Monitoring of Workload Arrival Functions for Mixed-Criticality Systems

Moritz Neukirchner, Philip Axer, Tobias Michaels, Rolf Ernst

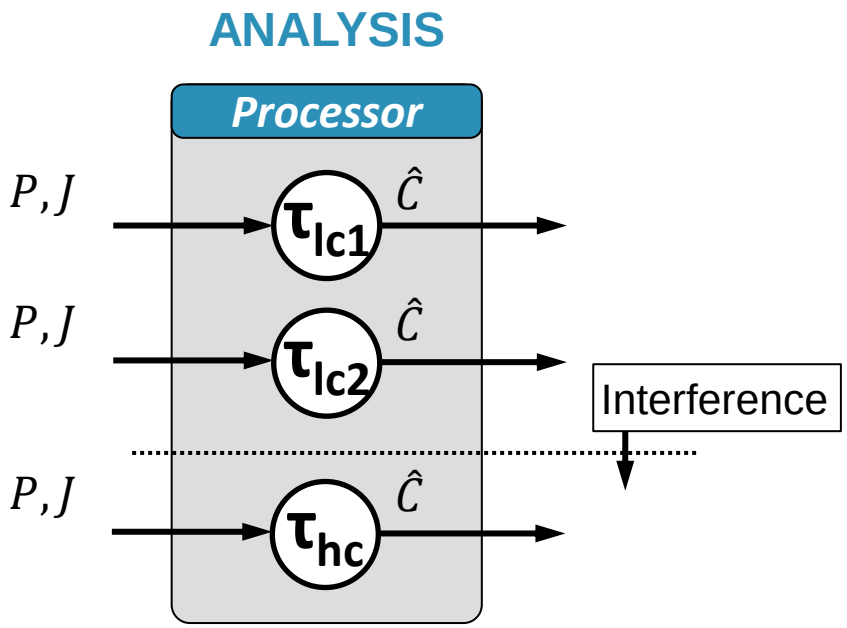
Requirement of Safety Standard IEC61508

IEC61508:
*"For a […] **system** that implements […] functions of **different safety** […] **levels**, **unless** it can be shown there is **sufficient independence** […], the requirements applicable to the **highest relevant safety** integrity **level shall apply** […]."*

What is
sufficient independence
?

IEC61508:
*"the **probability of** a **dependent failure** between the non-safety related and safety-related parts is **sufficiently low**"*

Mixed-Criticality and Sufficient Independence



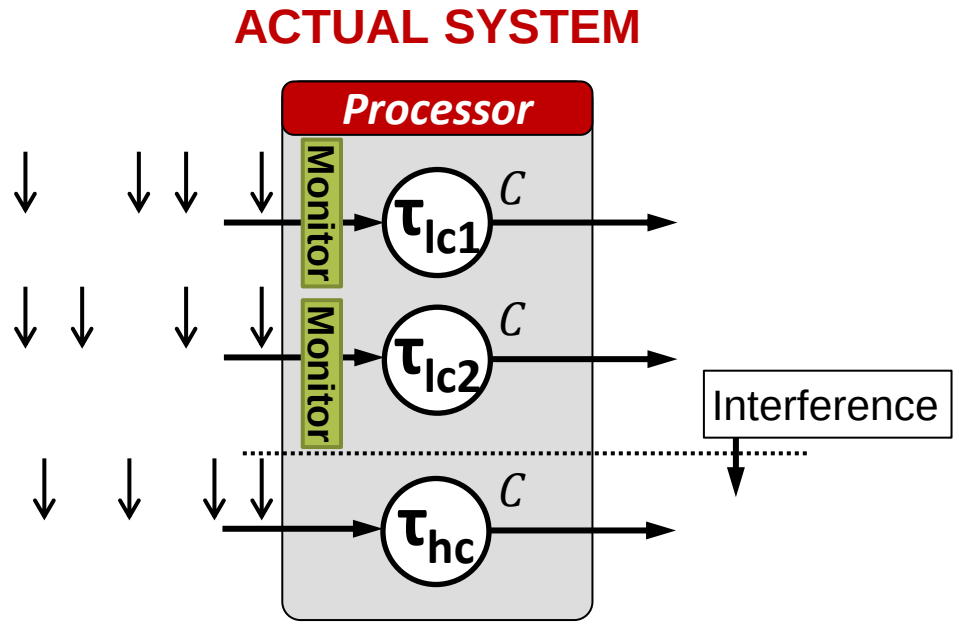
- Tasks of **different safety-criticality**
- Specification of activation pattern and WCET
- Timing analysis yields **maximum interference** from low to high criticality
- For certification of τ_{hc} only the **cumulative interference of higher priorities** is relevant
- Interference from untrusted (i.e. low criticality) tasks must not exceed analysis bounds

  → **Enforcement**

Mixed-Criticality and Sufficient Independence

- Actual execution times
  - Enforcement through **execution time monitors**
- Traces
  - Enforcement through **activation pattern monitors** (e.g. [Wrege96], [Lampka11], [Neukirchner12])
- **Enforced Interference**

$$I_{lc}(\Delta t) = \hat{C}_{lc1} * \left\lceil \frac{\Delta t + J_{lc1}}{P_{lc1}} \right\rceil + \hat{C}_{lc2} * \left\lceil \frac{\Delta t + J_{lc2}}{P_{lc2}} \right\rceil$$

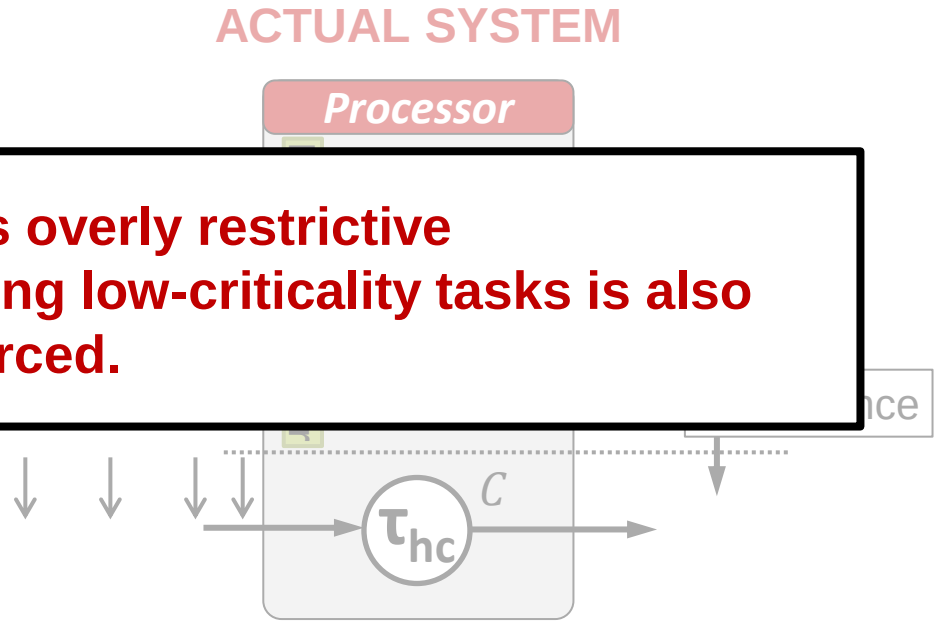


Mixed-Criticality and Sufficient Independence

- Actual execution times
  - Enforcement through **execution time monitors**
- **Traces**
- **Enforced Interference**

$$I_{lc}(\Delta t) = \hat{C}_{lc1} * \left\lceil \frac{\Delta t + J_{lc1}}{P_{lc1}} \right\rceil + \hat{C}_{lc2} * \left\lceil \frac{\Delta t + J_{lc2}}{P_{lc2}} \right\rceil$$

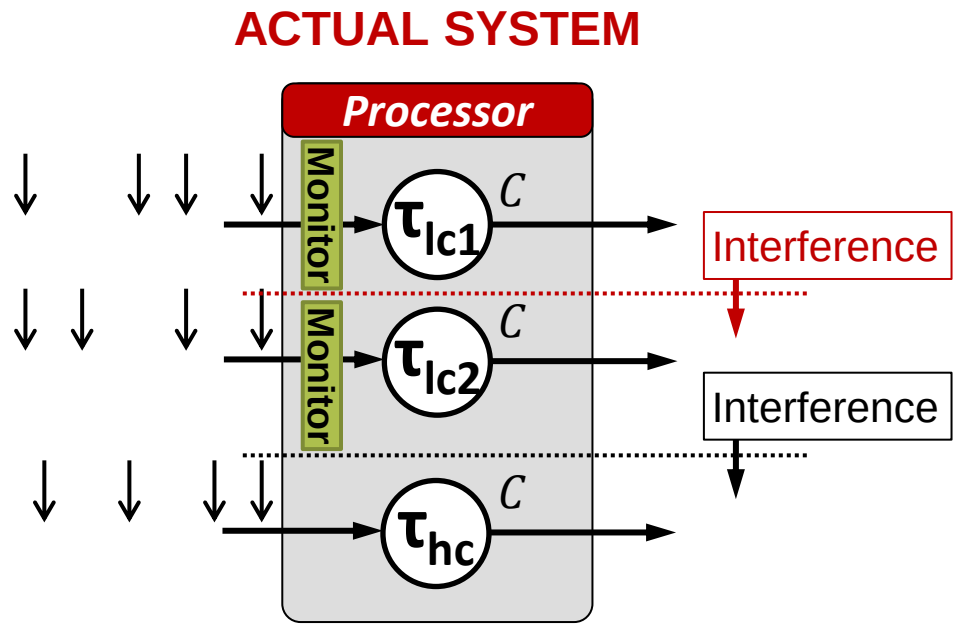
This monitoring is overly restrictive because the interference among low-criticality tasks is also enforced.



Mixed-Criticality and Sufficient Independence

- Actual execution times
  - Enforcement through **execution time monitors**
- **Traces**
  - Enforcement through **activation pattern monitors** (e.g. [Wrege96], [Lampka11], [Neukirchner12])
- **Enforced Interference**

$$I_{lc}(\Delta t) = \hat{C}_{lc1} * \left\lceil \frac{\Delta t + J_{lc1}}{P_{lc1}} \right\rceil + \hat{C}_{lc2} * \left\lceil \frac{\Delta t + J_{lc2}}{P_{lc2}} \right\rceil$$



- **Over-enforces** if low criticality tasks (typically) do **not** experience **worst-case simultaneously** (e.g. uncorrelated sporadic tasks)

Outline

- Modelling workload of arbitrarily activated tasks
- Monitoring of workload-arrival functions
  - Checking traces
  - Achieving constant runtime overhead
- Evaluation

Outline

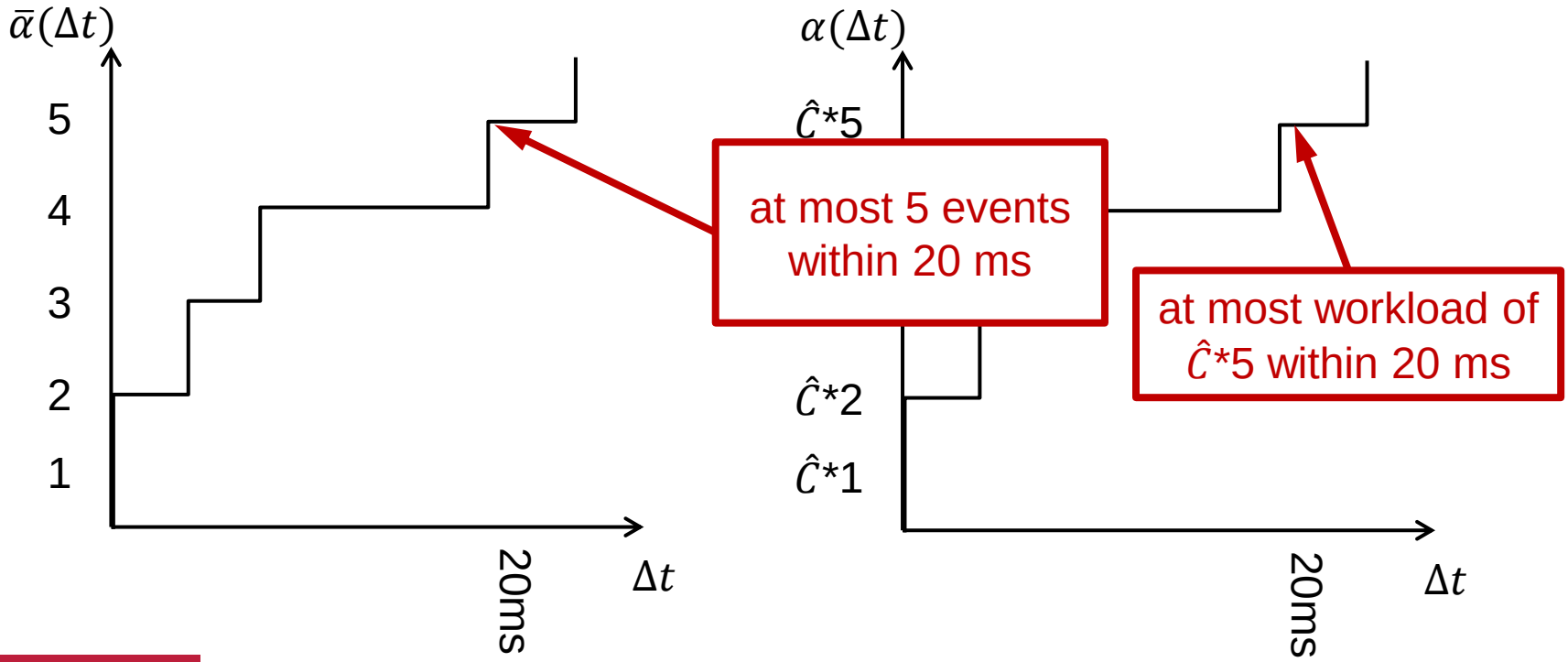
- Modelling workload of arbitrarily activated tasks

- Monitoring of workload-arrival functions
  - Checking traces
  - Achieving constant runtime overhead

- Evaluation

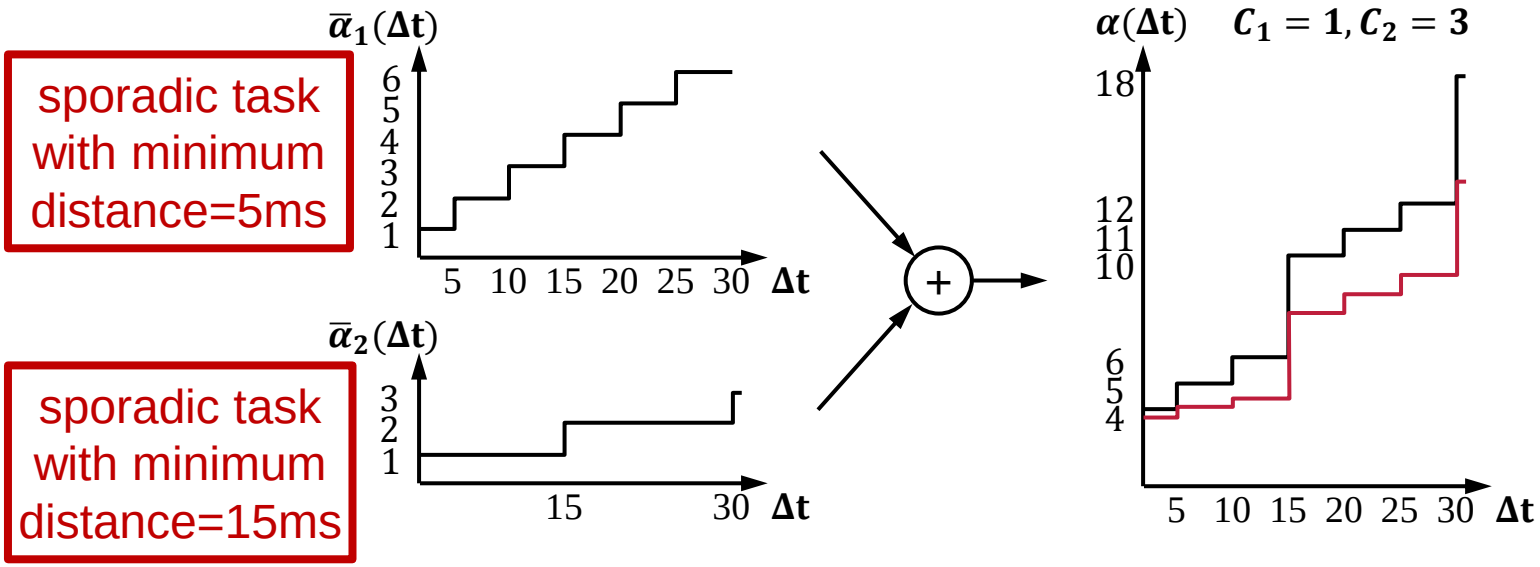
Modelling Arbitrary Activation Patterns

- **Event-arrival functions** specify the maximum number of events that may occur in a time-interval of size **Δt**
- **Workload-arrival functions (WAF)** specify the maximum workload that may be requested in a time-interval of size **Δt**



Workload-arrival functions for multiple tasks

- The **maximum interference** a task may have on lower priorities in a time interval Δt is given through its **WAF**
- **Sum of WAFs** of group of tasks is the maximum interference through the **group**
- Can encode **interference** from **correlated activations** (group WAF smaller than sum of individual WAFs)



Outline

- Modelling workload of arbitrarily activated tasks

- Monitoring of workload-arrival functions
  - Checking traces
  - Achieving constant runtime overhead

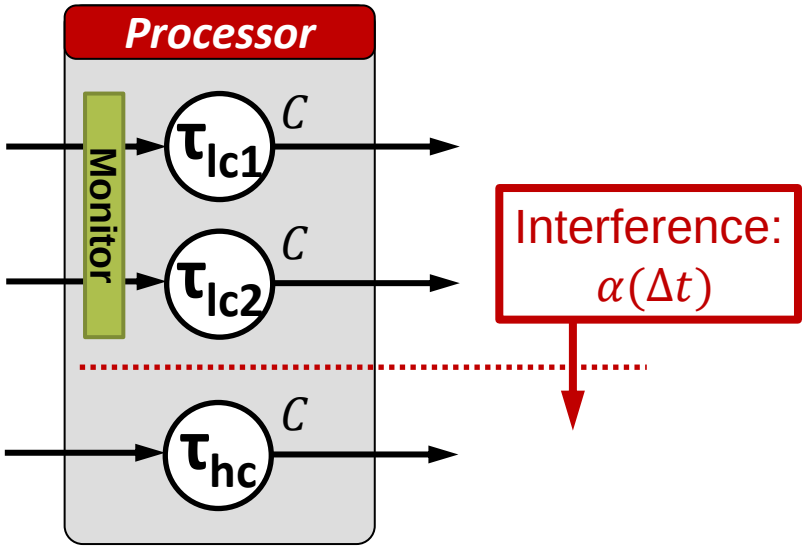
- Evaluation

Monitoring Workload-arrival functions for multiple tasks

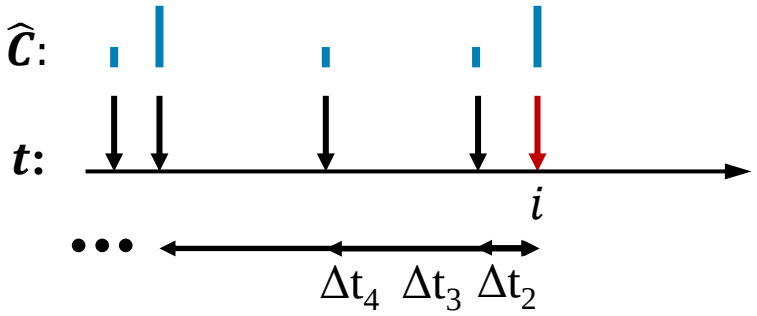
- **One monitor per** group of tasks of the same **criticality level**
- Monitor enforces workload-arrival function for group
- Monitored task may **exceed own budget at cost of another** in the group

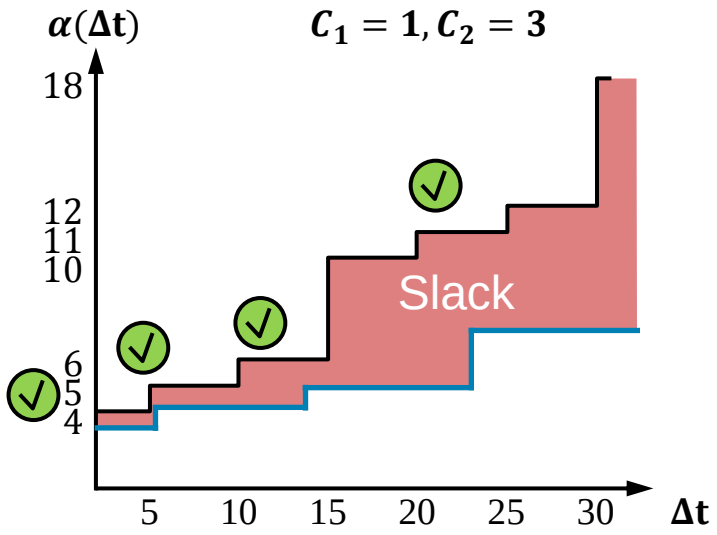
  → **Relevant for sporadic tasks** that rarely reach worst-case

- **Enforced Interference** on lower priorities:
  $I_{lc}(\Delta t) = \alpha(\Delta t)$



Satisfaction of Workload-Arrival Functions





- **Satisfaction check** for new event:

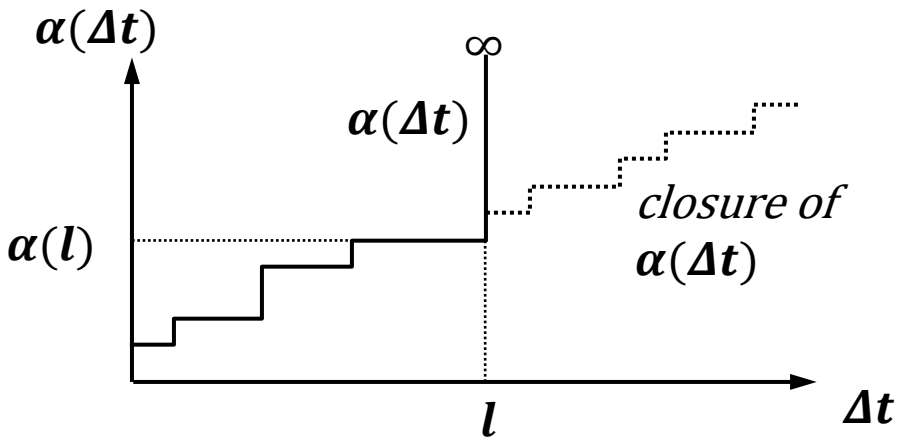
$$\forall j \leq i: \sum_{l=0}^{j} \sigma^C(l) \leq \alpha(\sigma^t(i) - \sigma^t(i-l))$$

- **Complexity depends** on **trace length** i

Achieving Constant Runtime Overhead

- **Limited arrival function** can be checked with **constant overhead** because everything beyond $\boldsymbol{l}$ is trivially satisfied
- Network Calculus [LeBoudec01]:

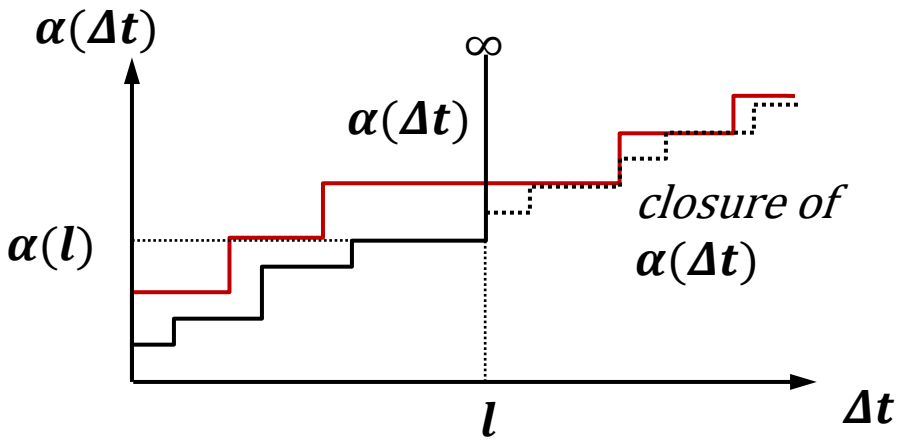
  *„It is **equivalent** whether a trace is **constrained** through any wide-sense increasing **arrival function or** through the corresponding **sub-additive closure**.“*
- Sub-additive closure is the largest sub-additive function smaller than a given arrival function



Achieving Constant Runtime Overhead

- Any **sub-additive closure** can be checked at **constant time** with complexity $\boldsymbol{O(l)}$
- **Arbitrary WAF** can be **conservatively monitored** with sub-additive closure smaller than the WAF

- **Memory complexity** is **bounded through discretization** of **workload** and monitoring according to inverse WAF



Outline

- Modelling workload of arbitrarily activated tasks
- Monitoring of workload-arrival functions
  - Checking traces
  - Achieving constant runtime overhead
- Evaluation

Evaluation

- **Implementation** in MicroC/OS-II on **Cortex-M3**
- Comparison: **Individual vs. Group Monitoring**

Evaluation of Slack Reclamation:

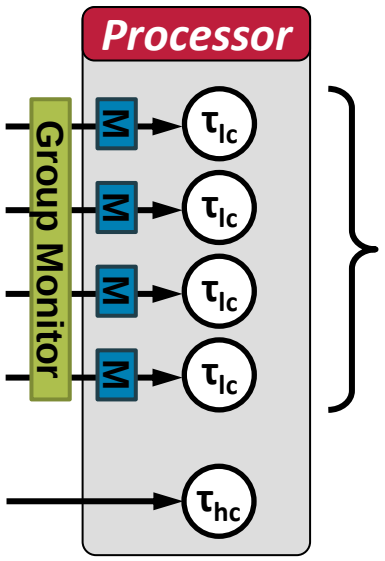
- Specified task set with sporadic activation
- Specified WAF/ Individual event-arrival functions

- **Number of violations** for individual vs. group monitoring

Evaluation of Slack Reclamation

- **Individual** vs. **Group Monitoring** of sporadic tasks
- Metric:
  - **Relative number of violations:**

$$violations\ group / violations\ individual$$



- 2-16 sporadic tasks
- 10%-80% utilization

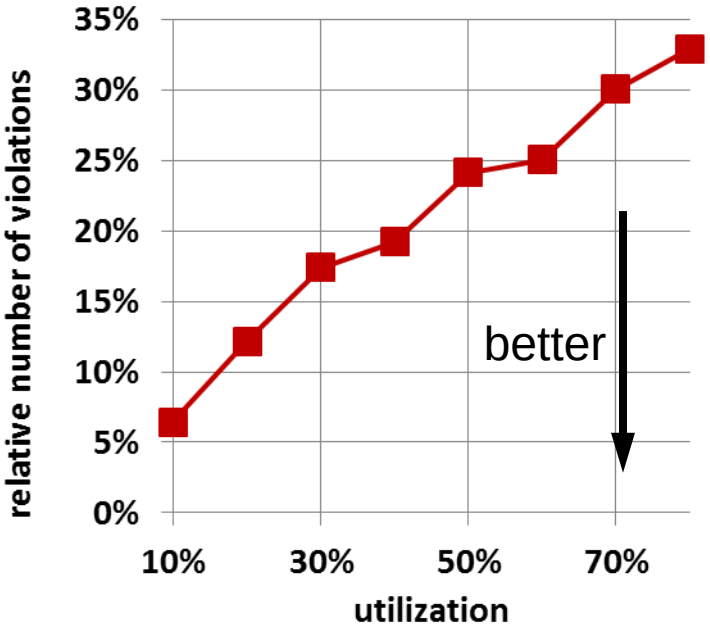
Investigated parameters:

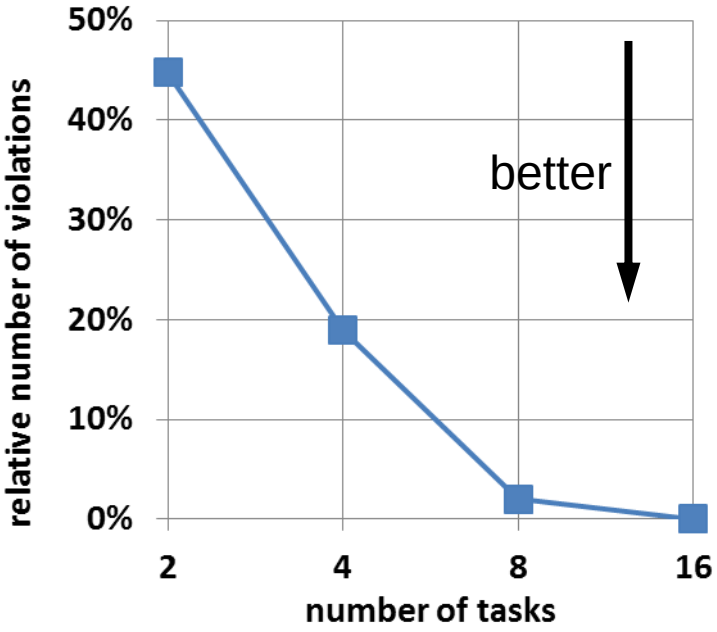
- total **number of tasks**
- **utilization** through sporadic tasks

Testcase Generation:

- **Random execution time** in [1ms,5ms]
- **Sporadic act.** randomly with **mean inter-arrival** rate (uniform distribution over [0, 2*d_{mean}])
- **Group WAF equal** to sum of **individual WAFs** → no correlation

Evaluation of Slack Reclamation





- **Reduction** of number of **violations** by **3x – 15x** over different util.
  - low sporadic load → correlation less probable
- **Reduction** of number of **violations** of at least **2x** over different task num.
  - more tasks → correlation less probable

Evaluation of Activation Correlations

Evaluation of Activation Correlation:

- What is the benefit if task activations are not independent?

Correlation:

An activation of one task must have a minimum distance to that of another

For a **given trace with activation correlation,**
what is the **tightest monitor configuration** that triggers **no exception?**

Evaluation of Correlated Sporadic Activations

- Specified task set with sporadic activation
- Minimum distance between activations in the group

- **Enforced Interference** of **Individual** vs. **Group** monitoring
- Record tightest configuration from trace
- **Interference** permitted by **ind. monitors**:

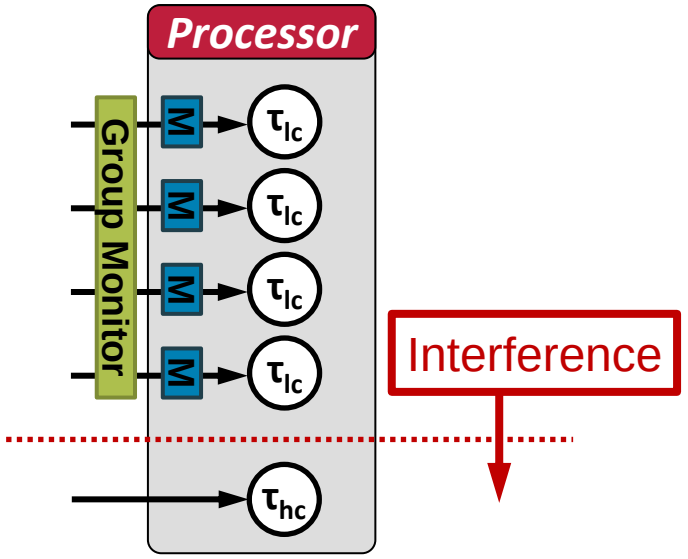
$$\mathrm{I}_{\mathrm{ind.}}(\Delta t) = \sum_j \bar{\alpha}_j(\Delta t) * C_j$$

- **Interference** permitted by **group monitors**:

$$\mathrm{I}_{\mathrm{group}}(\Delta t) = \alpha(\Delta t)$$

- Metric: **mean relative interference**

$$mean_{\Delta t}\left(\frac{I_{group}(\Delta t)}{I_{ind.}(\Delta t)}\right)$$



Evaluation of Correlated Sporadic Activations

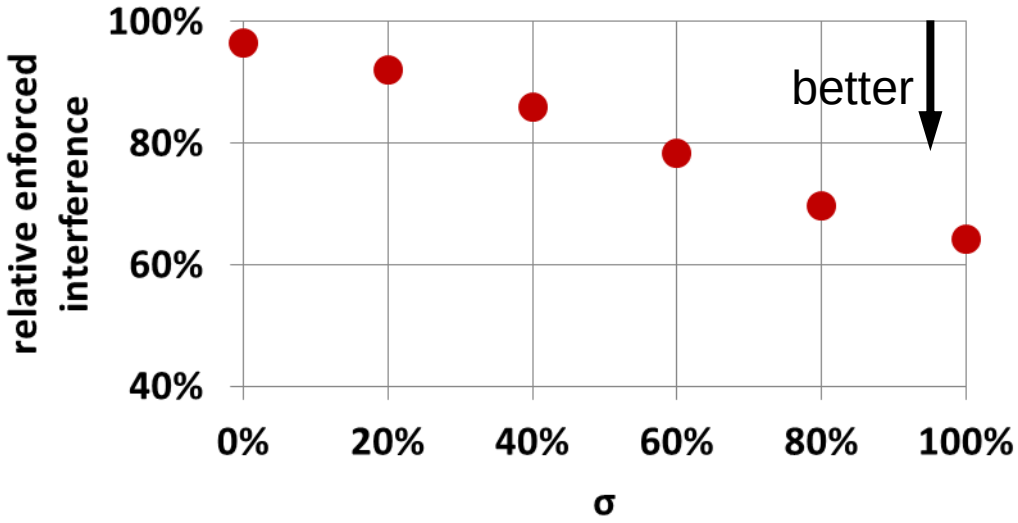
Testcase Generation:

- 4 sporadic tasks
- Tasks **activated at minimum inter-arrival rate**
- Additionally **minimum distance** between **activations in group**

$d_{min} = \sigma * \hat{C}_{max}$ with **correlation parameter** $\sigma \in [0,1]$

- $\sigma = 0 \rightarrow$ no correlation (critical instant possible)
- $\sigma > 0 \rightarrow$ no two activations at the same time

Evaluation of Correlated Sporadic Activations



- **Without correlation** ($\sigma = 0$) **enforced interference** of individual and group monitoring is **identical**
- **With correlation** ($\sigma > 0$) **enforced interference** through group monitoring **significantly lower**

Conclusion

- **Current monitoring** schemes enforce interference/activation pattern **per task** rather than per criticality level
- This **prevents slack reclamation** per class and over-isolates

We have presented

- **Monitoring groups** of tasks according to **workload-arrival functions**
- **Constant overhead** monitoring
- Allows to **reclaim slack within** a **criticality group**
- Allows to **encode** and **enforce** the effects of **correlations** among sporadic task activations

Thank you for your attention.