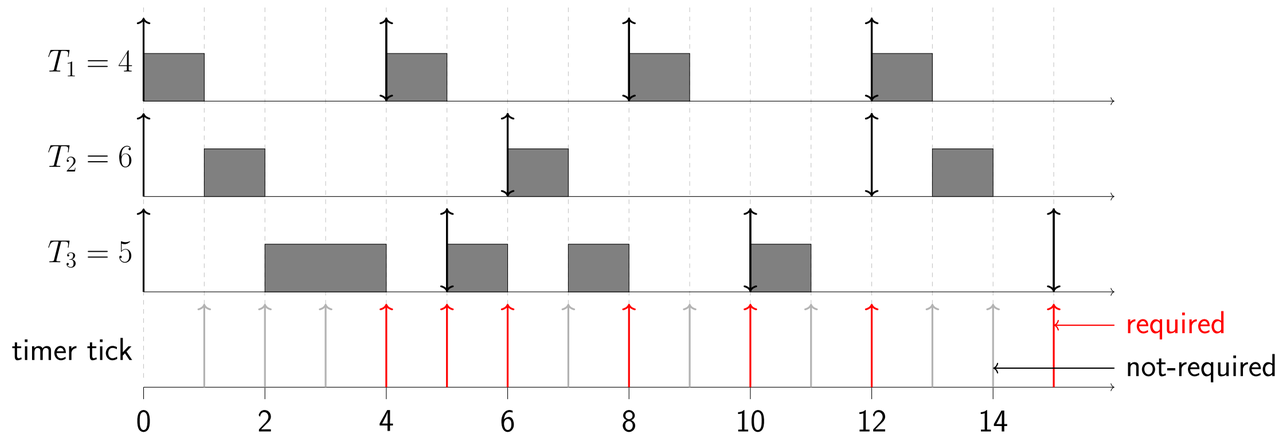
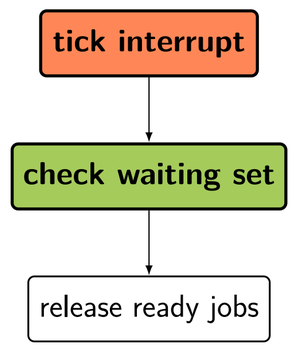
LazyTick: Lazy and Efficient Management of Job Release in Real-Time Operating Systems

Kay Heider[†], Christian Hakert[†], Kuan-Hsun Chen[⋆], Jian-Jia Chen[†‡]

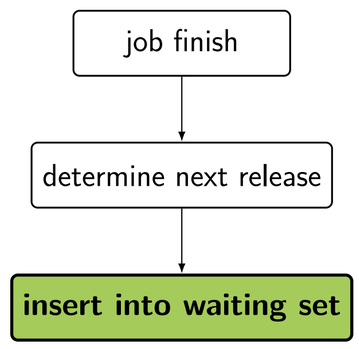
[†]TU Dortmund University, Germany [⋆]University of Twente, the Netherlands [‡]Lamarr Institute, Germany



Tick Interrupt



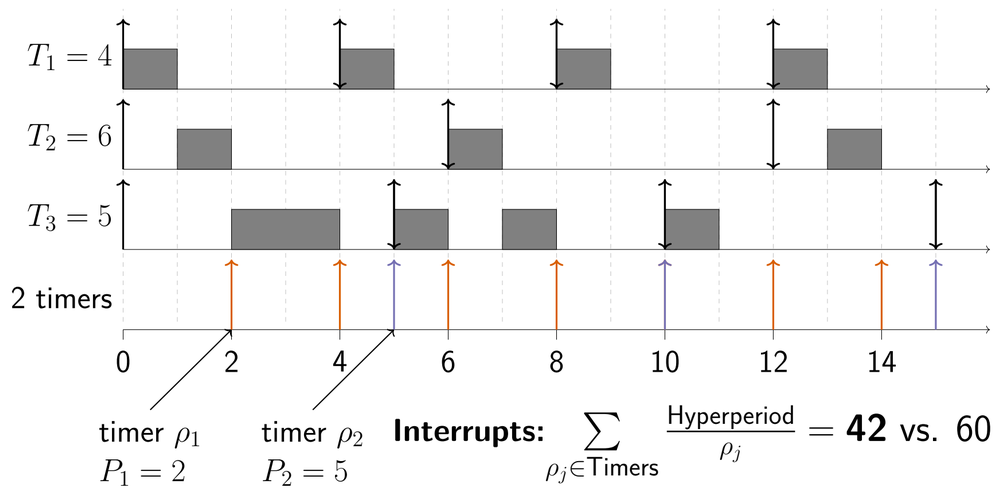
Task Delay



Overhead = #ticks · tick complexity + #delays · delay complexity

Reducing Tick Interrupts

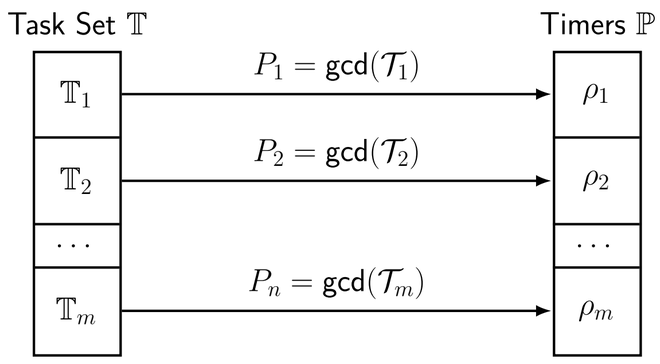
Idea: Employ multiple periodic timers instead of only one



- Multiple timers can reduce the number of not-required tick interrupts
- Smart assignment of tasks to timers reduces overhead

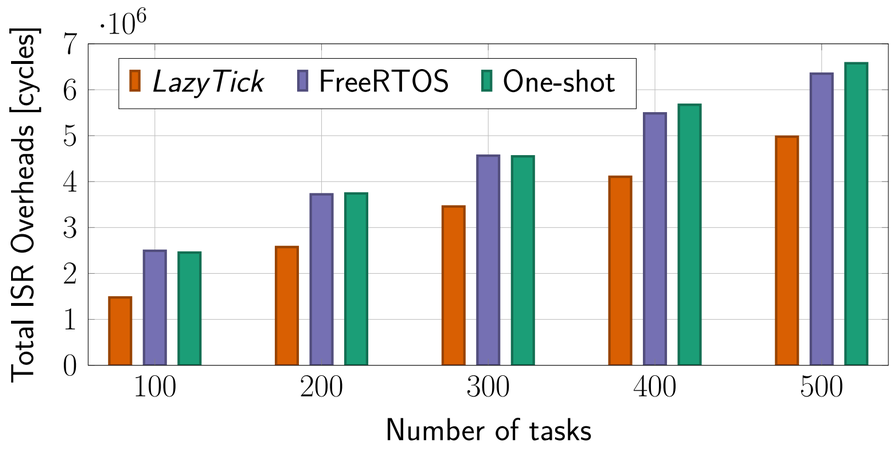
Multi-Timer Task Set Partition

Method: Partition task set, assign each a timer, exploit the gcd



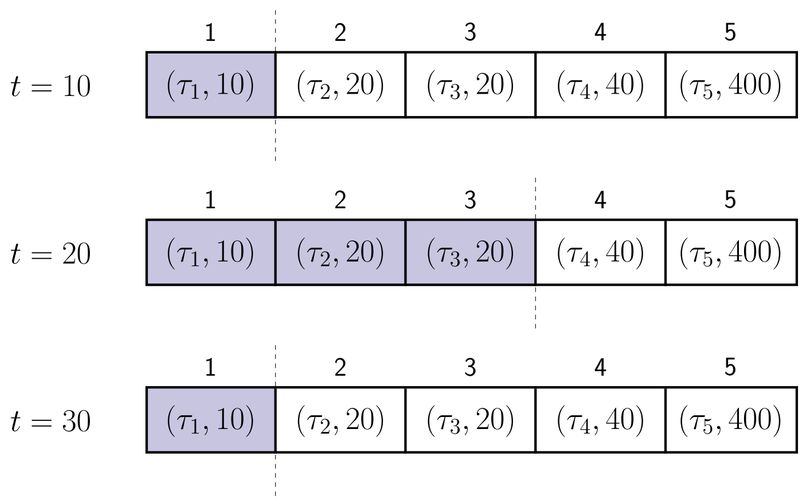
- Partition the task set according to the task periods, maximizing the gcd
- Assign each partition a separate timer with tick period configured to the gcd

Evaluation



Harmonic Task Sets

Method: Use a static array as waiting set

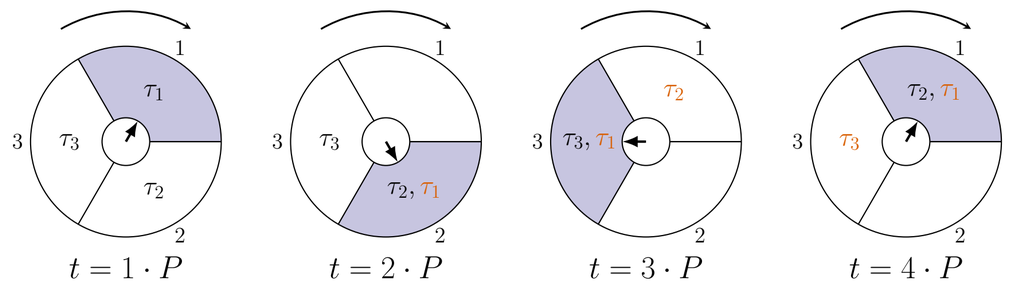


- Exploit fixed release order of harmonic tasks
- Each tick releases at least one job

Non-Harmonic Task Sets

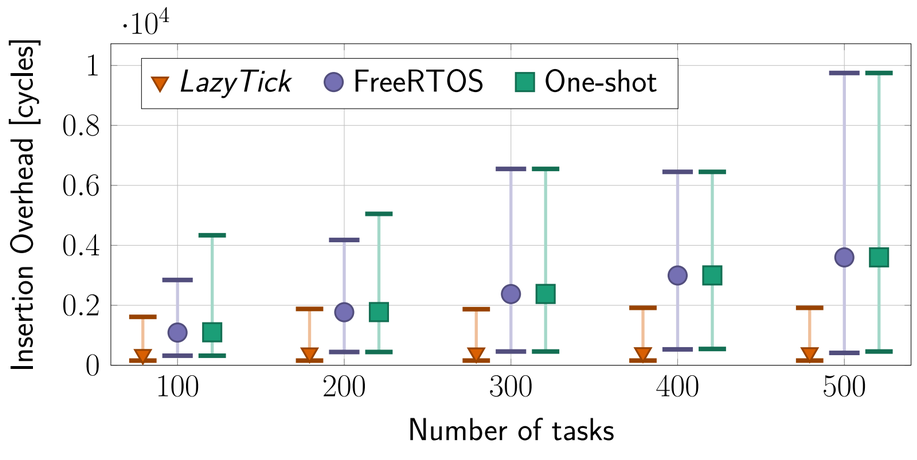
Method: Use a ring buffer as waiting set, advance by timer period

Tasks with $T_1 = 2, T_2 = 4, T_3 = 6$ Timer with $P = \gcd(2, 4, 6) = 2$



- A ring buffer slot holds an unsorted list of tasks, that release a job at the same tick
- After a job has finished, the task is inserted at the next slot according to its period

Evaluation



(Kay Heider) (Christian Hakert) (Kuan-Hsun Chen) (Jian-Jia Chen)

Artifacts Available V1.1 — Results Reproduced V1.1

(GitHub)

EMBEDDED SYSTEMS WEEK