Exercise 1: Resources (mandatory)

There are 3 periodic Tasks $T^k = (\Delta t_{exec}{}^k; \Delta t_{per}{}^k; \Delta t_{allocR}{}^k; \Delta t_{useR}{}^k)$; k=1 … 3 .

- $\Delta t_{exec}{}^k$ is the execution time (CPU) for each job of task k,
- $\Delta t_{per}{}^k$ is the period time of the tasks (defines the deadlines accordingly),
- $\Delta t_{allocR}{}^k$ defines the time difference between the start time of a job and the point in time when the resource is requested to be allocated by the job,
- $\Delta t_{useR}{}^k$ is the usage time (time period in which the resource is needed by the job. After this time period the resource will be released for the use by other jobs).

Remark: In contrast to the CPU C the resource R usually is non-interruptible!

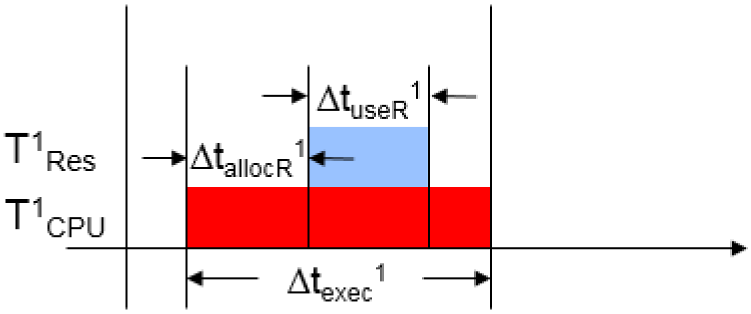


Fig. 1: Job with use of a resource R

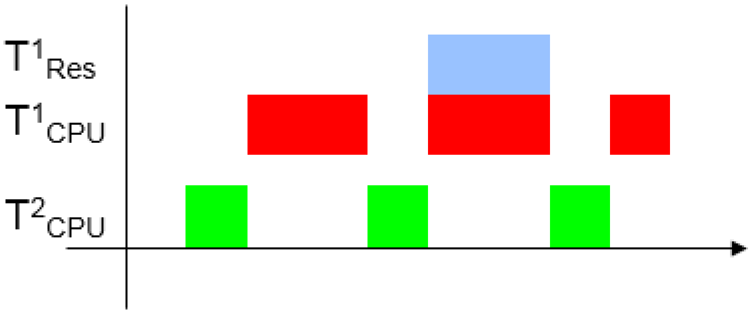


Fig. 2: RMS with use of a resource R by T^1

Please do a schedule for:

$T^1(5; 15; 2; 2)$; $T^2(2; 10; 1; 1)$; $T^3(7; 23; 1; 2)$
with RMS (including resources) without additional resource management.

Exercise 2: Resources (mandatory)

(a) Do a schedule for the tasks (the usage of the resource can't be interrupted)
T^1 (5; 14; 0; 0); T^2 (2; 10; 1; 1); T^3(8 ;23 ;2 ; 6) and
show, that there is a resource conflict.

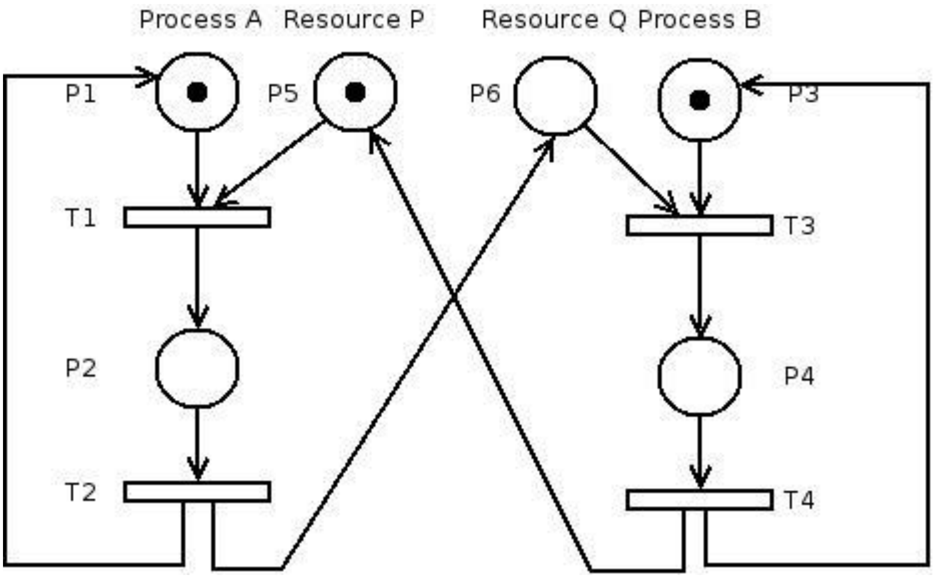
(b) Please show, that there is a solution using priority inheritance.

Exercise 3: Petri Nets (mandatory)

Petri Net:

a) Find the Paper "Wang: Petri Nets for Dynamic System Modeling" on the e-learning platform.
Read the paper thoroughly especially the sections about safeness, liveness, reachability and reachability analysis. Explain the terms safeness and liveness.

b) Is the petri net in the figure alive and/or safe? A detailed explanation is required! Use a reachability graph for testing the liveness.



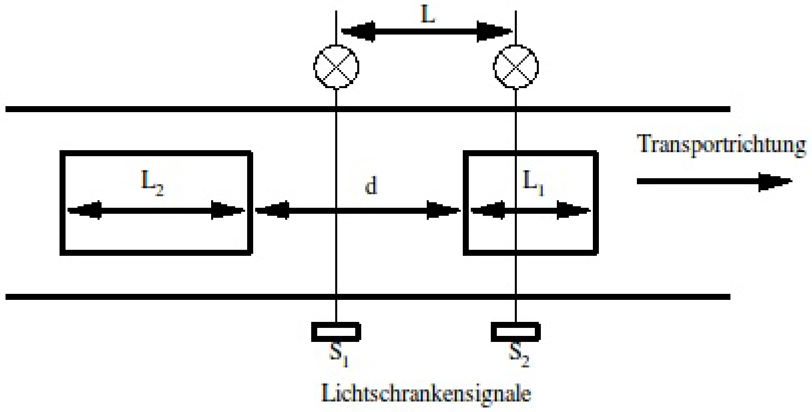
You should practice creating Petri-Nets.
You may develop as many as you would like to.
For attestation one is mandatory.
For fun and the ease of use you use PIPE v4.3.0
that you find on the Moodle platform.

However, in the examination you have to do it on paper!

Exercise 4: Petri Net: packet sorting machine (optional)

Develop a packet sorting machine, which should meet the following specification: You have a conveyor band with two photo sensors. The sensors will produce the signals S1 and S2. The packets have different sizes (L1 and L2).
Following rules exist:

- $L_1 < L < L_2$
- $L < d + L_1$
  (d: minimal distance between two arbitrary packets/
  Mindestabstand zwischen zwei beliebigen Paketen)
- An object inside the photo sensor will produce $S_i = 1$



If the sorter detects a large packet the output (O) will change to one (O = 1), until the packet has passed the first photo sensor(S1).
In all other cases the Output is zero (O = 0)
Hint: model the rising and falling edges of the photo sensors S1, S2 as transitions.

Exercise 6: Petri Net (optional): railways

Imagine a railway line with two tracks to two directions between two stations. In the middle part there is only one track available. If you can't imagine the situation, take a tour to with tramway no 17 in Frankfurt and find the situation in place between station "Varrentrappstraße" and "Nauheimerstraße" near Messe.
You need not to model the traffic lights. However, if you feel comfortable with your solution you may add the traffic lights

Exercise 7: Petri Net (optional): traffic lights

Try the examples in PIPE. Develop a Petri-Net for simple traffic lights:
a) for pedestrians,
b) for cars (replace the German state "red-yellow" by "yellow".
c) Combine a) and b).
d) When you feel comfortable you may
develop a real existing traffic crossing in Frankfurt
(avoid the big ones!)

Exercise 8: Petri Net (optional): selling machine

Try to develop the coin counter for a selling machine.
You start by creating two places for the debit value:
One is for 1.00€ for a soft drinking and the other is for 2.50€ for a beer.
You may insert coins with values of 1.00€ and 50ct.
If you fell comfortable you may add more debit values and or values for coins.

Exercise 9: Petri Net (optional): TCP/IP

Try to develop the transmission of a request/response pattern of TCP/IP
(this is part of the lecture "Networks" or "Distributed Systems").
You may use a colored Petri-Net in PIPE, but it is not necessary.

Exercise 10: Petri Net (optional): Watergate(Schleuse)

Try to develop a Petri-Net that simulats a watergate of a river.
If you don't know how this works, take a trip to "Schleuse Offenbach (Bus 46 stop Gerbermühle" or to "Schleuse Schwanheim (Bus 51 stop Ruhestein)".
At the beginning the watergate may cover only a single ship and no traffic lights.
Afterwords you may improve you system!