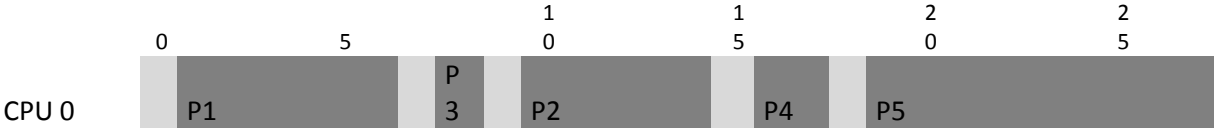
Aufgabe 1: Scheduling-Algorithmen (11 Punkte)

Gegeben seien die fünf nicht-periodischen Prozesse ohne I/O-Operationen aus Tabelle 1, die auf einem Einprozessor-System ausgeführt werden sollen. Nehmen Sie für diese Aufgabe an, dass der Prozesswechsel eine Zeiteinheit benötigt und der Prozessor für diese Zeit keine andere Operation ausführen kann. Diese Prozesswechselzeit wird auch benötigt, um einen Prozess zu initialisieren.

Prozess	P_1	P_2	P_3	P_4	P_5
Ankunftszeit (t_i)	0	5	6	7	9
Bedienzeitforderung (b_i)	6	5	1	2	8

Tabelle 1: Ankunftszeiten und Bedienzeitforderungen der Prozesse P_1 bis P_5

a) Illustrieren Sie die Ausführungsreihenfolge der fünf Prozesse für die nicht-verdrängende Scheduling-Strategie *beste Bediengüte*. Die Bediengüte *BG* ist hierbei definiert als $BG(Pi,t) = \frac{Wartezeit+Bedienzeit}{Bedienzeit}$. Bei gleicher Bediengüte wird der Prozess mit der kleinsten Bedienzeitanforderung ausgewählt.



Berechnungen:

P2	BG(P2, 7)=(2+5)/5=7/5	BG(P2, 9)=(4+5)/5=9/5	
P3	BG(P3, 7)=(1+1)/1=2		
P4	BP(P4, 7)=(0+2)/2=1	BG(P4, 9)=(2+2)/2=2	BG(P4, 15)=(8+2)/2=5
P5		BG(P5, 9)=(0+8)/8=1	BG(P5, 15)=(6+8)/8=7/4

b) Illustrieren Sie die Ausführungsreihenfolge der fünf Prozesse für die Scheduling-Strategie Round Robin mit Zeitscheibengröße Δt = 2 (zzgl. Prozesswechseldauer, falls nötig). Bei nicht vollständig ausgefüllten Zeitscheiben soll der Prozesswechsel vorgezogen werden, wobei die Prozesswechseldauer zu berücksichtigen ist und anschließend eine neue Zeitscheibe beginnt. Neue Prozesse sollen an das Ende des Warteraums eingestellt werden.

CPU 0	P1	P1	P1	P2	P3	P4	P5	P2	P5	P2	P5	P5

Wartende Prozesse	x	x	P2	P3	P4 P5 P2	P5 P2	P2	P5	P2	P5

c) (optional) Illustrieren Sie die Ausführungsreihenfolge der fünf Prozesse wie in Aufgabe 1b) f r die Scheduling-Strategie Round Robin, jedoch nun mit der Zeitscheibengröße $\Delta t = 4$.

d) (optional) Die Antwortzeit eines Prozesses sei als die Zeit zwischen seiner Ankunft und seiner Termination definiert. Berechnen Sie für jeden der drei resultierenden Schedules aus den Aufgaben 1a), 1b) und 1c) die durchschnittliche Antwortzeit der beteiligten Prozesse.

e) (optional) Die Größe (Dauer) einer Zeitscheibe beim Scheduling-Algorithmus Round Robin wird über einen Parameter Δt festgelegt. Nennen Sie jeweils einen Vor- und Nachteil für kurze bzw. lange Zeitscheiben.

Kurze Zeitscheiben ermöglichen ein schnelles Umschalten zwischen alle bereitstehenden Prozessen und somit eine schnellere Reaktion auf Ereignisse. Jedoch steht den einzelnen Prozessen pro „Aktivierung“ weniger Zeit zur Verfügung, somit kann unter Umständen nicht vollständig auf Aktion reagiert werden. Ein weiterer Nachteil besteht darin, dass viele Prozesswechsel durchgeführt werden, die Zeit kosten. Ein relativ kurzer Prozess kann dadurch lange brauchen, dass bis er wieder an der Reihe ist.

Aufgabe 2: Echtzeit & Mehrprozessor-Scheduling (20 Punkte)

a) Gegeben seien die drei Aufträge A_1 bis A_3 (ohne I/O-Operationen) aus der folgenden Tabelle, die auf einem Einprozessorsystem periodisch auszuführen sind und dabei stets zum Ende jeder Periode vollständig abgearbeitet sein müssen. Leiten Sie mathematisch her, warum es selbst mit einem idealen Scheduler nicht möglich ist, die Deadlines aller Aufträge einzuhalten.

Auftrag	A_1	A_2	A_3
Periodendauer (p_i)	4	7	3
Bedienzeitforderung (b_i)	1	3	1

Benutzt man die Formel zur Bestimmung der Prozessorauslastung (Folien B02 Seite 22), so kommt man auf eine Auslastung U von $U = \frac{1}{4} + \frac{3}{7} + \frac{1}{3} \approx 1{,}012 > 1$. Da die CPU nicht zu mehr als 100% ausgelastet sein kann, gibt es für diese Aufträge kein gültiges Schedule.

b) Gegeben seien die vier Aufträge B_1 bis B_4 (ohne I/O-Operationen) aus der folgenden Tabelle, die auf einem Einprozessorsystem periodisch auszuführen sind und dabei stets zum Ende jeder Periode vollständig abgearbeitet sein müssen. Die Perioden der Aufträge beginnen gemeinsam zum Zeitpunkt $t = 0$.

Auftrag	B_1	B_2	B_3	B_4
Periodendauer (p_i)	7	11	9	4
Bedienzeitforderung (b_i)	3	1	2	1

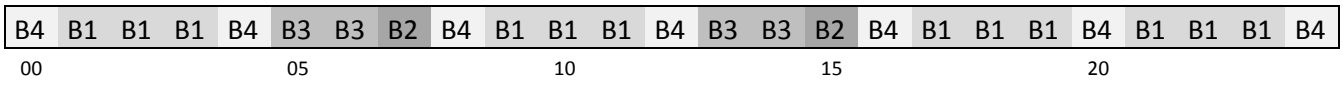
Illustrieren Sie für das Zeitintervall $t \in [0,25]$ die Ausführungsreihenfolge der Aufträge (d. h. die CPU-Belegung über die Zeit durch die vier periodisch zu bearbeitenden Prozesse) für die Strategien

i) (optional) Earliest Deadline First,

ii) Rate Monotonic Scheduling.

Der Aufwand für den Prozesswechsel soll in dieser Aufgabe vernachlässigt werden. Falls zu einem Zeitpunkt zwei oder mehr Aufträge dieselbe Priorität bzw. Deadline haben, soll der Auftrag mit der längsten Wartezeit ausgeführt werden. Falls Deadlines nicht eingehalten werden können, benennen Sie diese Fälle.

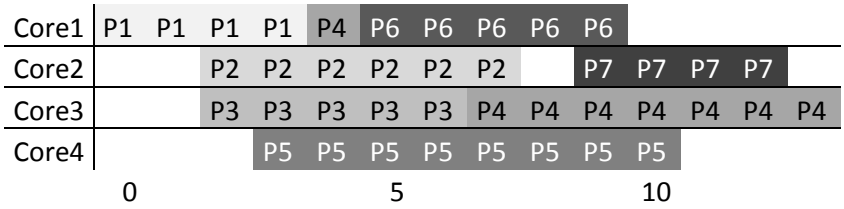
ii) Nimmt man an, dass Aufträge nicht während ihrer Ausführung unterbrochen werden können, so kommt man auf folgenden Schedule:



Können die Aufträge unterbrochen werden, so könnte B_2 nicht vor dem Ende seiner Deadline ausgeführt werden.

c) Gegeben seien die sieben Prozesse P_1 bis P_7 (ohne I/O-Operationen) aus der folgenden Tabelle, die auf einem Multiprozessorsystem mit vier CPU-Kernen auszuführen sind. Illustrieren Sie die Ausführungsreihenfolge dieser Prozesse auf dem System für die verdrängende Scheduling-Strategie Timesharing (auch Load sharing) nach Prioritäten (je höher die Zahl desto wichtiger der Prozess). Der Aufwand für den Prozesswechsel ist dabei zu vernachlässigen. Sind mehrere CPU-Kerne frei, wird der mit der niedrigsten Nummer gewählt.

Prozess	P_1	P_2	P_3	P_4	P_5	P_6	P_7
Ankunftszeit (t_i)	0	2	2	3	3	5	9
Bedienzeitforderung (b_i)	4	6	5	8	8	5	4
Priorität	2	5	4	1	3	4	8



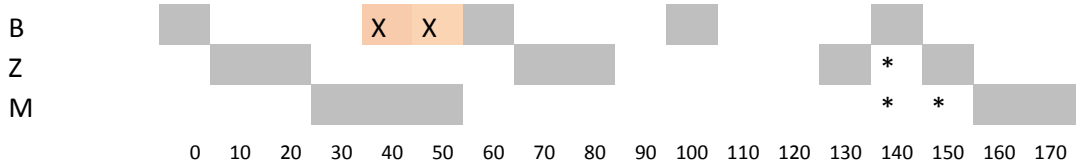
Aufgabe 3: Prioritätsinversion (9 Punkte)

Erlaubt man Prozessen den exklusiven Zugriff auf Ressourcen unter Verwendung von Synchronisationsprimitiven wie zum Beispiel Semaphoren, kann es zu unerfreulichen Effekten kommen, sofern keine entsprechenden Maßnahmen getroffen werden. Zwei dieser Effekte sind Prioritätsinversion und Deadlocks.

Beinahe hätte die Mars-Pathfinder-Mission 1996/97 ein unrühmliches Ende gefunden. Informieren Sie sich z. B. im Internet[1] über die Gründe, warum sich der Computer der Mars-Sonde ständig selbst neu starten musste.

a) Illustrieren Sie (unter Verwendung einer Darstellung in Anlehnung an Abbildung 1 mit einer geeigneten Skalierung der Zeitskala) mit Hilfe dieses Hintergrundwissens die Ausführungsreihenfolge der drei periodischen Aufträge *B* (Bus-Management), *Z* (z. B. Sensor auslesen) und *M* (Meteorologie-Auftrag) unter Verwendung einer Scheduling-Strategie nach festen Prioritäten f r das Intervall [0-170], wobei der Aufwand f r den Prozesswechsel in dieser Aufgabe vernachlässigt werden soll. Die Prioritäten seien folgendermaßen definiert: *Prio*(*B*) > *Prio*(*Z*) > *Prio*(*M*). Beachten Sie hierbei, dass der M-Auftrag und der B-Auftrag ihren Zugriff auf eine gemeinsame Ressource mittels einer Semaphore synchronisieren und die Verdrängung eines Auftrags nur geschieht, wenn der höher priorisierte Auftrag alle erforderlichen Ressourcen erhält.

Auftrag	*M*	*B*	*Z*
Periodendauer p_i	115	40	60
Bedienzeit b_i	30	10	20

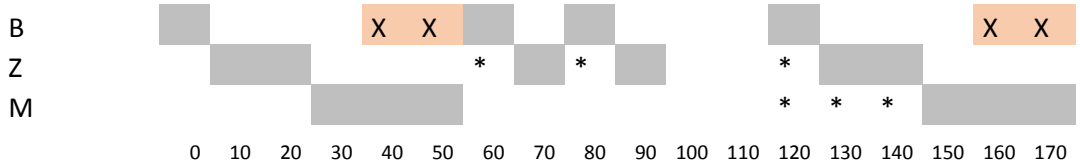


X = höher Prozess möchte, kann aber nicht (Prozessinversion)

* = niedriger Prozess möchte, darf aber nicht

Annahme: Der Startpunkt der nächsten Ausführung verschiebt sich entsprechend, falls der Prozess zu seinem eigentlichen Startzeitpunkt nicht ausgeführt werden konnte.

[1] Glenn Reeves: What really happened on Mars?
http://research.microsoft.com/en-us/um/people/mbj/Mars_Pathfinder/Authoritative_Account.html



X = höher Prozess möchte, kann aber nicht (Prozessinversion)
* = niedriger Prozess möchte, darf aber nicht

Annahme: Die Ausführung des Prozess muss in seiner Periodendauer passieren.

b) (optional) Erklären Sie anhand ihrer Illustration das Phänomen der Prioritätsinversion.

Zum Zeitpunkt 40 möchte der Prozess B mit der höchsten Priorität wieder arbeiten, jedoch kann er nicht, da der Prozess M (mit geringer Priorität) noch eine Ressource exklusiv verwendet. Somit kann Prozess B nicht starten/weitermachen. Unter Umständen (nicht in unserem Beispiel) kann ein Prozess einen noch niedrigeren Prozess verdrängen und dann muss einen höheren Prozess noch länger auf die Freigabe von der vom tieferen Prozess verwendeten Ressource warten, da dem ‚mittleren' Prozess alle Ressourcen zu Verfügung stehen. Damit wird sozusagen die Priorisierung ausgehebelt.

c) (optional) Wie hat das Software-Team der Mars-Pathfinder-Mission das Problem schließlich gelöst? Illustrieren Sie die neue Ausführungsreihenfolge unter Beachtung ihrer Lösung.

Die Idee besteht in der Verwendung der Prioritätsvererbung. Wenn ein höherer priorisierter Prozess auf einen niedriger priorisierten Prozess wartet, dann wird die Priorität des niedrigen Prozess kurzfristig auf die des höheren Prozess angehoben. Dadurch kann der niedrig priorisierte Prozess nicht mehr verdrängt werden und der Prozess rechnet so schnell zu Ende wie er eben kann, um dem höher priorisierten Prozess die Ressource zur Verfügung zu stellen.
Als Illustration gelten die gleichen wie bereits oben, da keine Prozessverdrängung festgestellt werden konnte. Der am niedrigsten priorisierte Prozess M rechnet zu Ende und direkt danach ist der Prozess B dran, falls wieder eine neue Periode für ihn angefangen hat.