TAG DER INFORMATIK

DAY OF COMPUTER SCIENCE 2025

Thursday, 5 June

```
$ JOB_FAIR.JSON < {
    "Time": "10:30 - 16:00",
    "Location": "Computer Science Center (E2)",
    "_info": "Get ice cream and party wristbands!"
}

$ PARTY.JSON < {
    "Time": "20:00 - 00:30",
    "Location": "Das LIEBIG, Liebigstr. 19",
    "_info": "Wristband required!"
}
```

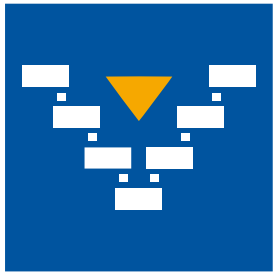
More info at www.tdi.ac

INFORM itestra be excellent
CONSILEON LANCOM SYSTEMS VECTOR
MAGMA WORLDLINE think-cell amprion
REWE DIGITAL DSA CISCO
ModuleWorks SOPTIM oculavis IVU TRAFFIC TECHNOLOGIES EPG Ehrhardt Partner Group

Organised for the Department of Computer Science of RWTH Aachen by the Chair of Artificial Intelligence Methodology (i14)
Theaterstrasse 35-39, 52062 Aachen | Contact: Katharina Isabel Franke | tdi@cs.rwth-aachen.de

Informatik 11
Embedded Software

RWTH AACHEN UNIVERSITY



12.24196

Introduction to Embedded Systems

Prof. Dr.-Ing. Stefan Kowalewski | Julius Kahle, M. Sc.

Summer Semester 2025

Part 4

Real-Time Systems

Content

1. Real-Time Requirements
2. Real-Time Operating Systems – Example OSEK
3. Scheduling
4. Deadlocks & Priority Inversion
5. Priority Ceiling & Inheritance

Real-Time Requirements

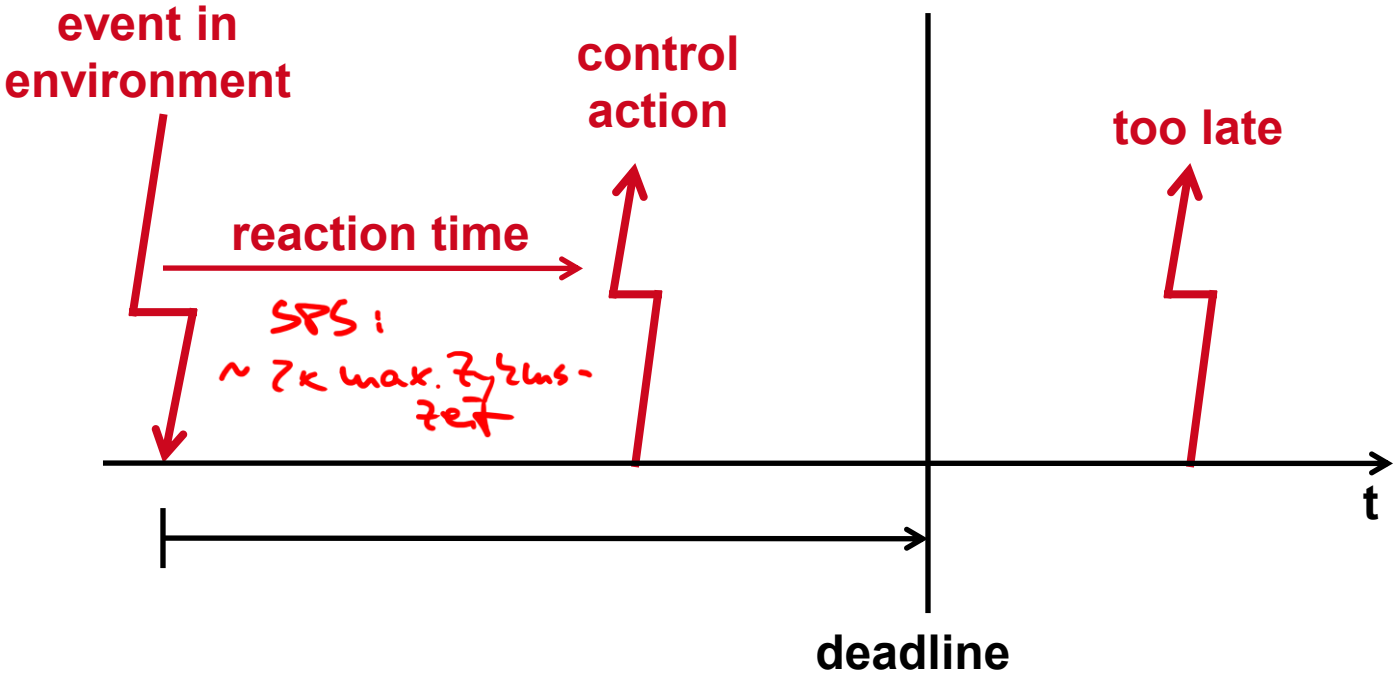
Real time ~~=~~ fast~~?~~

- Computation is correct
- Computation is finished in time

Real-Time Requirements

For embedded systems:

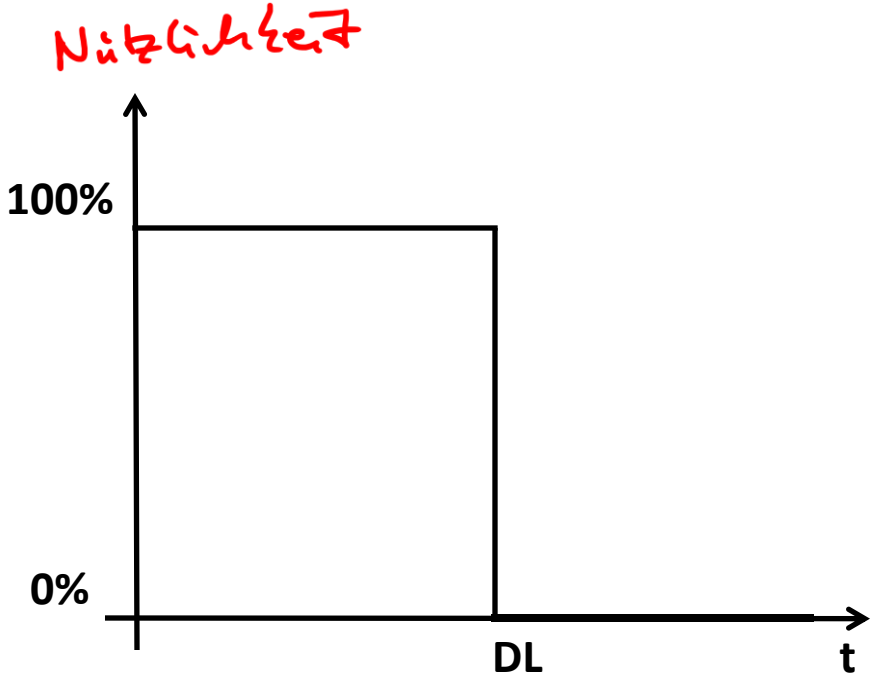
Physical environment does not wait for control actions!



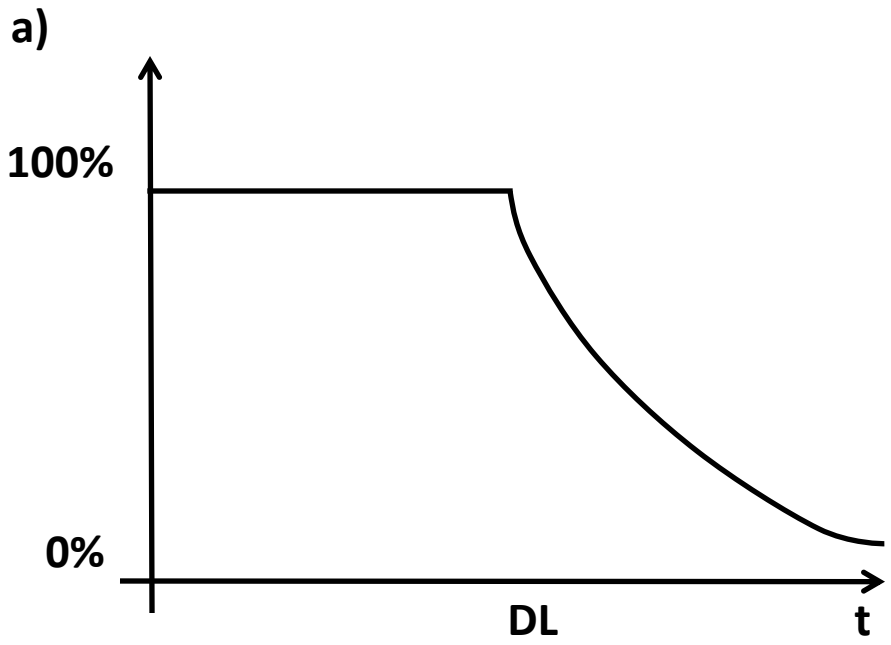
Real-Time Requirements

Hard vs. soft real-time requirements

Hard:



Soft:



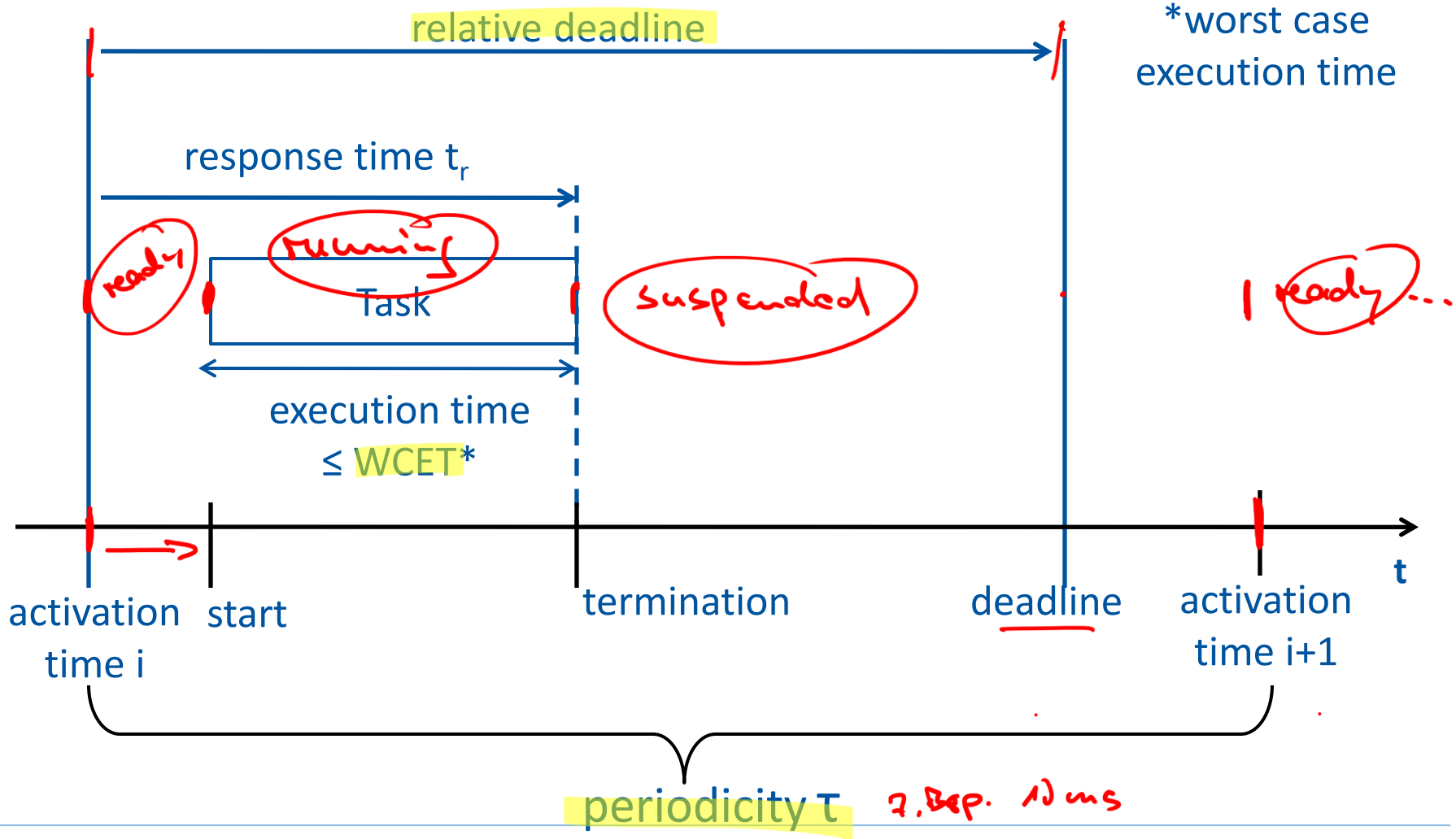
b) statistically a sufficient number of reactions in time.

How to Fulfill Real-Time Requirements?

- Polling
- Main loop and interrupts
- Real-time operating system

1 Prozessor
n tasks

Real-Time Requirement Parameters for Multitasking OS



OSEK: History

1993: Start of project

„Offene Systeme und deren Schnittstellen für Elektronik im Kraftfahrzeug“

(Coordinator: Prof. Kiencke, University of Karlsruhe)

1994: PSA and Renault joinedwith French initiative

„VehicleDistributed Executive“ (VDX) ➔ OSEK/VDX

Today: ISO standard

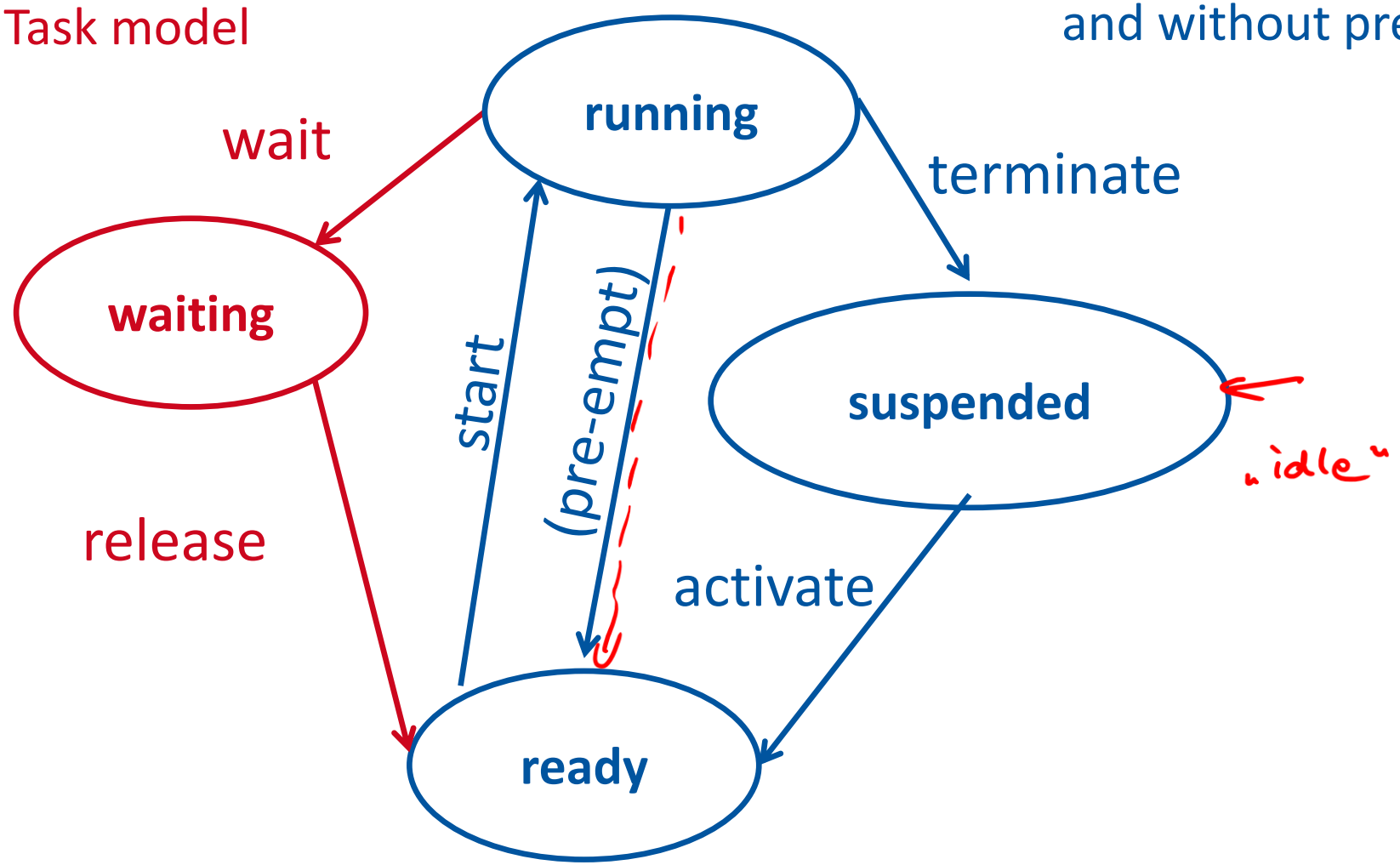
AUTOSAR OS VECTOR, ETAS

OSEK: Task Model

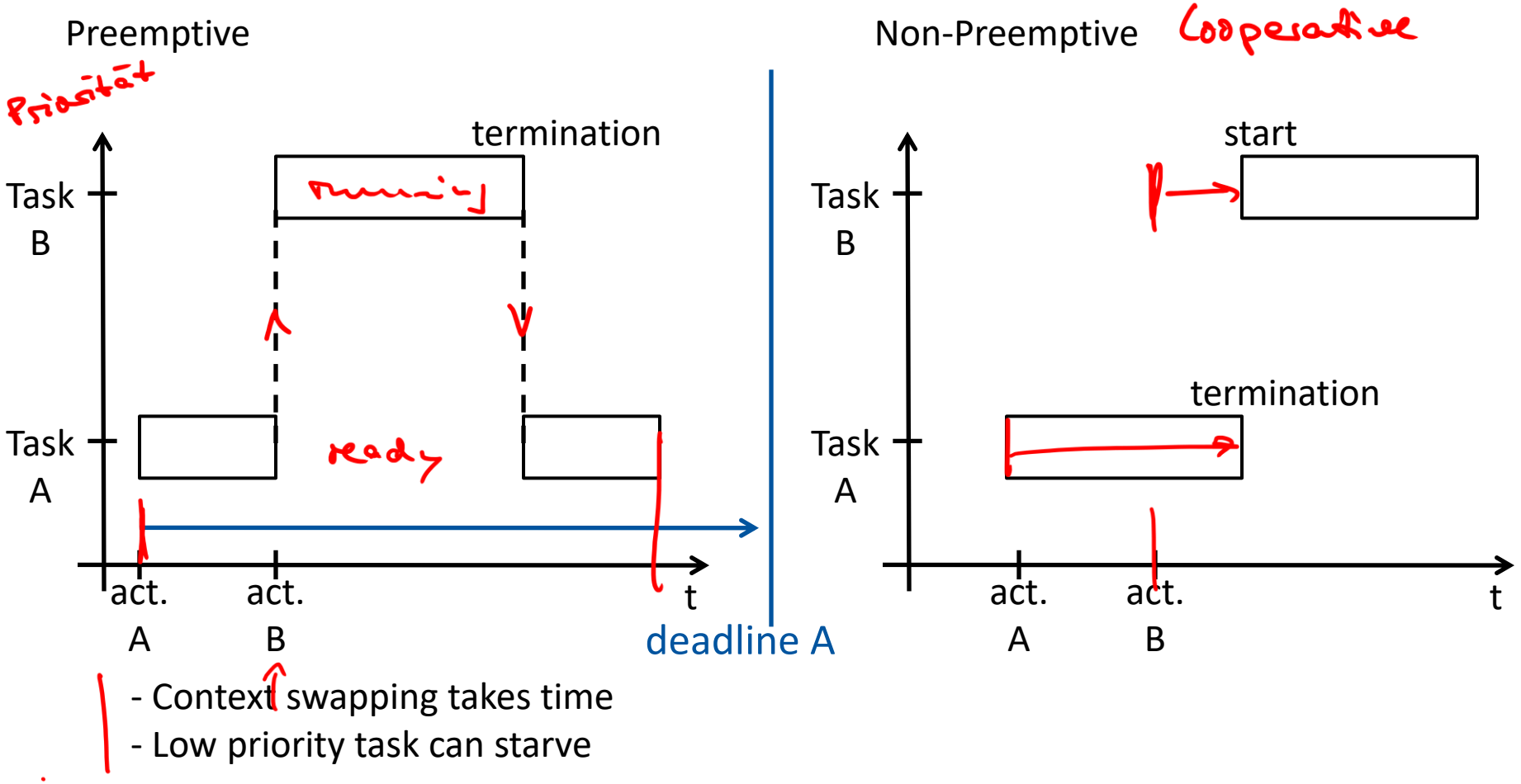
(für die Zustände, in denen sich Tasks befinden)

Extended Task model

Basic Task model with and without preemption

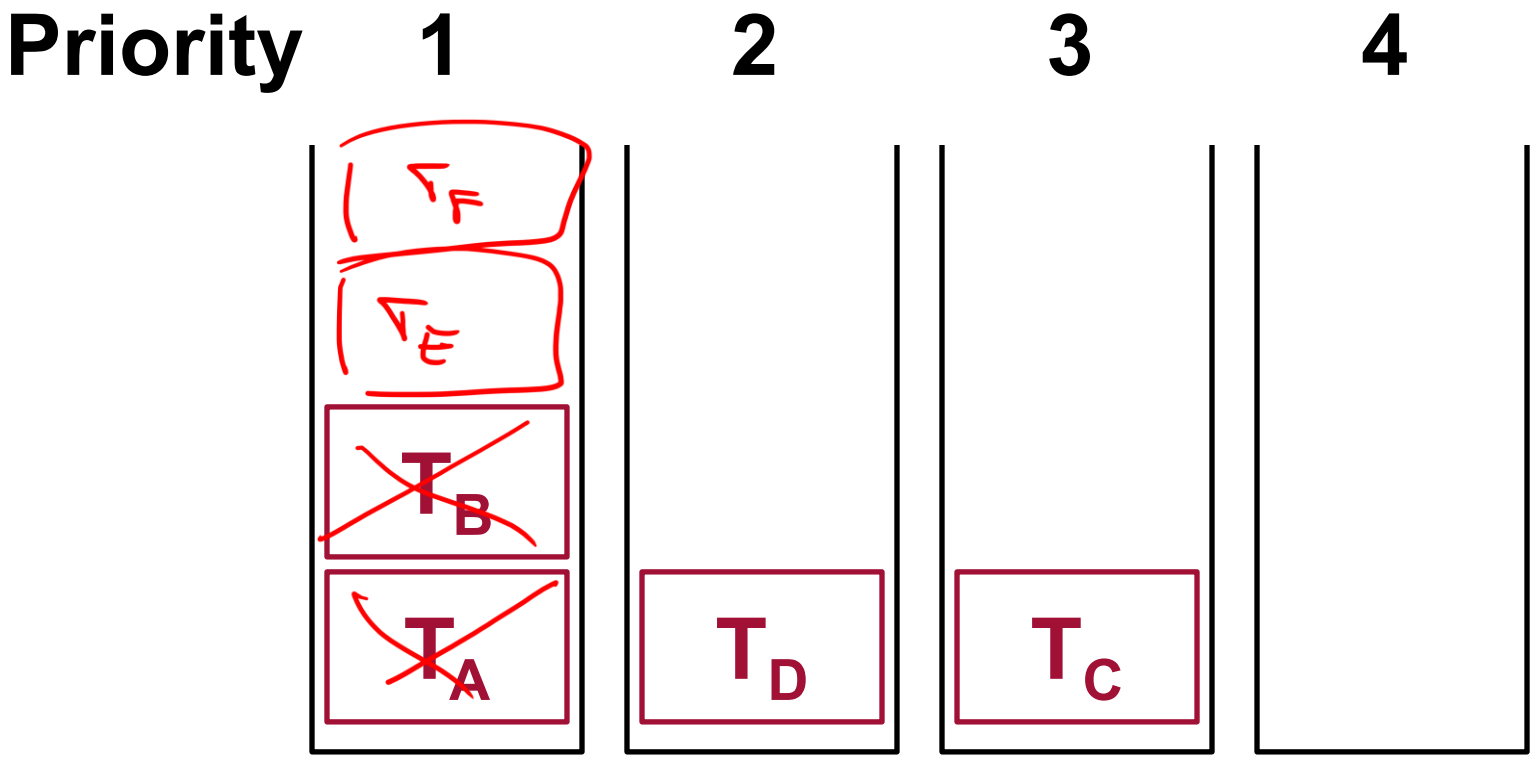


OSEK: Cooperative Scheduling – Pre-emption



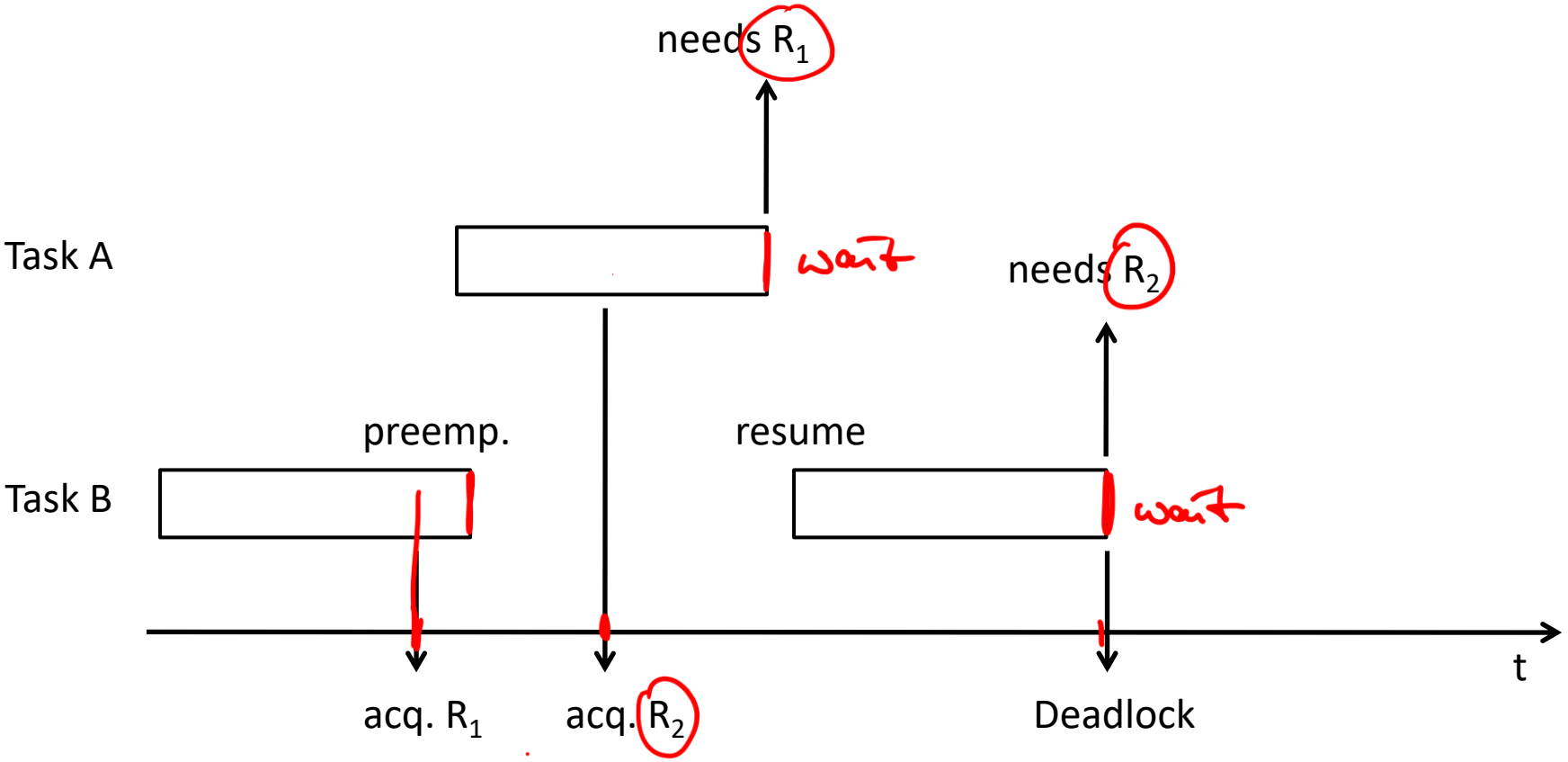
- Context swapping takes time
- Low priority task can starve

OSEK: Priority Queues



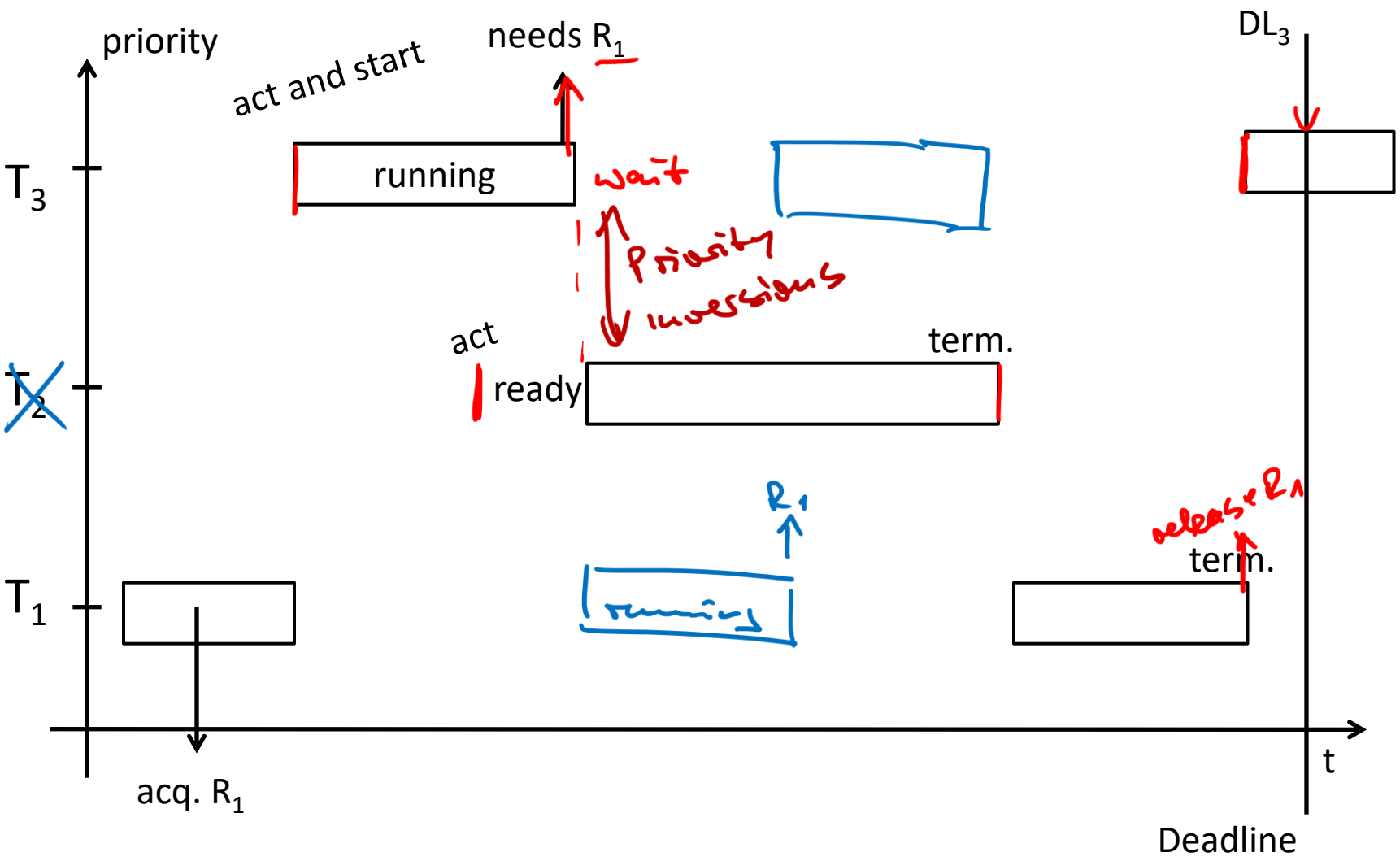
Problem: Deadlock

Extended Task Modell
2 Tasks, 2 Ressourcen



Problem: Priority Inversion

3 Tasks, 1 Ressource



Example for Priority Inversion: Mars Pathfinder Project

Elements:

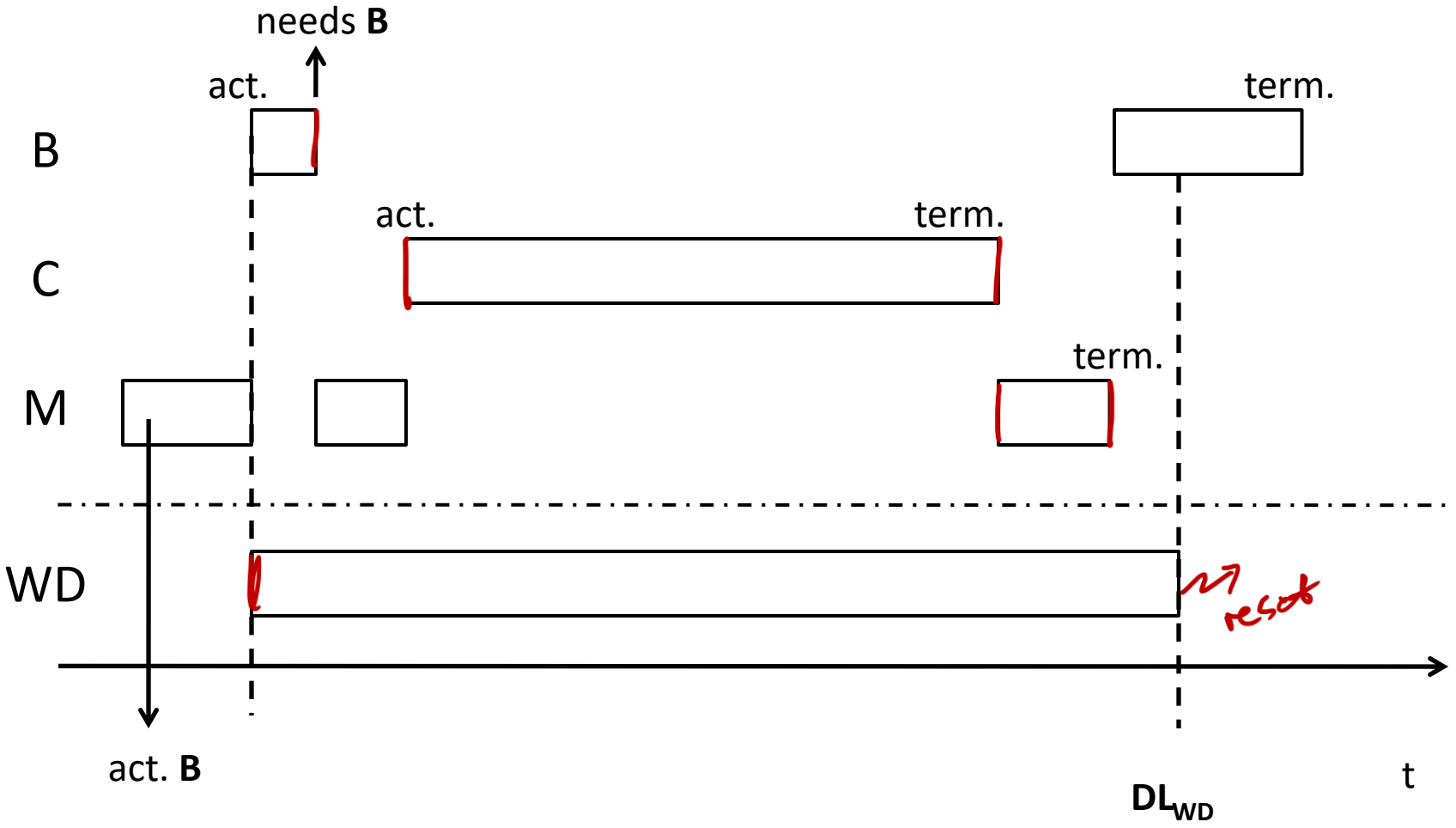
- One resource:
  - **B:** Information bus
- Three tasks:

B:	Bus management	needs **B**	highest priority
C:	Communication	does not need **B**	medium priority
M:	Metrological data gathering	needs **B**	lowest priority

- Watchdog timer WD, reset by bus management

Example for Priority Inversion: Mars Pathfinder Project

Problem:



Priority Inheritance Protocol

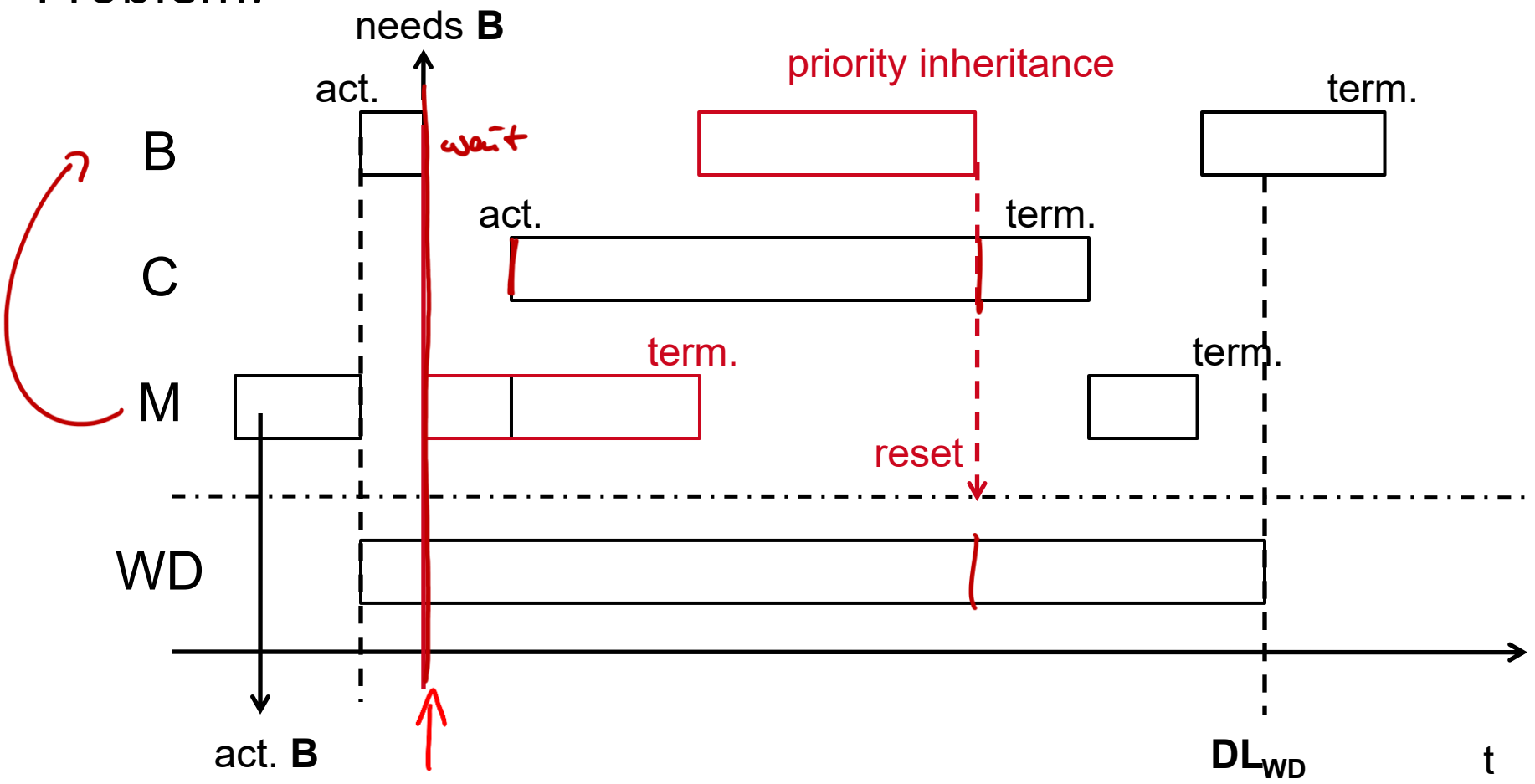
- Task A (high priority) needs a resource R
- Task B (low priority) currently holds R
- As soon as A starts waiting for R
  - B inherits A's priority

  ⇒ B cannot be preempted by an intermediate priority

  ⇒ No priority inversion possible
- Complex:
  - Current holder of a resource must be determined
  - Holder's priority must be changeable while running

Example for Priority Inversion: Mars Pathfinder Project

Problem:



Priority Ceiling Protocol

- **Ceiling priorities** are assigned to the resources
- A resources priority is the highest priority of all tasks that may use it
- A task is upgraded to the resource's priority as soon as the task holds it.
- Can be calculated during design time
- Not optimal: T2 is upgraded to R_1's priority even if T4 is inactive

high T5

T4 → R_1 — Ceiling priority of R_1 = Prio(T4)

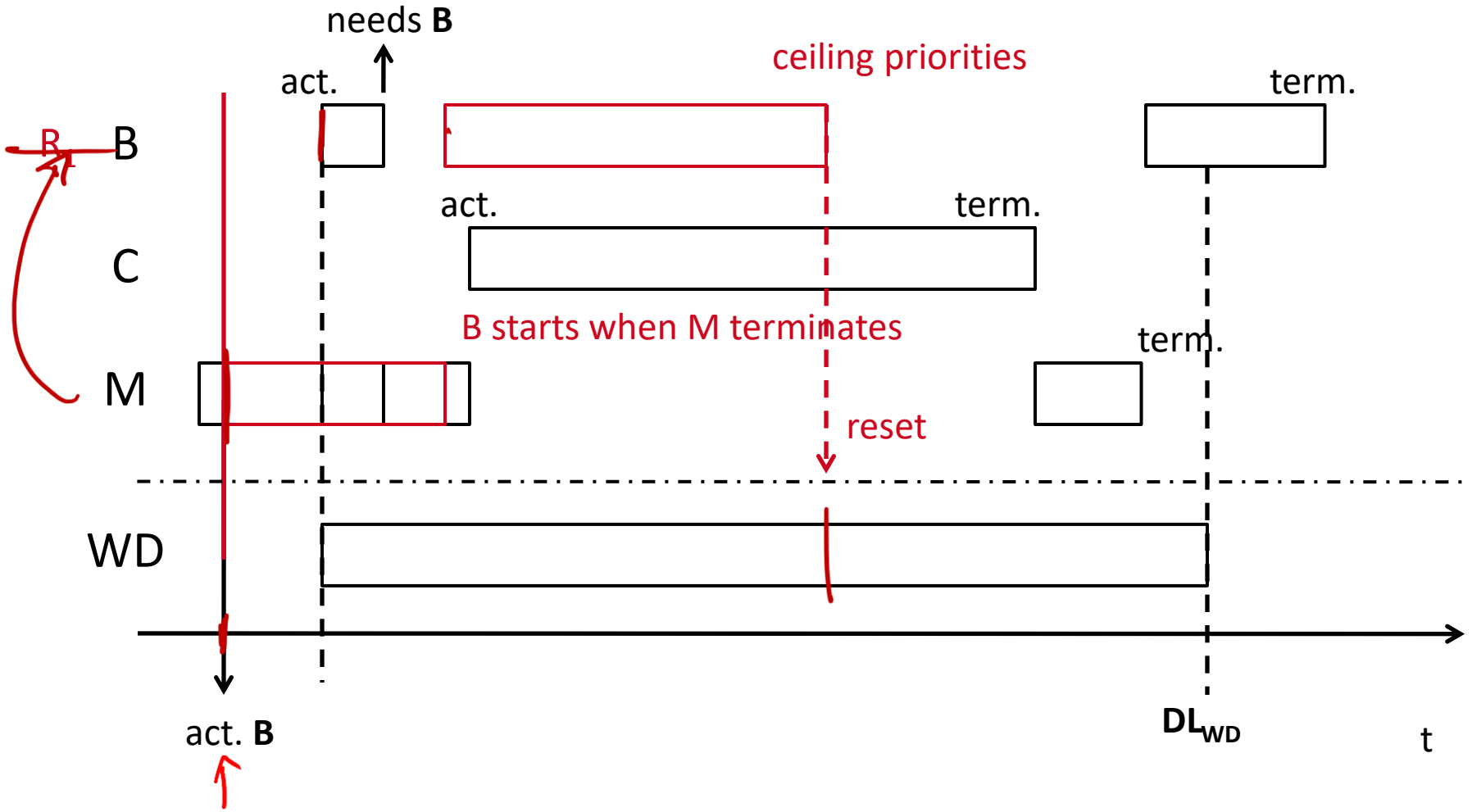
T3

T2 → R_1

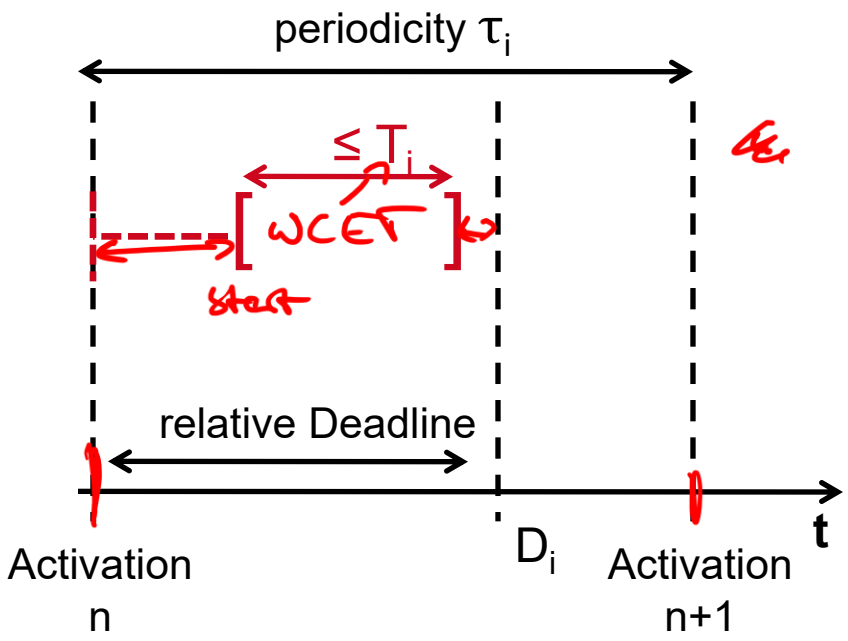
low T1

Example for Priority Inversion: Mars Pathfinder Project

Problem:



Reminder: Task parameters



A Task is characterized by

$\mathbf{T}_i = (\tau_i, T_i, D_i)$

i=1,2,… → Task System

Definitions

Schedule: (Plan)

Mapping of an execution sequence to a task system

Feasibility: (Planbarkeit)

A schedule is feasible, if no deadline is violated.

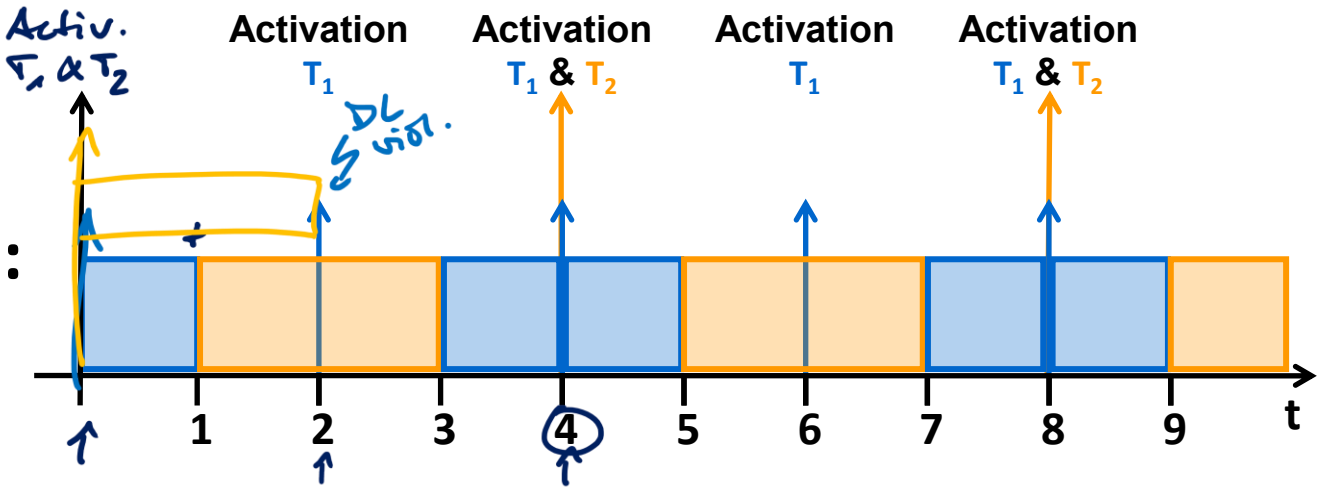
Schedulability:

A task system is schedulable, if a feasible schedule exists.

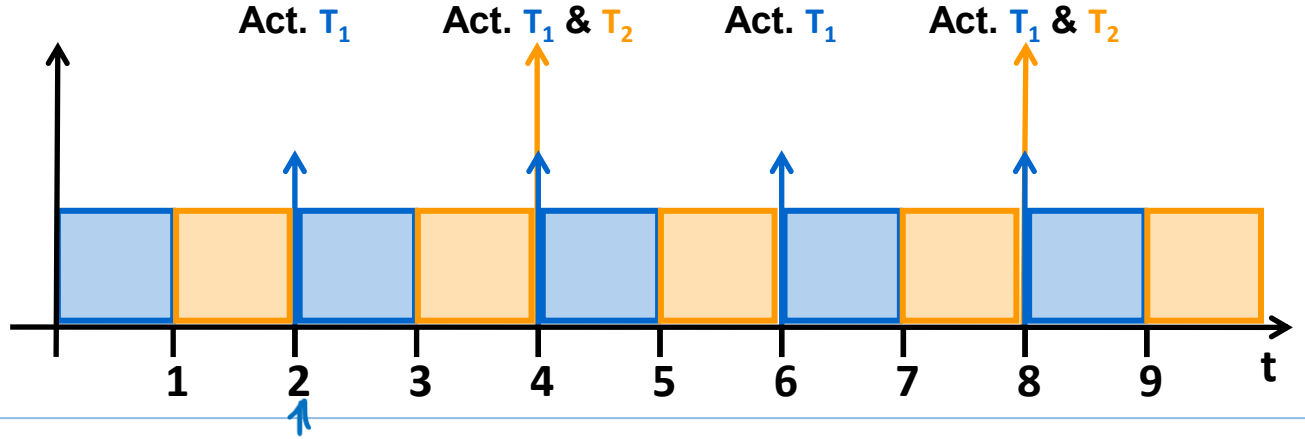
Example 1 - Cooperative vs. Preemptive Scheduling

$T_1 = (2, 1, 2), T_2 = (4, 2, 4)$

Cooperative:



Preemptive:

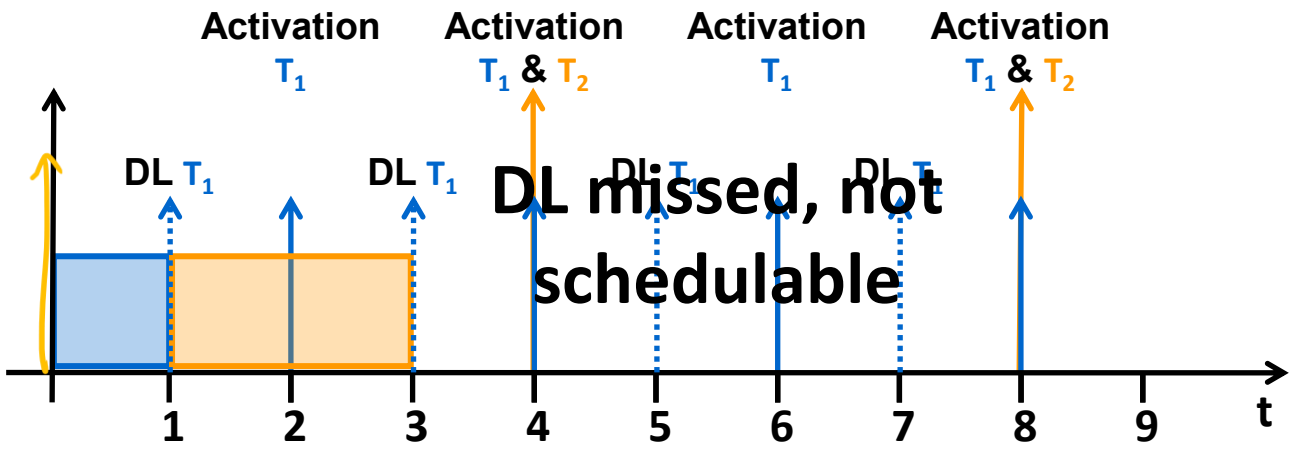


Example 2 - Cooperative vs. Preemptive Scheduling

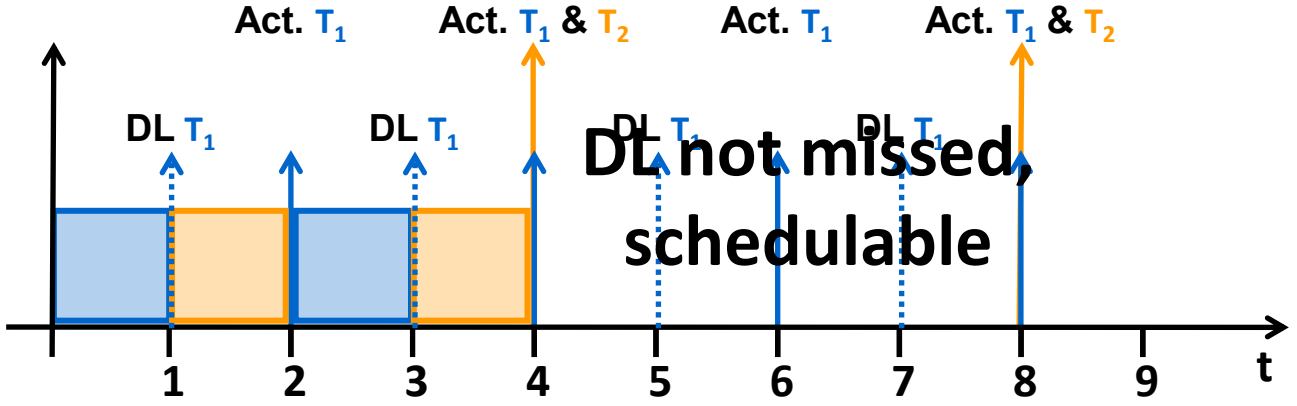
$T_1 = (2, 1, 1)$, $T_2 = (4, 2, 4)$

$Prio(T_1) > Prio(T_2)$

Cooperative:

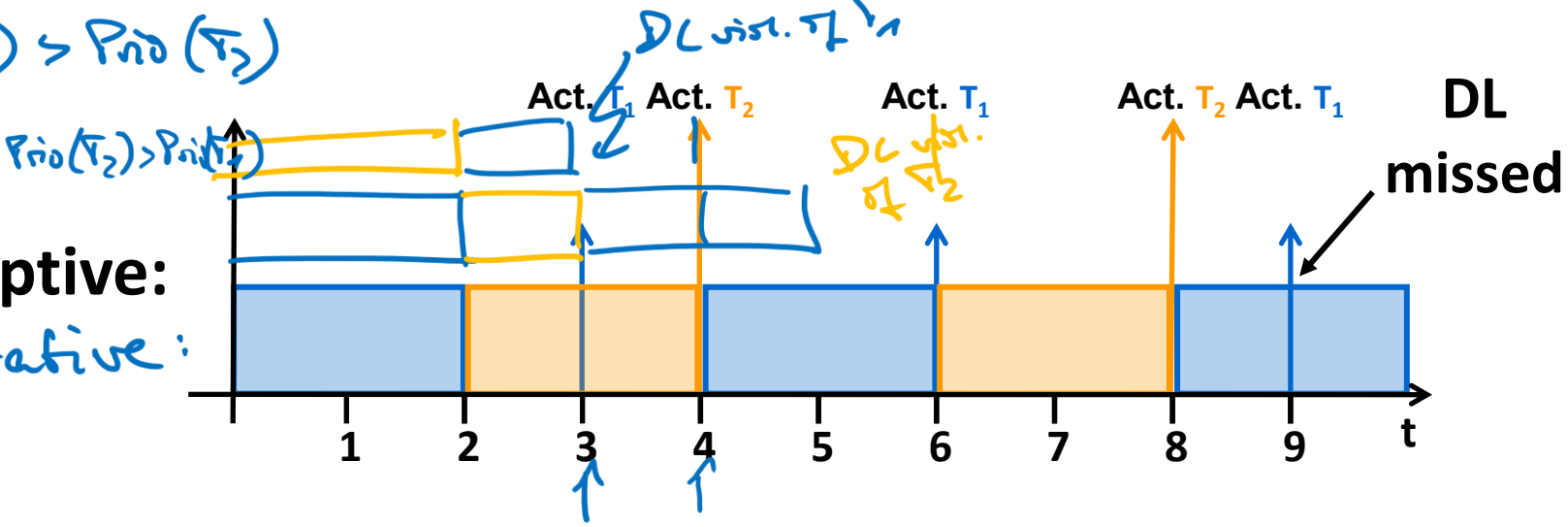


Preemptive:



Example 3 – Cooperative vs. Preemptive Scheduling

$T_1 = (3, 2, 3)$, $T_2 = (4, 2, 4)$



Not schedulable using preemption

→ scheduling impossible

Utilization

What was the problem in example 3?

→ CPU Utilization *U*

$$U = \sum_{i=1}^{m} \frac{T_i}{\tau_i}$$

Anteil einer Task T_i am Zeitbedarf

Example 3: $U = \frac{2}{3} + \frac{2}{4} = \frac{14}{12} > 1$

U > 1 => Task system is not schedulable.

U ≤ 1 necessary condition for schedulability.

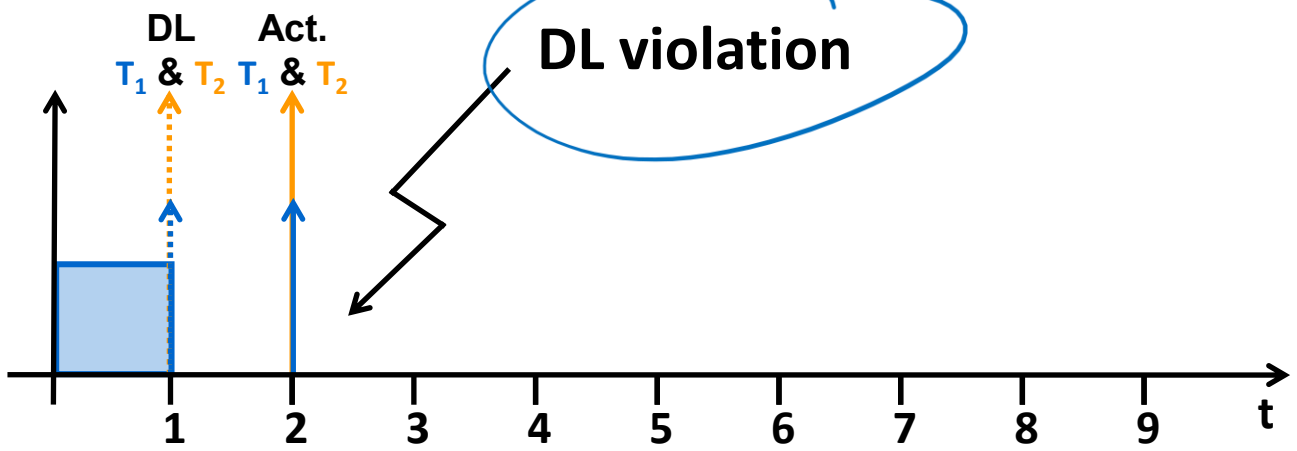
Questions

1. Is every task system with $U \leq 1$ schedulable?
   Or: Is $U \leq 1$ a sufficient condition for schedulability? nein
2. Is there an algorithm which generates a feasible schedule for every schedulable task system? (i.e. an „optimal“ algorithm)

Example 4 - Utilization

$T_1 = (2, 1, 1)$, $T_2 = (2, 1, 1)$

$$U = \frac{1}{2} + \frac{1}{2} = 1$$



Earliest Deadline First Scheduling

EDF

Liu, Layland 1973

Execute the task with smallest D first.

Rate Monotonic Scheduling

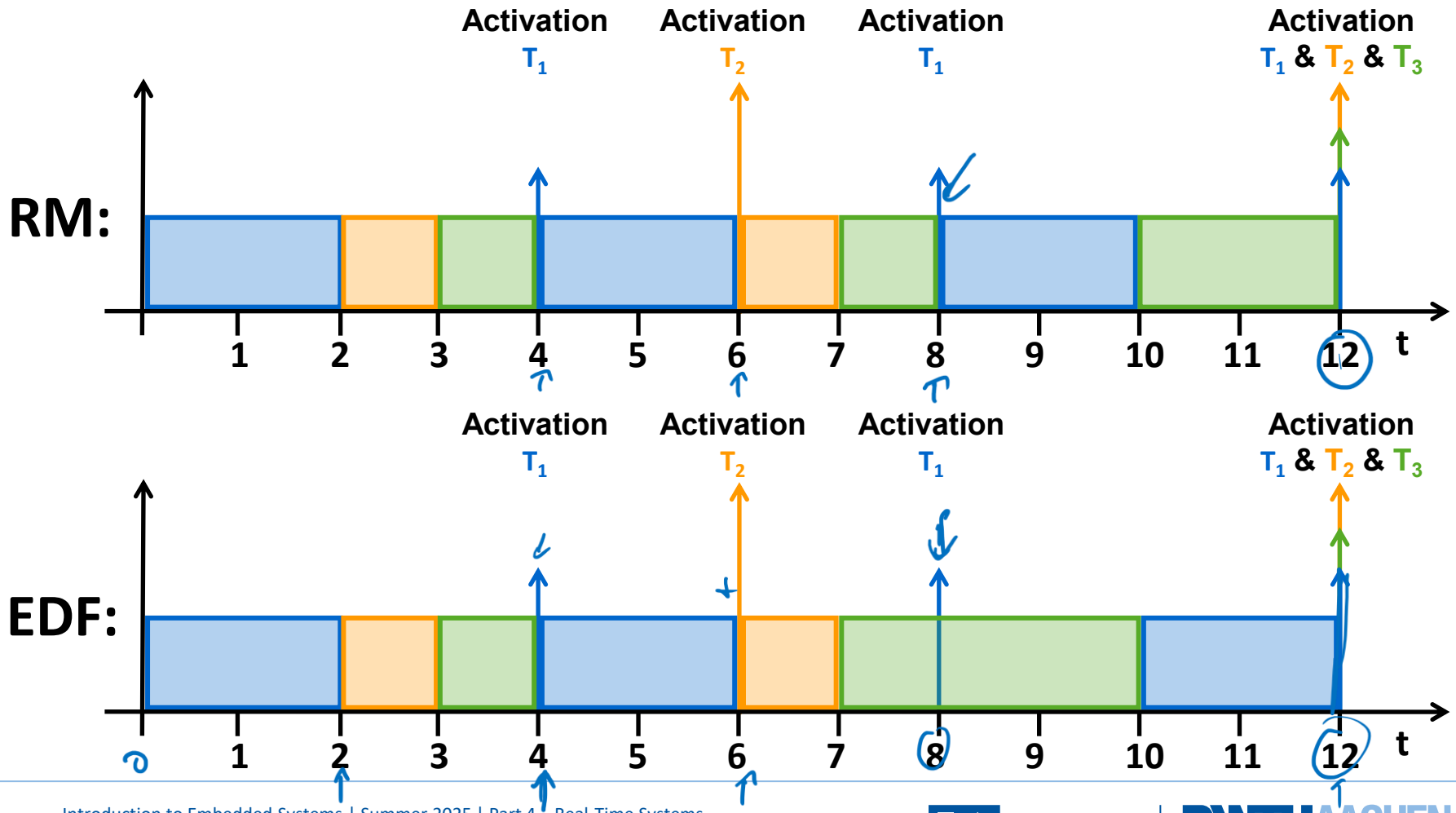
$$\text{Rate} = \frac{1}{\tau} = \frac{1}{\text{Periodizität}}$$

$$\text{Prio}(T_1) > \text{Prio}(T_2) > \cdots > \text{Prio}(T_m) \iff \frac{1}{\tau_1} > \frac{1}{\tau_2} > \cdots > \frac{1}{\tau_m}$$

A task system is RM-schedulable, if a feasible RM-schedule exists.

Example 5 – RM vs. EDF

$T_1 = (4, 2, 4)$, $T_2 = (6, 1, 6)$, $T_3 = (12, 4, 12)$ **U=1** $\text{Prio}(T_1) > \text{Prio}(T_2) > \text{Prio}(T_3)$



Example 6 – RM vs. EDF

$T_1 = (4, 2, 4)$, $T_2 = (5, 2, 5)$, $T_3 = (10, 1, 10)$ U=1 $\text{Prio}(T_1) > \text{Prio}(T_2) > \text{Prio}(T_3)$