A Framework for the Optimization of the WCET of Programs on Multi-Core Processors

Max John, Michael Jacobs
Saarland University, Saarbrücken

SAARLAND UNIVERSITY
COMPUTER SCIENCE

Introduction

Scenario

- Embedded systems
- Timing-critical applications
  - Strict deadlines, e.g. in automotive applications
  - Need of tight WCET bounds

Implementation (HW & SW)

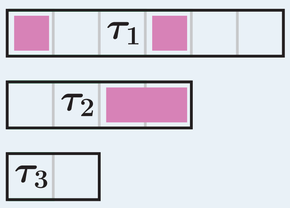
- Multi-core processor with shared bus
  - Exploit task parallelism
  - However: cores interfere
- Usage of a TDMA bus
  - Cores no longer interfere
- Use static task scheduling

Challenge

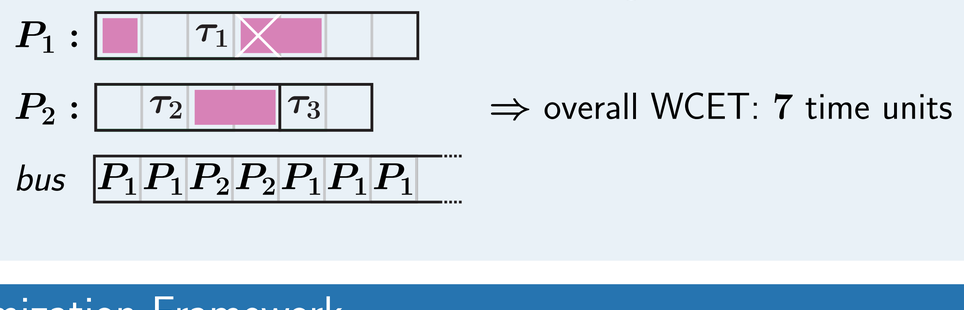
- Find static system schedule and bus schedule
- However: Optimal schedule hard to obtain
- Approximation framework needed

System Model

- Tasks
  - Single execution behavior
  - Determined by length and bus accesses



- System schedule
  - Assigns tasks to processor cores
  - Determines the tasks' execution order
- Bus schedule
  - If a bus request is denied
    $\Rightarrow$ Processor core blocked until access is granted



$\Rightarrow$ overall WCET: **7** time units

Optimization Framework

- Goal: Reduce the overall WCET
- By integrated construction of
  - System schedule
  - Bus schedule

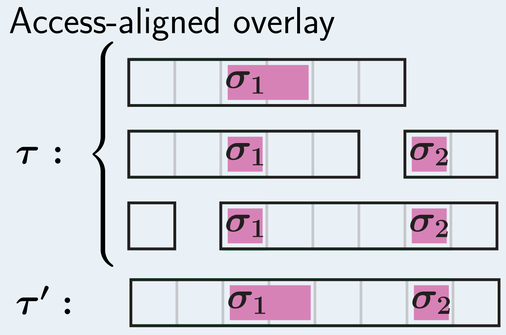
Data: *tasks*: set of tasks,
n: number of processor cores,
th: task selection heuristic,
bh: bus schedule heuristic
(*sys*, *bus*) $\leftarrow$ empty schedules for n processor cores;
while *tasks* $\neq \emptyset$ **do**
  task $\leftarrow$ *th*(*tasks*, *sys*, *bus*);
  p_idle $\leftarrow$ find first idle core in (*sys*, *bus*);
  sys $\leftarrow$ add *task* in *sys* to *p_idle*;
  bus $\leftarrow$ *bh*(*sys*, *bus*, "*partial*");
  tasks $\leftarrow$ *tasks* $\setminus$ {*task*};
end
bus $\leftarrow$ *bh*(*sys*, *bus*, "*complete*");
return (*sys*, *bus*);

- Modularity: plug in heuristics
  - Task selection heuristic (*th*)
  - Bus schedule heuristic (*bh*)

Real-World Programs in our System Model

- Real-world programs have multiple behaviors
- Soundly over-approximate them by a single one
  - Simple overlay

  - Access-aligned overlay

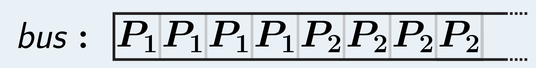


Steps towards more General Systems

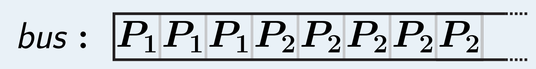
- Task dependencies
  - Constraint task selection accordingly
  - Side effect: sometimes none of the remaining tasks selectable
  - Solution: return dummy task τ_d in those cases: $\boxed{\tau_d}$
- Task priorities
  - Constraint task selection and bus schedule heuristic accordingly
- Restrict the granularity of the bus schedule
  - Many systems have a bus processor ratio K, i.e.

$$\forall n \in \mathbb{N}.\ n \not\equiv 0 \mod K \Rightarrow bus(n) = bus(n-1)$$

  - Thus some bus schedules are no longer legal



$\Rightarrow$ Legal bus schedule for $K = 4$



$\Rightarrow$ Illegal bus schedule for $K = 4$

  - Constraint bus heuristic to produce legal schedules only

Future Work

- Develop access-aware task selection heuristics
  - Only non-access-aware heuristics exist
- Experiments
  - Extract traces from real-world programs
  - Evaluate effectiveness of heuristics
  - Soundly combine several traces to one
    $\rightarrow$ Determine degree of over-approximation

Acknowledgement

http://www.uni-saarland.de

AVACS http://www.avacs.org