AUTOSAR

CPU Load

How to avoid common pitfalls + a new approach

Peter Gliwa

16th AUTOSAR Open Conference

27/28 May 2025 – Version 4

Bruges

Agenda

- Introduction
- CPU load
- TASK load (new idea, proposal)
- Summary

Agenda

- **Introduction**
- CPU load
- TASK load (new idea, proposal)
- Summary

What is ‘load’?

rob-bier.medium.com

Source: BSE-Galerie

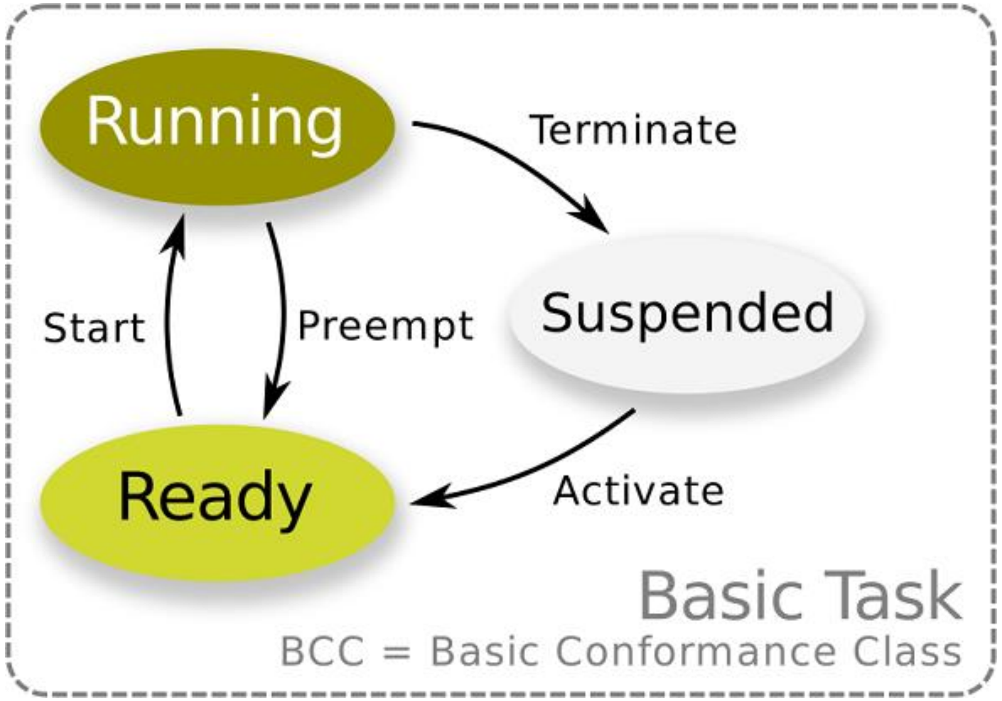
Overload

Load

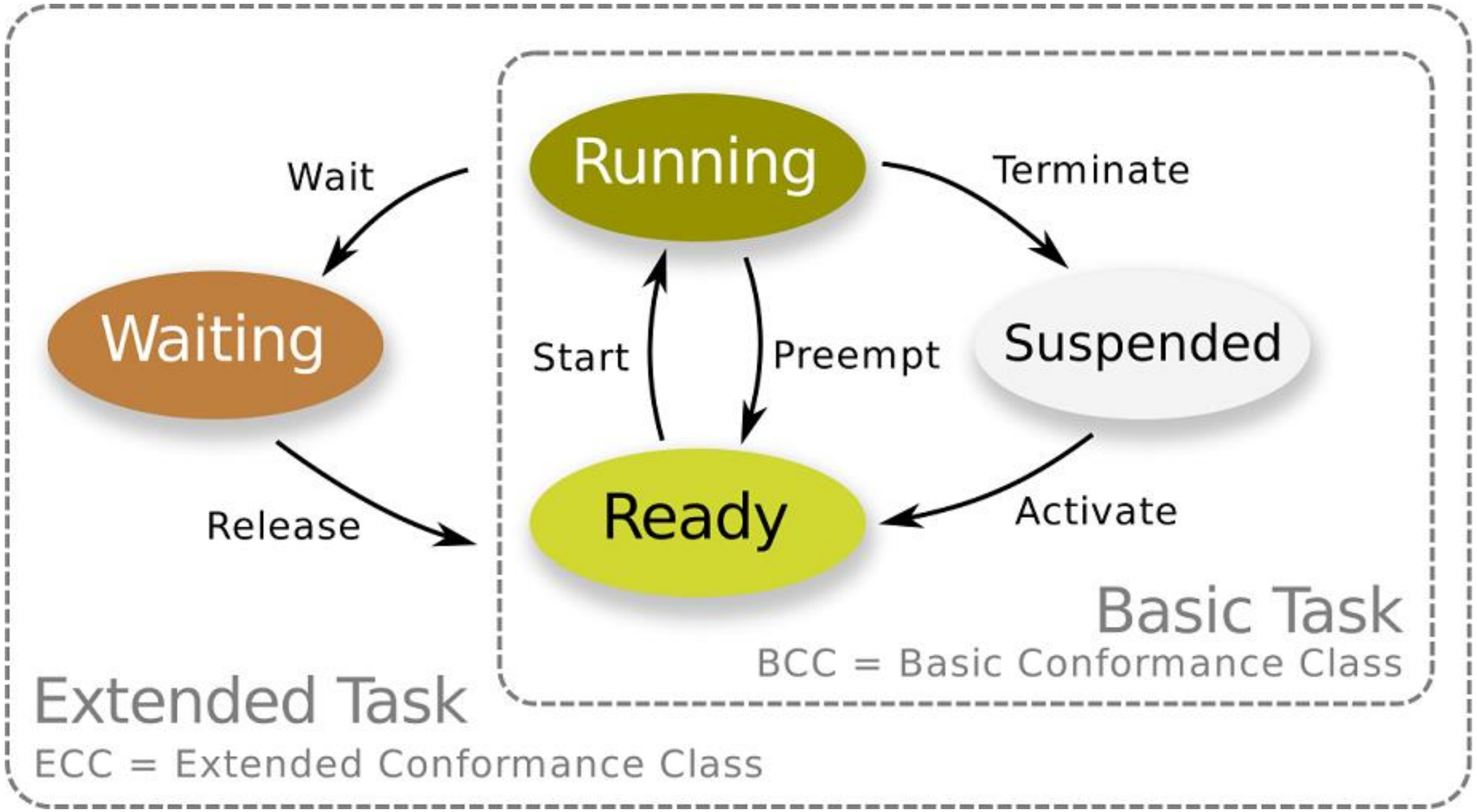
Embedded Software Timing

- Some of the following tables, pictures, etc. are taken from my book **Embedded Software Timing**
- Available in five languages: DE, EN, CN, KR, JP

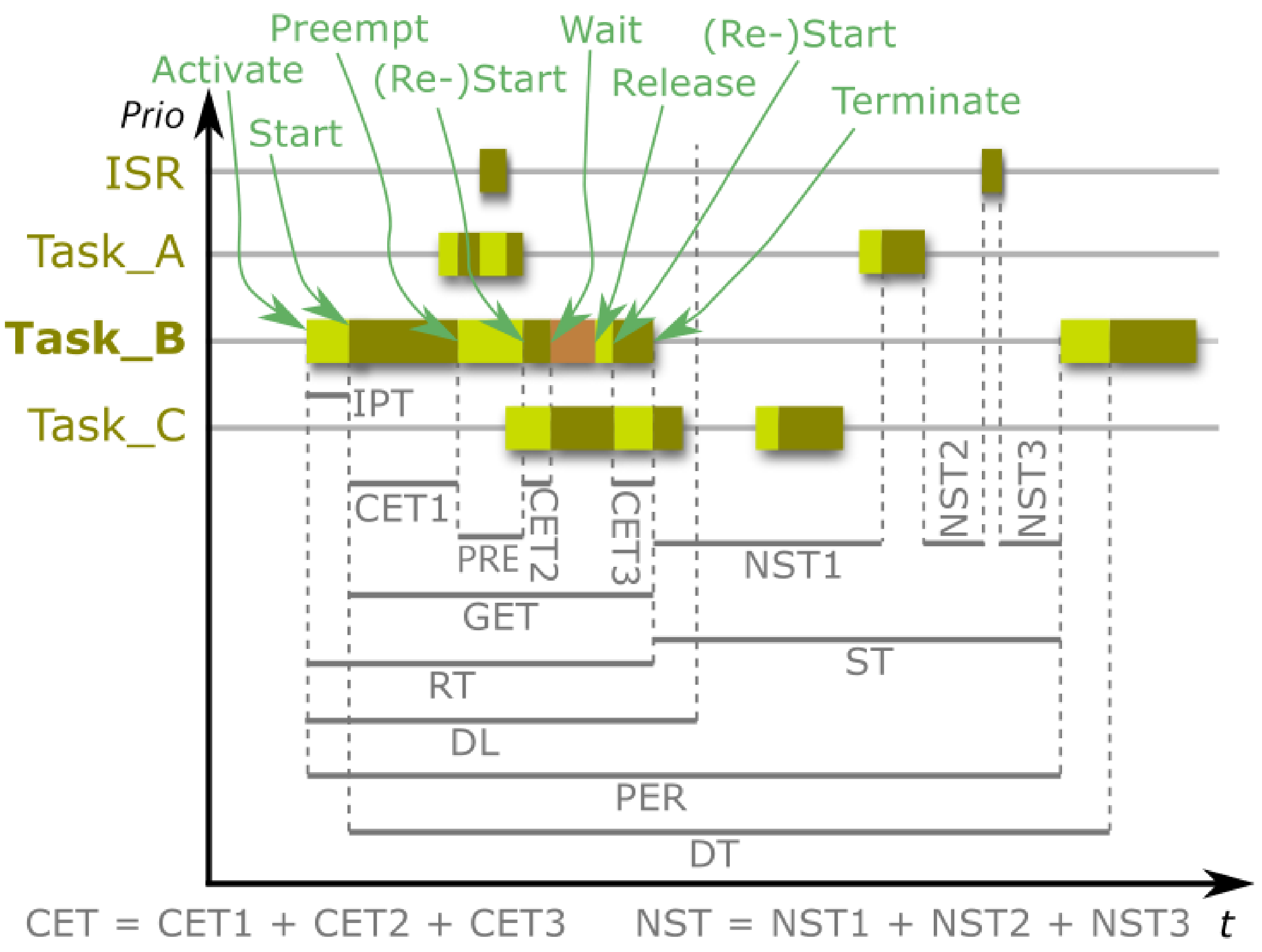
AUTOSAR / OSEK Task States (BCC)



AUTOSAR / OSEK Task States (BCC & ECC)



Timing Parameters



- **IPT (Initial Pending Time)**
  Ready time before task starts
- **CET (Core Execution Time)**
  Time spent in running state, i.e. executing
- **GET (Gross Execution Time)**
  From start to termination (cf. pin toggle)
- **PRE (PREemption Time)**
  Sum of ready times without IPT
- **RT (Response Time)**
  cf. schedulability analysis; DL (DeadLine) = limit for RT
- **Period (PERiod)**
  time difference between two subsequent activations
- **DT (Delta Time)**
  time difference between two subsequent events of the same type; observed period; cf. jitter
- **ST (Slack Time)**
  duration of the ‘gap’
- **NST (Net Slack Time)**
  headroom: time which could be added to CET

Relevant for **CPU load** calculation

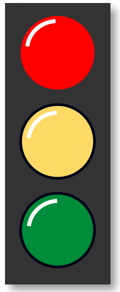
Relevant for **TASK load** calculation

Agenda

- Introduction
- **CPU load**
- TASK load (new idea, proposal)
- Summary

CPU Load: Manager’s Darling

- CPU load – managers love it!
  - Complicated scheduling/timing world reduced to one single number
  - Even traffic lights can be derived
    - **>85% = red**
    - **70% .. 85% = yellow**
    - **< 70% = green**



BTW: AUTOSAR does not ‘know’ CPU load although it is the most widely used timing parameter.

- Engineers: don’t smile at this approach!
- Let this expectation guide the definition/calculation of the CPU load!

CPU Load: Definition & Calculation

CPU load

$$U = \frac{t_e}{t_o}$$

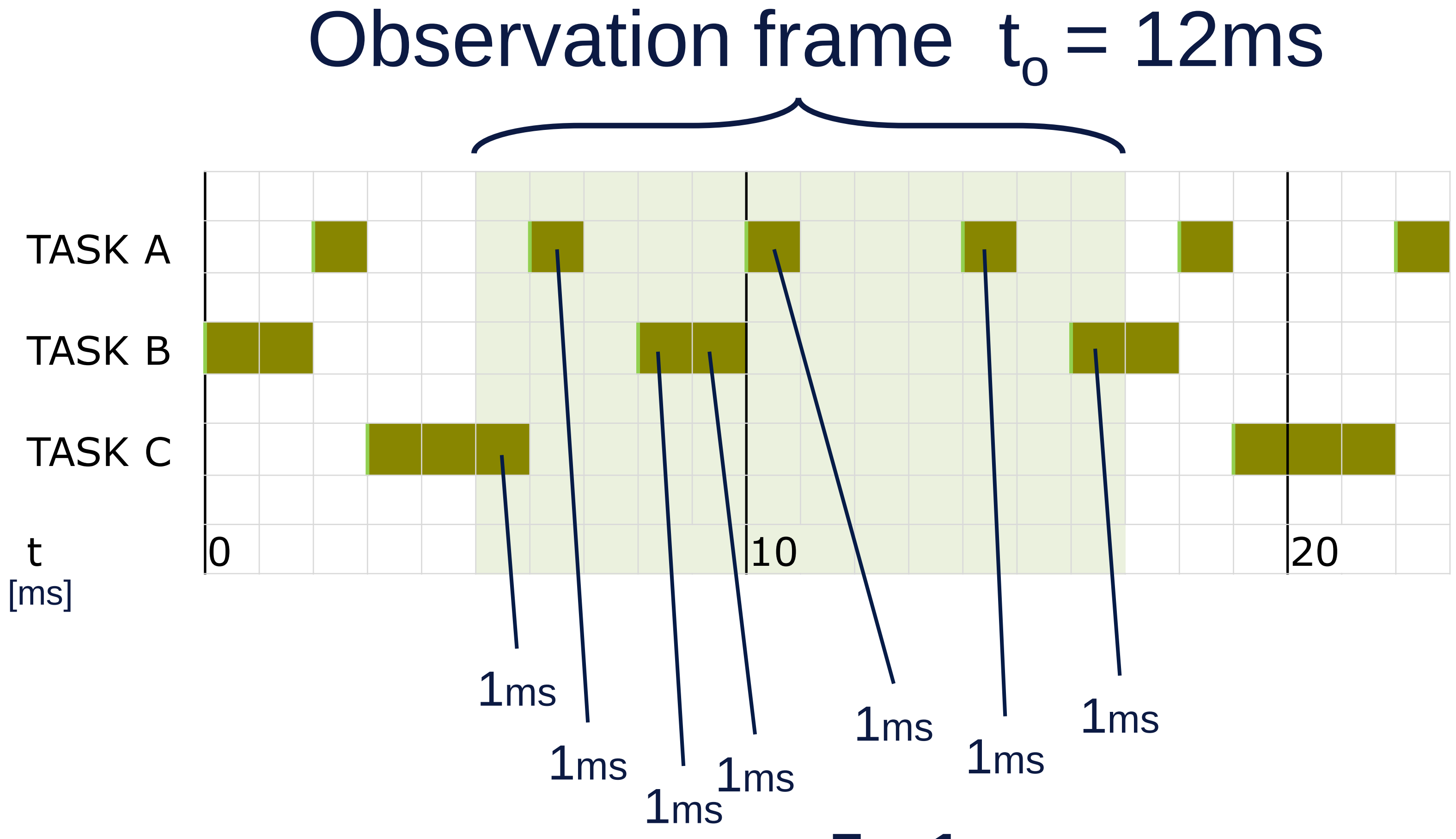
t_e in other words: time the CPU spent processing code other than idle code

t_o: duration of the observation

$$t_e = \sum_{n=1}^{N} CET_n$$

t_e is the sum of all CETs (TASKs and ISRs) that fall into the observation.

CPU Load: Example

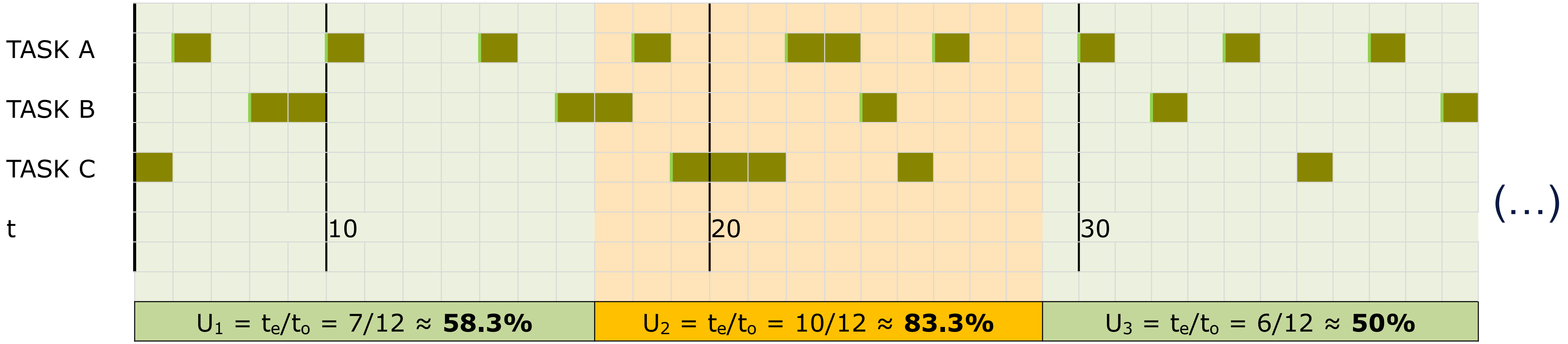


7 x 1ms

$U = t_e / t_o$
= 7ms / 12ms
≈ **58.3%**

‘CPU Load’ Means ‘Max. CPU Load’

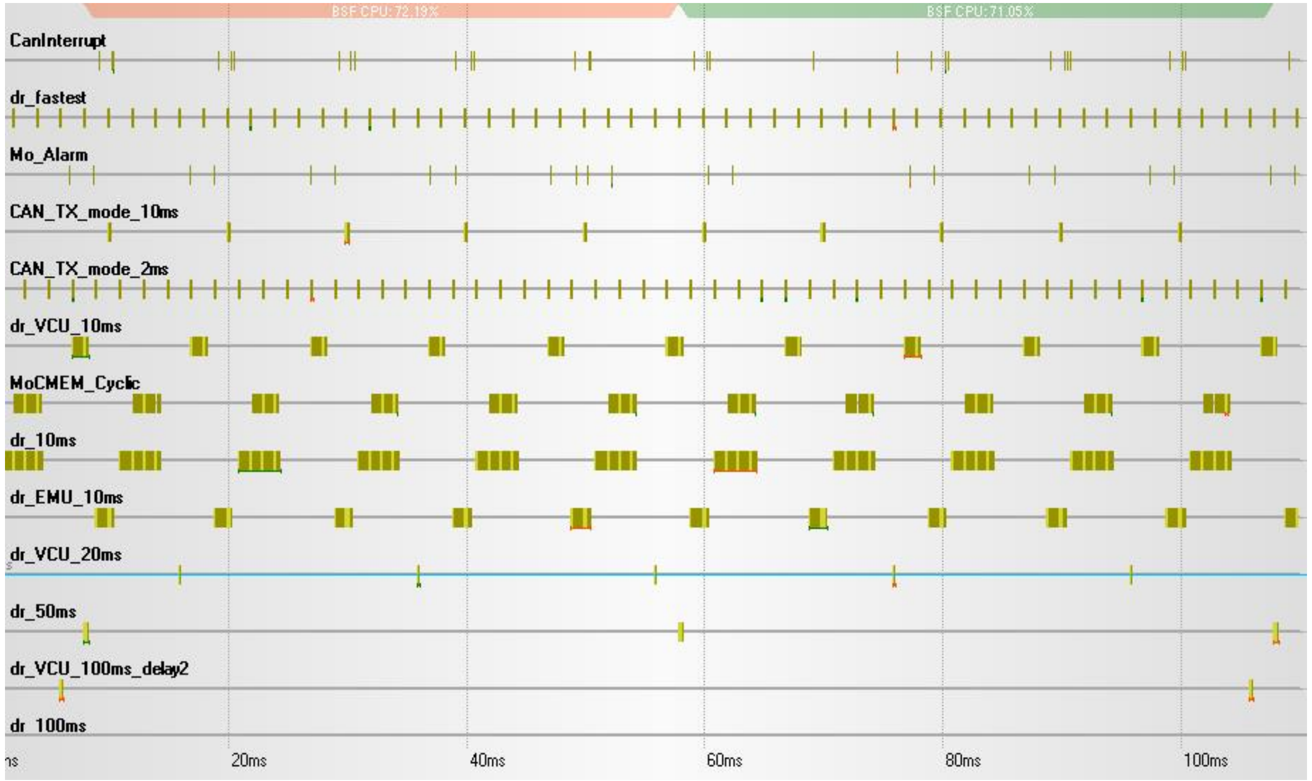
The end of one observation frame marks the beginning of the next one.



Which of the CPU loads is ‘the’ CPU load? Max? Average?

$$U = \max(U_1..U_n)$$

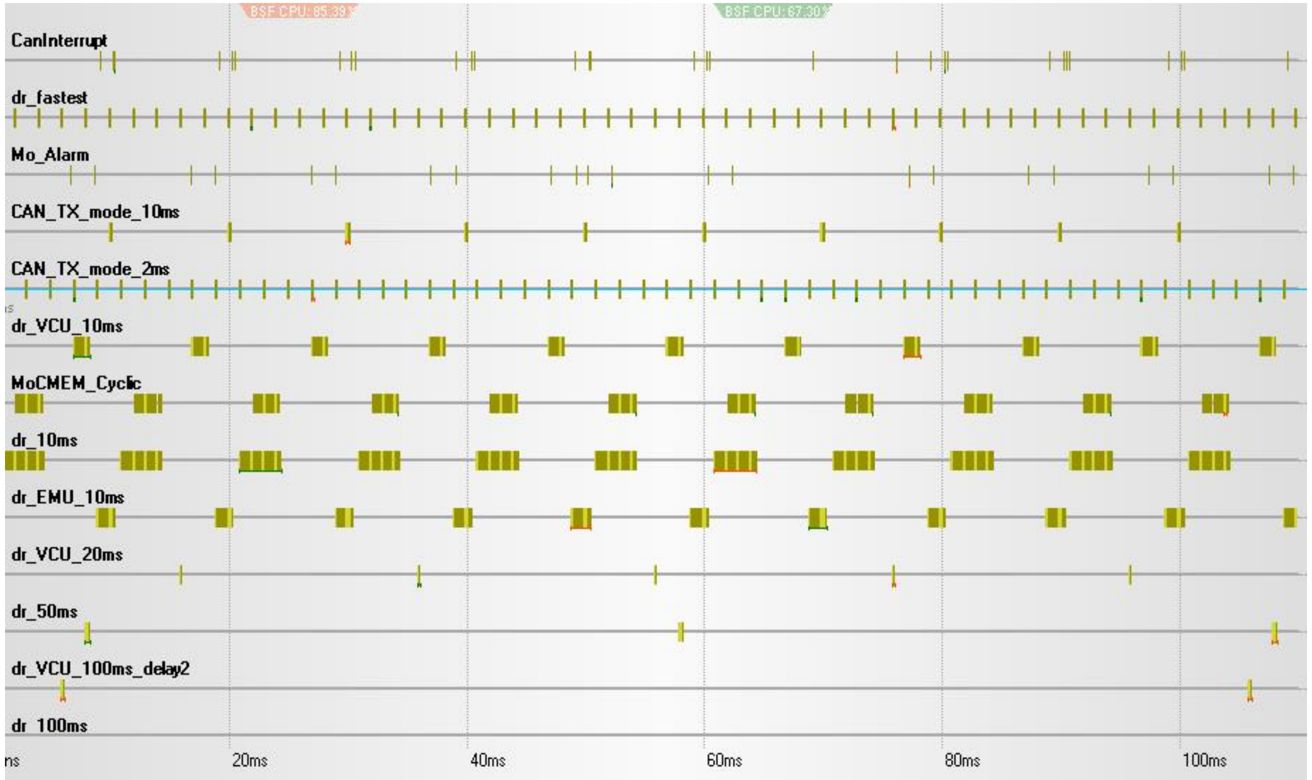
CPU Load: **Big** Observation Frame



- BSF-event(*) here: activation of **50ms** TASK
- The resulting max. CPU load is **72.19%**

*: **B**asic **S**cheduling **F**rame event defining t_o

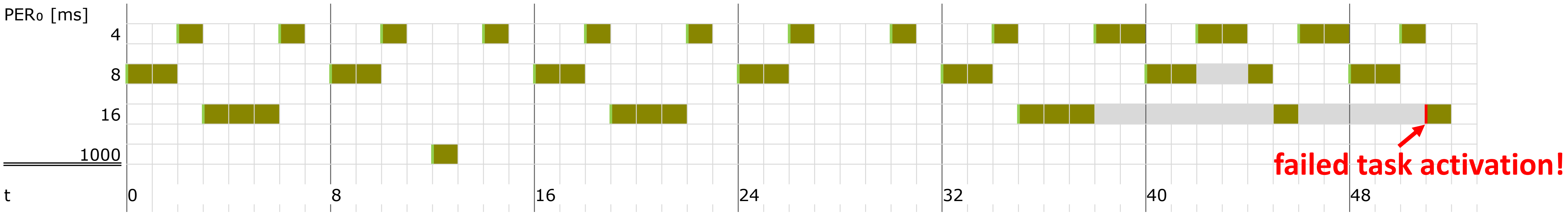
CPU Load: **Small** Observation Frame



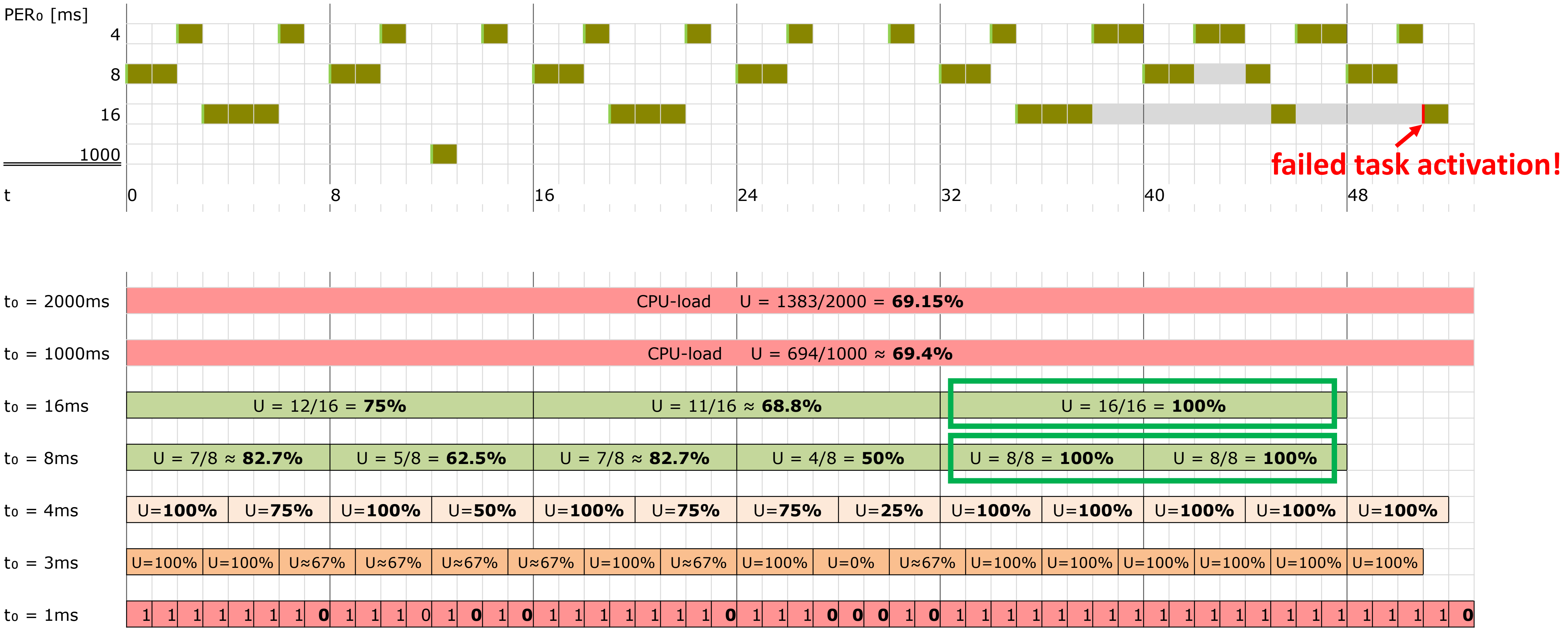
- BSF-event(*) here: activation of **10ms** TASK
- The resulting max.
  CPU load is
  85.39%

*: **B**asic **S**cheduling **F**rame event defining t_o

CPU Load: How to Select the Observation Frame



CPU Load: How to Select the Observation Frame



Which observation frame should you use?

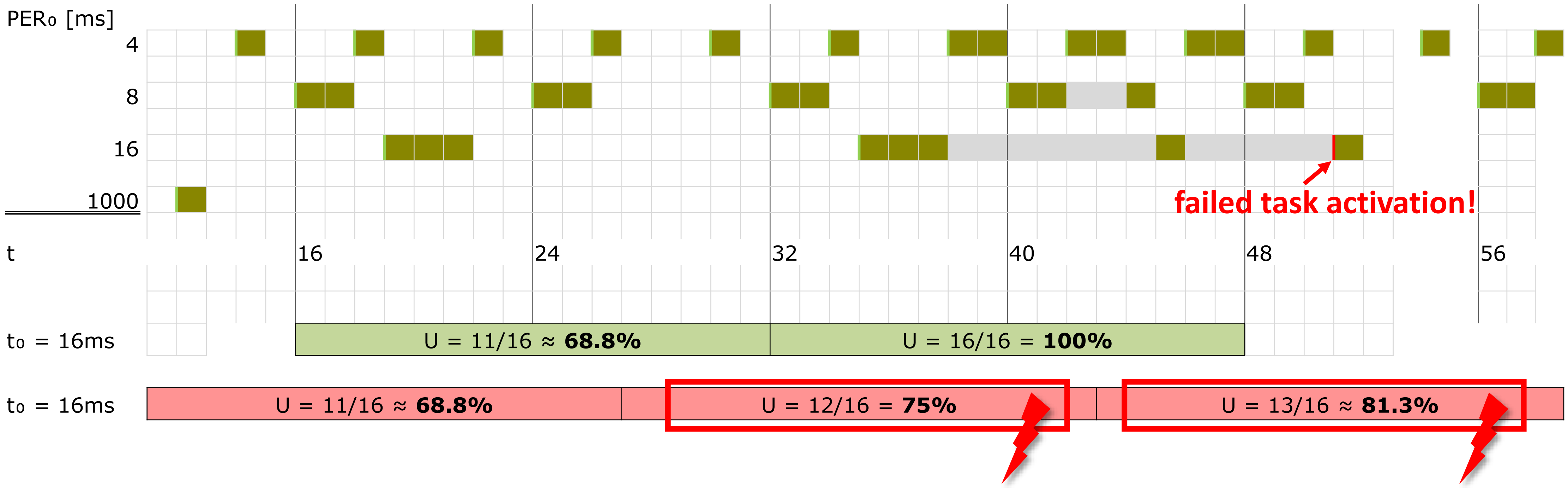
- Unfortunately, there is no silver bullet.
- The ‘right’ observation frame depends on the project’s schedule and requirements.
- However, with the skills you just gained, you should be able to pick the right t_o.
- The main control loop (if there is one) is typically a good starting point.
- t_o will probably be in the range 10ms to 50ms.

Agenda

- Introduction
- CPU load
- **TASK load (new idea, proposal)**
- Summary

Are we there yet?

Not quite. Even with t_o = 16ms, there is a problem if we shift the observation frame a bit.



Use a sliding window?

Cf. moving/sliding average vs. intermediate average

<table>
<tr><td>Index i</td><td>1</td><td>2</td><td>3</td><td>4</td><td>5</td><td>6</td><td>7</td><td>8</td><td>9</td><td>10</td><td>11</td><td>12</td></tr>
<tr><td>Value x_i</td><td>96</td><td>20</td><td>28</td><td>36</td><td>53</td><td>27</td><td>41</td><td>32</td><td>62</td><td>36</td><td>73</td><td>21</td></tr>
<tr><td>Overall average</td><td colspan="12">44</td></tr>
<tr><td rowspan="3">Intermediate average</td><td colspan="4">45</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td colspan="4">38</td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td colspan="4">48</td></tr>
<tr><td rowspan="9">Moving average</td><td colspan="4">45</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td colspan="4">34</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td colspan="4">36</td><td></td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td colspan="4">39</td><td></td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td colspan="4">38</td><td></td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td colspan="4">41</td><td></td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td colspan="4">43</td><td></td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td colspan="4">51</td><td></td></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td colspan="4">48</td></tr>
</table>

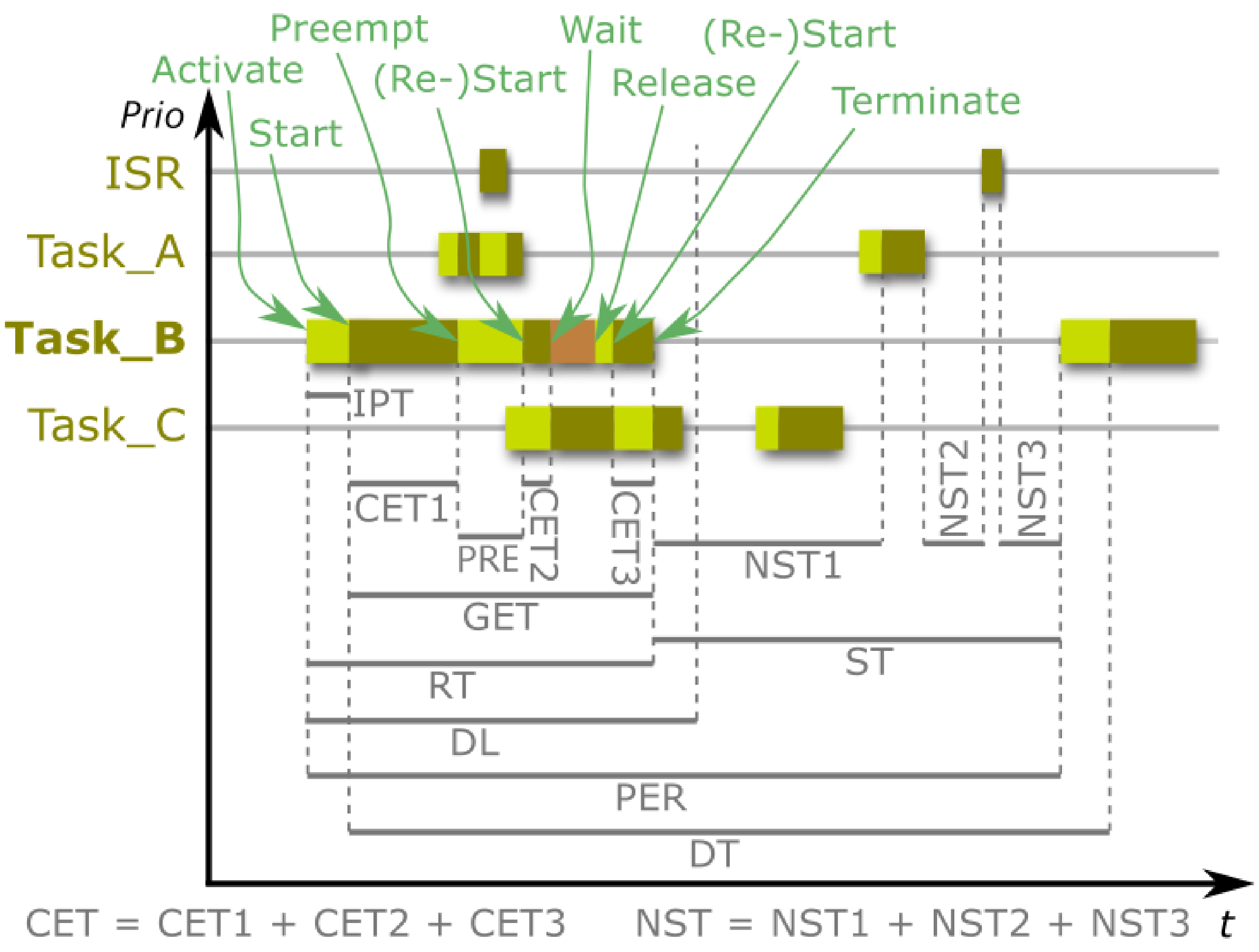
CPU load is always an average of the load L in infinitesimal time frames (idle loop is executed: L=0; TASK, ISR, etc. is executed: L=1).

Previously discussed approach (subsequent observation frames)

Possible alternative?

Too complicated, too costly to calculate efficiently

Introduction of ‘TASK load’ → recap NST



- **IPT (Initial Pending Time)**
  Ready time before task starts
- **CET (Core Execution Time)**
  Time spent in running state, i.e. executing
- **GET (Gross Execution Time)**
  From start to termination (cf. pin toggle)
- **PRE (PREemption Time)**
  Sum of ready times without IPT
- **RT (Response Time)**
  cf. schedulability analysis; DL (DeadLine) = limit for RT
- **Period (PERiod)**
  time difference between two subsequent activations
- **DT (Delta Time)**
  time difference between two subsequent events of the same type; observed period; cf. jitter
- **ST (Slack Time)**
  duration of the ‘gap’

Relevant for **TASK load** calculation

- **NST (Net Slack Time)**
  headroom: time which could be added to CET

Introduction of 'TASK load' → the idea

- The NST of a task nicely indicates the 'head-room' of this task.
- If there is no more headroom (NST = 0), the task is 'overloaded'.
- The unit of the NST is s (or ms or µs or ns etc.). Not a good fit for 'load'.
- Could we use a relative NST? What would it be normalized with?
- For periodical tasks, we could say:

Time 'used up' within period

$$N_T = \frac{PER - NST}{PER} = 1 - \frac{NST}{PER}$$

Period

TASK load definition

- Each pair of subsequent occurrences of $TASK_T$ results in an NST.
- The TASK load N_T of each $TASK_T$ would obviously be the possible or observed *maximum* NST. Cf. $max(U_0 \ldots U_n)$ discussed earlier.
- Obviously, there are as many N_T results as there are TASKs.
- Definition:

  The TASK load N of a system with q periodical TASKs is the maximum N_T of all tasks.

  $$N = max(N_{T0}..N_{Tq})$$

- What about non-periodic TASKs and ISRs? I don't know.
  Need to think about it…

Agenda

- Introduction
- CPU load
- TASK load (new idea, proposal)
- **Summary**

Summary

- People tend to think it is obvious what ‘CPU load’ means.
- Different definitions/assumptions/understandings lead to different results and, in the end, to problems in real projects.
- Suggestion: discuss and establish a *common understanding and definition*. Maybe

Load = max(U, N)

- Whatever load definition you pick, *thinking* about this topic and deriving actions will very likely **improve the quality of embedded software!**

AUTOSAR

Thank You!

Peter Gliwa
Dipl.-Ing. (BA)

Geschäftsführer (CEO)

GLIWA

GLIWA GmbH embedded systems
Pollinger Str. 1
82362 Weilheim i.OB.
Germany

fon +49 - 881 - 13 85 22 - 10
fax +49 - 881 - 13 85 22 - 99
mobile +49 - 177 - 2 57 86 72

peter.gliwa@gliwa.com
www.gliwa.com

BOSCH Continental Ford gm STELLANTIS TOYOTA VOLKSWAGEN GROUP