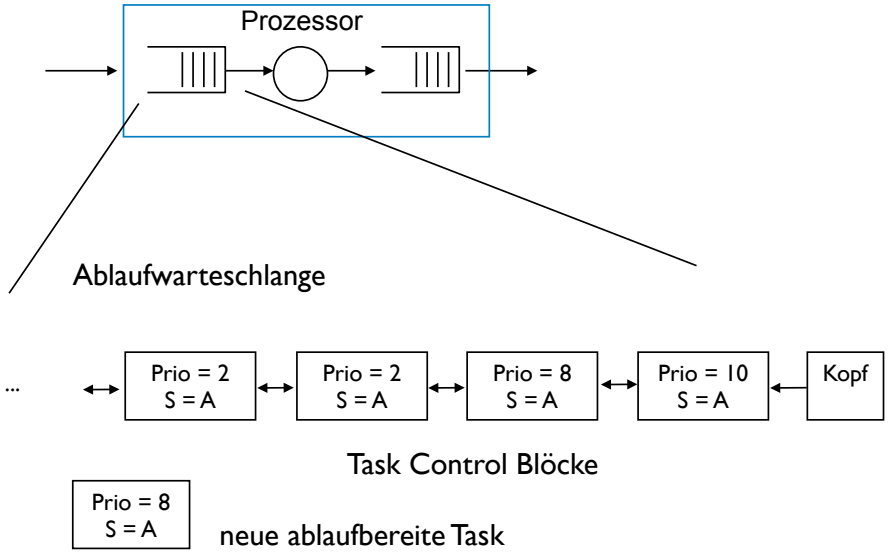
Frage 7: Welche Reaktionszeit ist für die winkelgenaue Steuerung (mit 1 Grad Genauigkeit) eines Antriebs mindestens erforderlich, der mit 3000 Umdrehungen pro Minute läuft? Welchen Vorteil bringen synchrone Uhren in den Controllern (Antriebe, Feldbusklemmen, bzw. Prozessrechner)?

Frage 8: Uhrenvergleich: Damit die Uhren synchron bleiben, müssen sie durch ein geeignetes Protokoll von Zeit zu Zeit nachgestellt werden. Hierzu übernimmt ein Gerät die Zeitbasis

(Master-Clock), alle anderen Geräte werden nach dieser Uhr gestellt (Slave Clocks). Das Stellen der Uhren erfolgt durch Versand von Nachrichten nach einem geeigneten Protokoll. Hierbei ist der Gangunterschied der Uhren festzustellen und ausserdem die Laufzeit der Nachricht zwischen den Geräten zu berücksichtigen. Beschreiben Sie ein Verfahren, mit dem sich die Uhr eines Gerätes nach der Uhr in einem anderen Gerät stellen lässt. Hinweis: Gehen Sie schrittweise vor: (1) ohne Berücksichtigung der Laufzeit, (2) mit Berücksichtigung der Laufzeit.

45. Ablaufwarteschlange

Die folgende Abbildung zeigt eine Ablaufwarteschlange zu dem Zeitpunkt, als durch Eintreffen eines Ereignisse eine bisher wartende Task ablaufbereit geworden ist.



Frage 1: Wie reagiert der Scheduler auf die neue ablaufbereite Task? In welcher zeitlichen Reihenfolge werden die Tasks in einem Multi-Tasking System mit einem Prozessorkern (CPU) abgearbeitet?

Frage 2: In welcher zeitlichen Reihenfolge werden die Tasks in einem Multi-Tasking System mit zwei Prozessorkernen abgearbeitet? Beschreiben Sie die Aktivität des Dispatchers in einem Diagramm.

46. Verfahrene Situationen

In einem Multi-Tasking System kann es zu verfahrenen Situationen kommen.

Frage 1: Erläutern Sie die Ursache der verfahrenen Situation in der Abbildung.

Frage 2: Wodurch entstehen solche Situationen in einem Multi-Tasking System? Wie lassen sich solche Situationen vermeiden? Wie lassen sie sich auflösen?

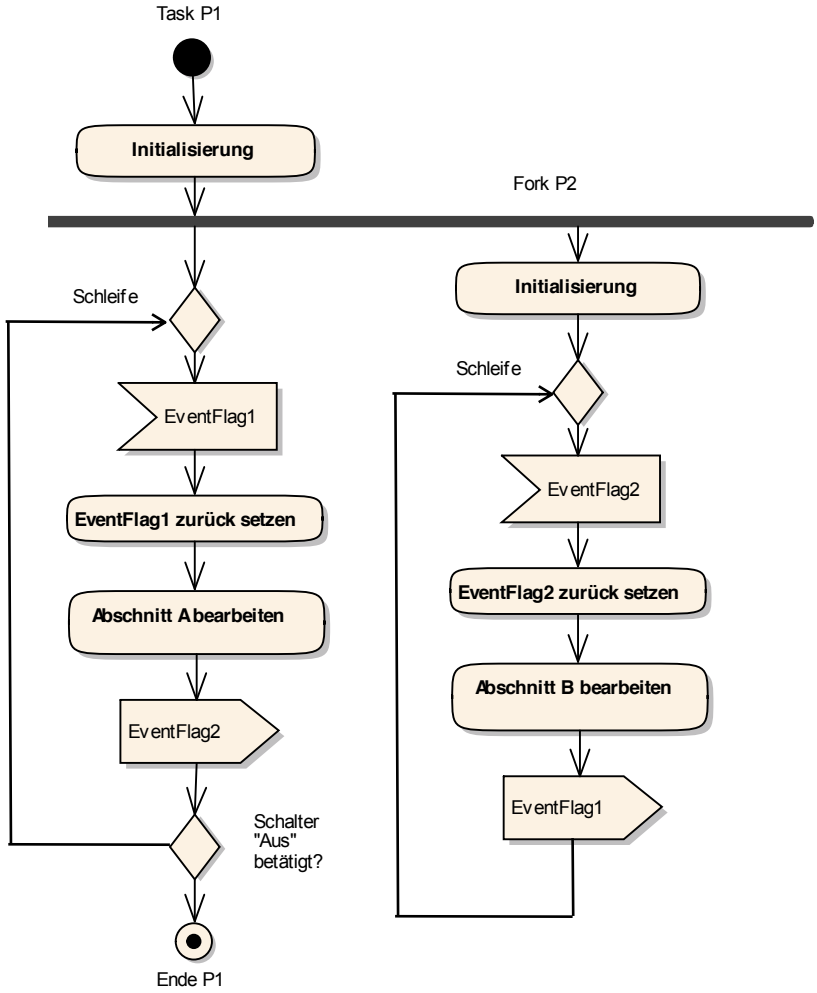
47. Task-Synchronisation

Teil 1

Folgendes Diagramm zeigt den Ablauf zweier Tasks.

Frage 1: Erläutern Sie den Ablauf. Was genau wird durch die Verwendung der Ereignis-meldungen (Event Flags) erreicht?

Frage 2: Eine Abfüllanlage soll mit Hilfe dreier paralleler Tasks realisiert werden: (1) Flasche abfüllen, (2) Kronkorken aufsetzen, (3) Etikett aufkleben. Die Aufgaben sollen in auf einander folgenden Arbeitsschritten erledigt werden. Erstellen Sie ein Aktivitäts-diagramm.



Teil 2

Drei Antriebsachsen sollen synchronisiert werden. Jede Antriebsachse soll mit Hilfe einer eigenen Task gesteuert werden. Bezugspunkt für jede Achse ist eine vorgegebene Position, die von einem Drehgeber gemeldet wird. Durchläuft die Achse diesen Bezugspunkt, wird ein Alarm erzeugt.

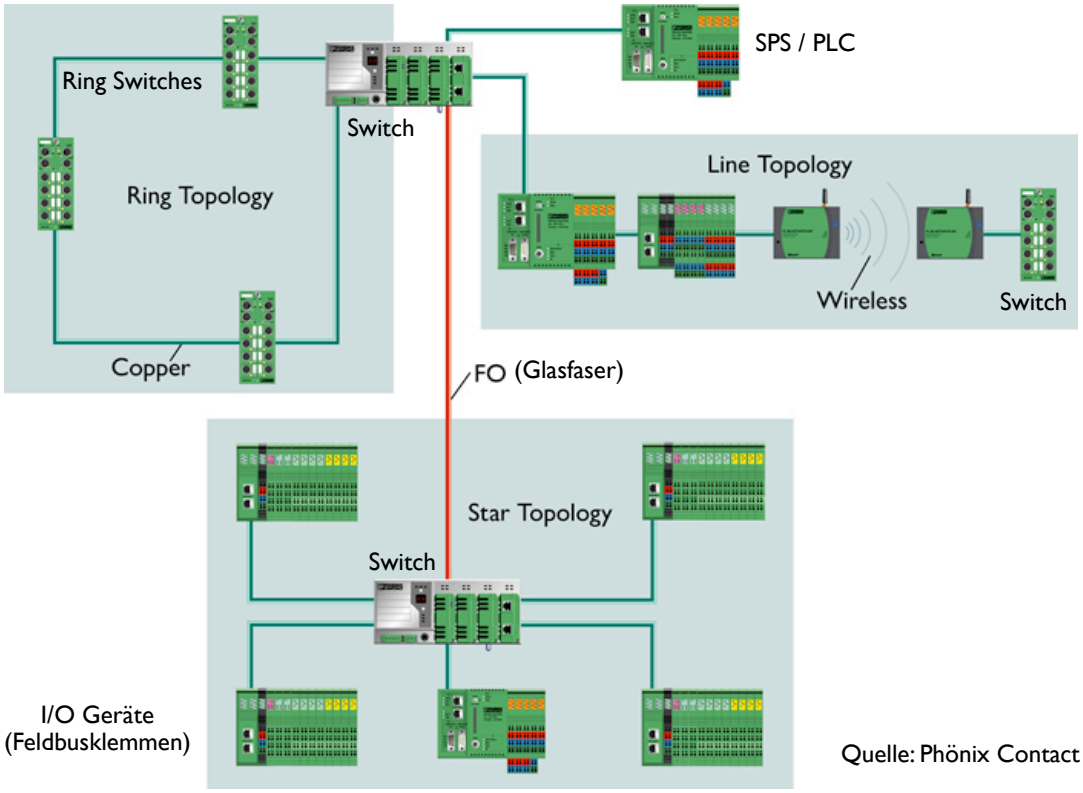
Frage 3: Mit welcher Methode kann der Gleichlauf der Achsen gewährleistet werden? Beschreiben Sie ein Konzept in Worten.

Frage 4: Erstellen Sie ein Aktivitätsdiagramm.

48. Ein weiterer Feldbus

Teil 1

Folgende Abbildung zeigt verschiedene Möglichkeiten zum Vernetzung von Feldbussen. Als Prozessrechner arbeitet eine Speicherprogrammierbare Steuerung (SPS) oben rechts in der Abbildung. Aktoren und Sensoren werden an I/O Geräte (Feldbusklemmen) angeschlossen, die an die Ethernet Switches angeschlossen sind. Geräte mit mehreren Ethernet Anschlüssen haben eingebaute Switches und können daher ebenfalls Meldungen weiter geben.



Quelle: Phönix Contact

Das Netz wird als Fast Ethernet mit 100 Mbit/s betrieben. Zum Austausch der Prozessdaten werden kurze Ethernet Rahmen von 64 Bytes verwendet, von denen 44 Bytes für Nutzdaten zur Verfügung stehen. Die SPS organisiert als I/O Controller bzw. Busmaster den Buszyklus. In jedem Switch werden die Meldungen gespeichert und weitergeleitet. In den I/O Geräten werden die Meldungen empfangen und Meldungen mit dem aktuellen Status der Eingänge als Antwort gesendet.

Frage 1.1 (4 Punkte): Es werden 10 I/O Geräte in Stern-Topologie an einen Switch angeschlos-sen. Der I/O Controller organisiert den Buszyklus so, dass alle Geräte der Reihe nach abgefragt werden und antworten. Wie lange dauert der Buszyklus mindestens?

Frage 1.2 (4 Punkte): Es werden 10 I/O Geräte in Linien-Topologie bzw. in Ring-Topologie miteinander verbunden. Der I/O Controller organisiert den Buszyklus so, dass alle Geräte der Reihe nach abgefragt werden und antworten. Wie lange dauert der Buszyklus mindestens?

Frage 1.3 (6 Punkte): Skizzieren Sie auf der Zeitachse einen möglichen Buszyklus zu Frage 1.1 und Frage 1.2 aus Sicht des I/O Controllers (Geräte abfragen und Rückmeldungen).

Frage 1.4 (4 Punkte): Für den zu steuernden Prozess genügt eine Zykluszeit von 10 ms. Es werden maximal 10 I/O Geräte angeschlossen. Die Anordnung soll jedoch Einzelfehler im Netz verkraften. Welche Topologie wählen Sie? Begründen Sie Ihre Entscheidung durch Vergleich mit den anderen Topologien.

Teil 2

Damit das Netzwerk neben den zeitkritischen Prozessdaten auch regulären Verkehr übertragen kann, wird ein Zeitmultiplex eingeführt, d.h. alle Controller, Switches und Geräte werden im gleichen Takt zwischen einem Zeitschlitz für Prozessdaten und einem Zeitschlitz für regulären Verkehr umgeschaltet. Der Takt für den Zeitmultiplex kann von 250 µs bis 1 ms eingestellt werden. Es wird abwechselnd ein Zeitintervall für Prozessdaten verwendet, das folgende Zeitintervall für regulären Verkehr.

Frage 1.5 (4 Punkte): Zur Steuerung des Prozesses ist eine maximalen Reaktionszeit von 1 ms erforderlich. Hierbei beschreibt die Reaktionszeit aus Sicht des Prozesses die Zeit zwischen dem Senden einer Meldung mit Statusinformation und dem Empfang einer Meldung mit Steuerinformation. Wählen Sie eine Linienkonfiguration für 10 Geräte. Kann die gewünschte Reaktionszeit erreicht werden? Begründen Sie Ihre Antwort.

Frage 1.6 (4 Punkte): Wählen Sie eine Einstellung für den Zeitmultiplex. Begründen Sie Ihre Entscheidung. Skizzieren Sie den Buszyklus auf der Zeitachse aus Sicht des I/O Controllers (Geräte abfragen und Rückmeldungen empfangen).

49. Echtzeit-Strategie

Folgendes Diagramm (nächste Seite) zur Aufgabe zeigt die Vorbereitung eines zeitkritischen Manövers zur Ausführung auf dem Bordcomputer einer Raumstation in naher Zukunft. Bei Durchsicht der Planung reagieren die Führer der Station wie folgt:

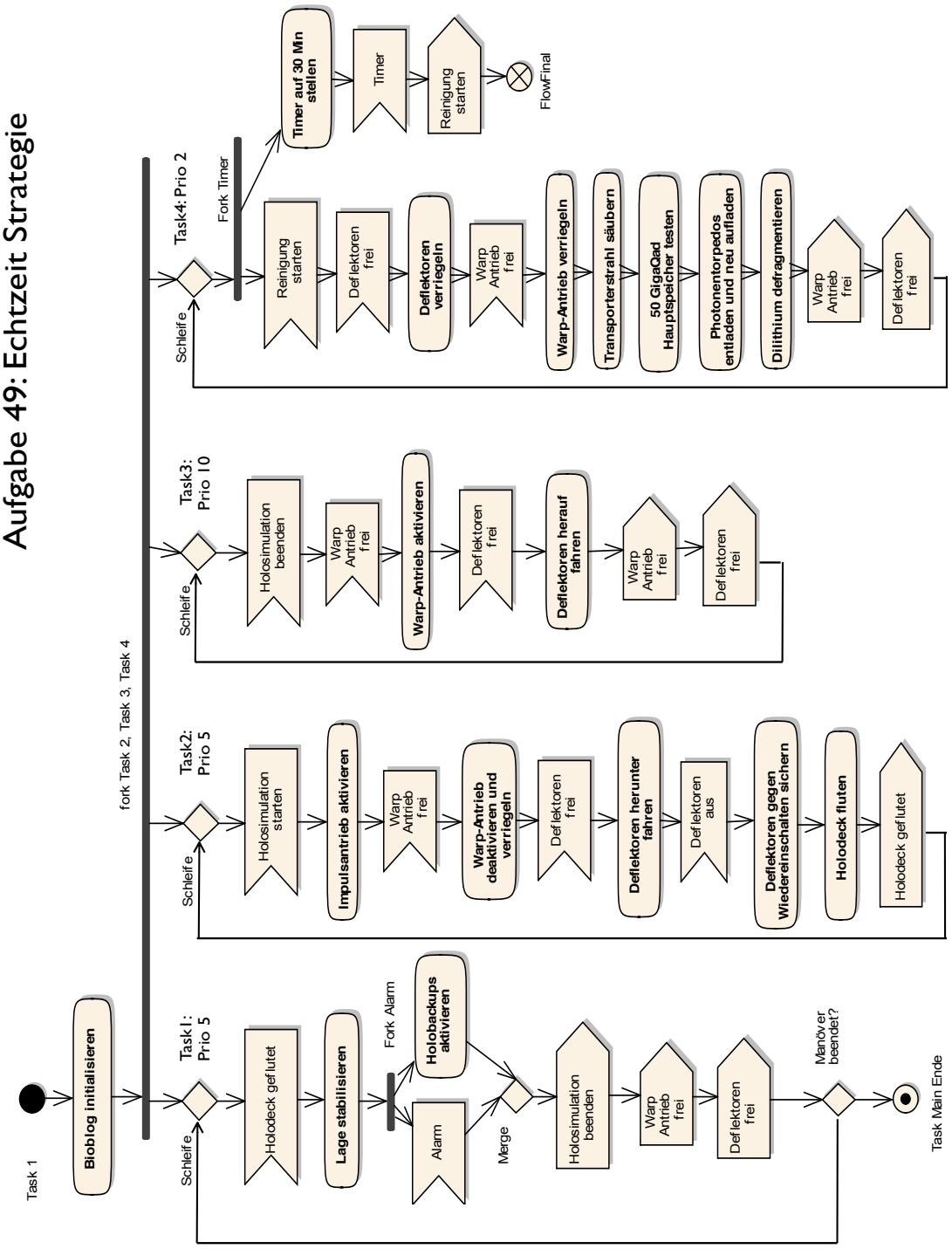
Frage 3.1 (6 Punkte): Warum? Erläutern Sie die Mängel in der Planung. Hinweis: Der Bord-computer der Station verfügt über einen leistungsfähigen Mehrkern-Prozessor, ordnet jeder Task einen eigenen Prozessorkern zu und verarbeitet alle Tasks wirklich parallel. Einstiegspunkt in Task 2 ist eine Benutzereingabe zum Start des Manövers („Holosimulation starten“).

Frage 3.2 (8 Punkte): Das Manöver soll auch auf Stationen laufen, die nur über Bordcomputer mit einem einzelnen Prozessor verfügen, der im Multi-Tasking betrieben wird. Welche zusätzlichen Probleme und Mängel ergeben sich hier?

Frage 3.3 (6 Punkte): Wie lässt sich der Plan modifizieren, damit er auf einem System mit echtem Parallelbetrieb der Tasks funktioniert? Hinweis: Einige Worte zur Beschreibung und Begründung der Massnahmen genügen, es ist kein Diagramm erforderlich.

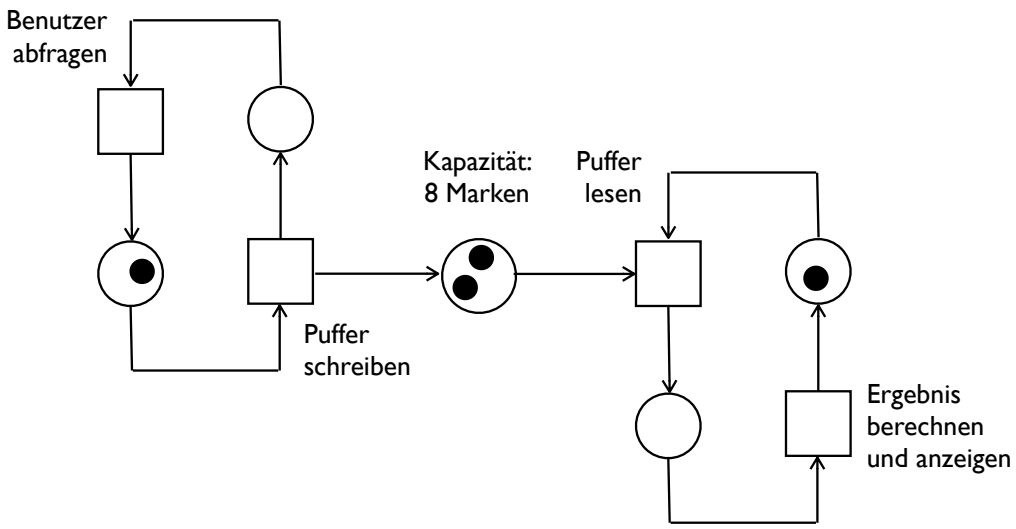
Frage 3.4 (8 Punkte): Wie lässt sich der Plan modifizieren, damit er auch auf einem Multi-Tasking System mit einem Prozessorkern funktioniert? Hinweis: Einige Worte zur Beschreibung und Begründung der Massnahmen genügen, es ist kein Diagramm erforderlich.

Aufgabe 49: Echtzeit Strategie



50. Task-Synchronisation

Zwei Tasks sind über einen Ringpuffer miteinander verbunden, wie in der folgenden Abbildung dargestellt.



Frage 4.1 (4 Punkte): Die Startbedingen werden folgendermassen festgelegt: Task 1 (links im Diagramm) erhält eine Marke, die Koppelstelle (Ringpuffer) erhält keine Marke, Task 2 (rechts im Diagram) erhält keine Marke. Welcher Ablauf ergibt sich?

Frage 4.2 (8 Punkte): Erläutern Sie die Funktionen und die Kooperation beider Tasks mit den in der Abbildung gezeigten Startbedingungen. An welchen Stellen warten die Tasks? Wodurch erfolgt die Synchronisation?

Frage 4.3 (8 Punkte): Erstellen Sie ein Aktivitätsdiagramm für die produzierende Task und für die konsumierende Task. Hinweis: Verwenden Sie Ereignismeldungen (Event Flags) und Zähler. Legen Sie die Startbedingungen fest (Initialisierung).

Frage 4.4 (8 Punkte): Erweitern Sie den in der Abbildung gezeigten Ablauf um eine weitere Task 3, die ebenfalls aus dem Ringpuffer liest. Hinweis: Darstellung als Petri-Netz genügt. Beschreiben Sie die Funktionsweise der Anordnung. Beschreiben Sie, wie sich die beiden konsumierenden Tasks mit Ereignismeldungen und Zählern synchronisieren lassen.

Englisch - Deutsch

Context Switch	Kontextumschaltung
Deadlock	Verklemmung
Dispatching	Einlastung
Interrupt	Alarm
Instruction Pointer	Befehlszeiger
Fork	Verzweigung (eines Kontrollflusses)
Heap	Halde
Join	Zusammenführung (eines Kontrollflusses)
Livelock	aktive Behinderung eines Prozesses
Mailbox	Postfach
Resources	Betriebsmittel
Scheduling	Einplanung
Stack	Stapel
Stack Pointer	Stapelzeiger
Task	Rechenprozess

Abkürzungen

BSR	Bit- und Statusregister
DR	Datenregister
UML	Unified Modelling Language

Literatur

(1) Eberhard Kienzle, Jörg Friedrich, Programmierung von Echtzeitsystemen, Carl Hanser Verlag, 2008, ISBN-13: 978-3-446-40735-0

(2) Karl Friedrich Gebhardt, Realzeit-Systeme, Vorlesungsmanuskript, Angewandte Informatik, DHBW Stuttgart

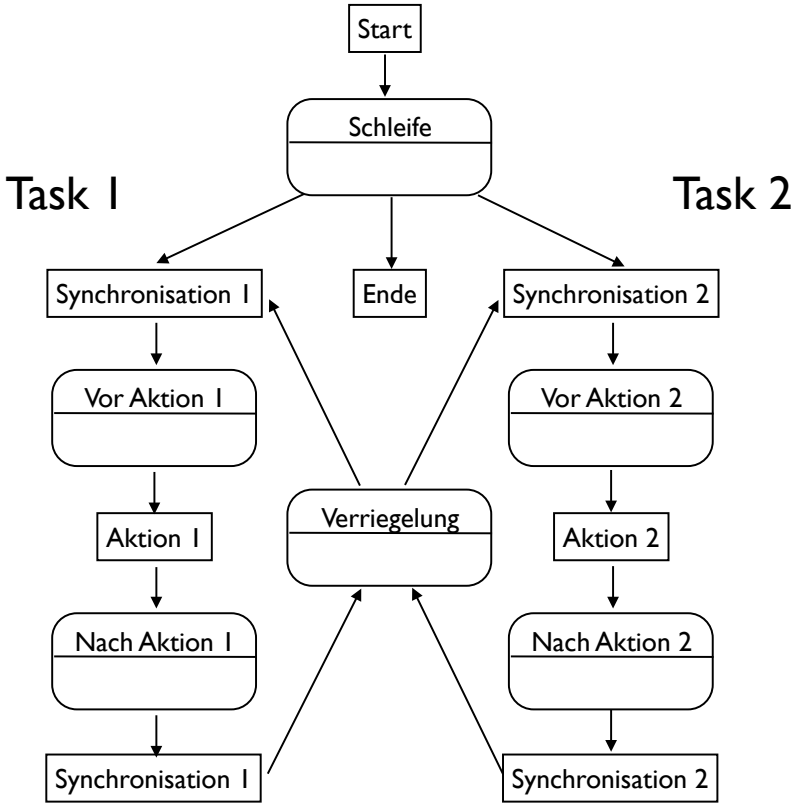
(3) Heinz Wörn, Uwe Brinkschulte, Echtzeitsysteme: Grundlagen, Funktionsweisen, Anwendungen, Springer, 2005, ISBN-13: 978-3540205883

(4) Andrew S. Tanenbaum, Modern Operating Systems, Prentice Hall International, 2007, ISBN-13: 978-0138134594

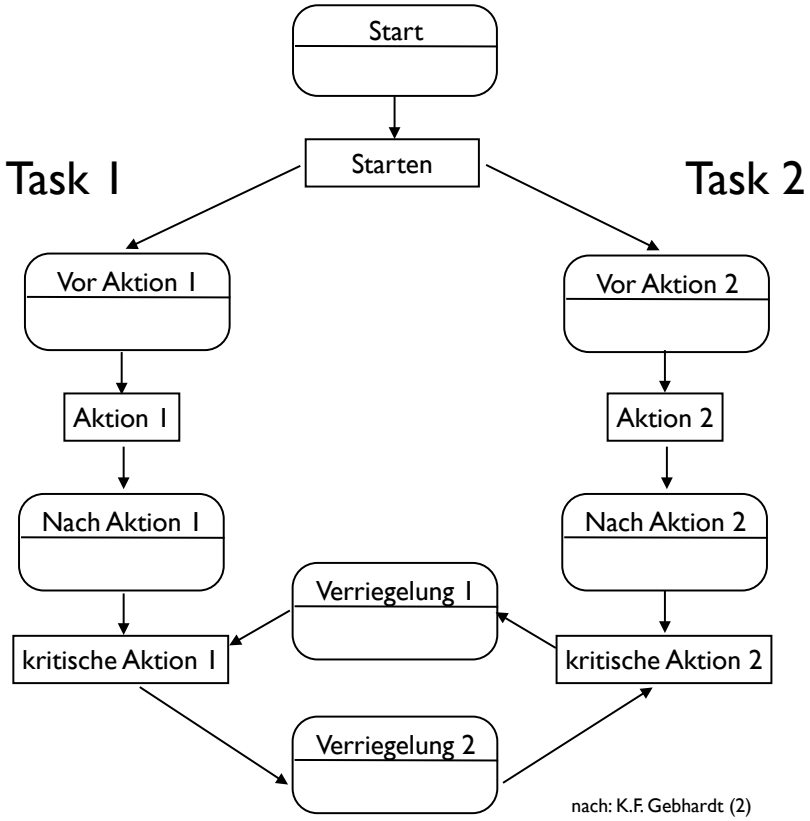
(5) Chris Rupp, Stefan Queins et al, UML 2 glasklar, Praxiswissen für die UML Modellierung, Carl Hanser Verlag GmbH & CO. KG; 4. aktualisierte und erweiterte Auflage, 2012, ISBN-13: 978-3446430570

(6) Peter Hruschka, Chris Rupp, Agile Softwareentwicklung für Embedded Real-Time Systems mit der UML, Carl Hanser Verlag GmbH & CO. KG, 2002, SBN-13: 978-3446219977, siehe auch SOPHIST GmbH, Nürnberg

Spielfelder für das Markenspiel



nach: K.F. Gebhardt (2)



nach: K.F. Gebhardt (2)