

Parallelization of a Diesel Engine Management System

Sebastian Kehr
Dr. Bert Böddeker

DENSO AUTOMOTIVE Deutschland GmbH

parMERASA Dissemination Event
23.09.2014 – Barcelona

- DENSO
- Motivation and Objectives
- Automotive Software
- Parallelization
- Conclusion



Established	December 16, 1949
--------------------	-------------------

Capital	187.4 billion yen (EUR 1.7 billion)
----------------	-------------------------------------

Net sales	
Consolidated basis	3,580.9 billion yen (EUR 32.6 billion)

Net income	
Consolidated basis	296.0 billion yen (EUR 2.7 billion)

Employees	
Consolidated basis	132,276

Consolidated subsidiaries	183
(Japan 62, North America 28, Europe 34, Asia/Oceania 53, Others 6)	

Affiliates under the equity method	32
(Japan 13, North America 4, Europe 2, Asia/Oceania 11, Others 2)	

Notes:

EU€ amounts have been translated, for convenience only, at the rate of 110 yen = EUR € 1,

/ as of March 31, 2013

Body Electronics

- Meters
- Head up Displays
- Windshield Wipers
- Windshield Washers
- Horns
- Flashers



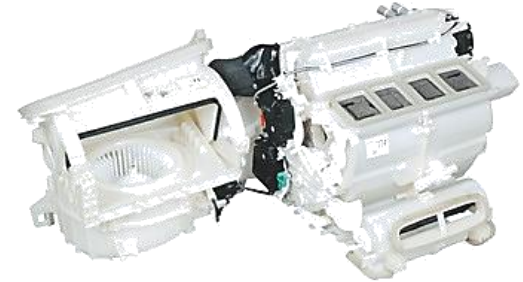
Powertrain

- Gasoline Engine Management Systems
- Diesel Engine Fuel Injection Systems
- Hybrid and Electric Vehicle Components
- Engine Electrical Equipment
- Engine Cooling Systems



Thermal

- Automatic Air Conditioner
- Compressors
- Heat Exchangers
- Cool & Hot Boxes
- Air Purifiers



Driving Control and Safety

- ABS, TRC, VSC Controls
- Door Lock Controls
- SRS Air Bags
- Cruise Controls
- Radar / Camera Systems
- Corner Clearance Warning Systems



Information and Communication

- IVI-Systems
- Electronic Toll Collection (ETC) Systems
- Data Communication Module



Location: DENSO AUTOMOTIVE Deutschland GmbH
Freisinger Str. 21
85386 Eching
Germany

24 km distance to Munich

Branch offices: Aachen, Frankfurt, Köln, Stuttgart, Wolfsburg

Main Business: Development of A/C & Thermal systems.
Sales of air conditioners, heaters, electric/
electronic products and powertrain
components. Extensive testing facilities.

Testing Facilities: 1 Vehicle climatic chamber
2 Noise chambers
2 Climatic chambers
4 Temperature control chambers
1 Component noise chamber

Site Area: 7,800 m²

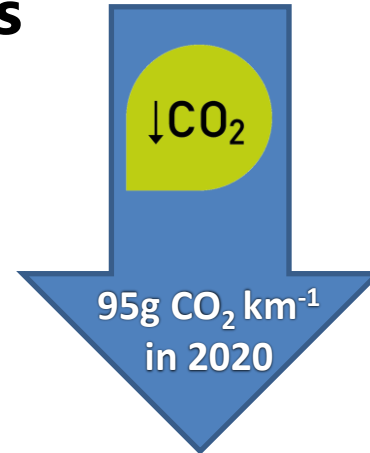
Founded: 1984

Employees: 500



Automotive Market Challenges

- Reduce energy consumption
- Advanced driver assistance
- Automated driving



› *Multi-core ECU seen as platform for future applications*

Objectives

- Develop a WCET-aware parallelization approach
- Understand potential for parallelism in automotive software
- Reduce runtime of *worst-case scenario* (to reduce clock speed or energy consumption)

AUTOSAR – Standardised software architecture

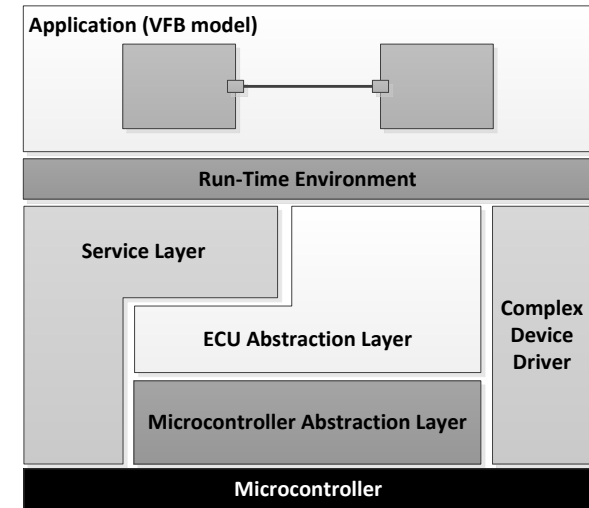
- Layered basic software with uniform structure and interfaces
- Component-based application structure (VFB model)
- Run Time Environment (RTE) separates components

Runnable – Elementary code pieces

- Implement behaviour of Software-Component
- Communicate through the RTE

Task – Unit of Scheduling

- Group Runnables with the same release period or interrupt
- Execution with fixed period or sporadic after an (external) event
- Fixed priority preemptive scheduling (Typical: rate-monotonic)



Case study: Diesel EMS

- 4-cylinder diesel engine
- About 30 sensors (temperature, ...)
- About 25 actuators (injection, ...)
- CAN interfaces

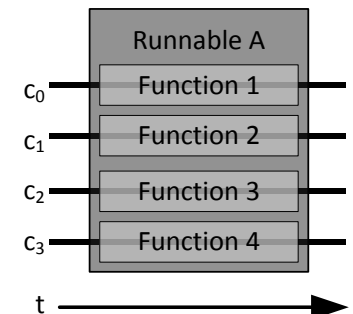
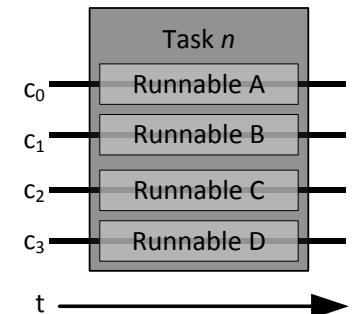
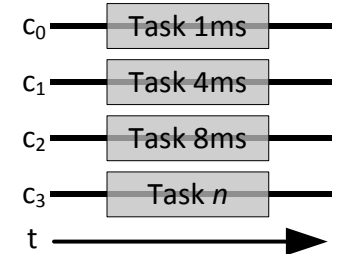
Software properties

- Typical automotive application
- Potentially complex enough to have benefits from parallelization
 - About 3000 source file with 90 000 lines of code
 - 12 tasks and about ~800 Runnables
- Two drifting timescales
 - A sensor at the camshaft triggers interrupt every 30° of a revolution
 - Status of a set of sensors is actively sampled periodically (polling)

Parallelism in automotive software

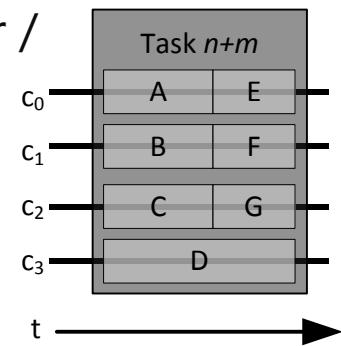
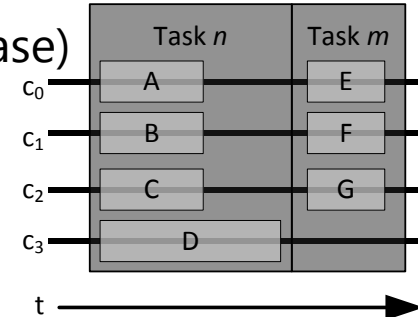
- *Inter-task* – Parallel execution of tasks
 - Parallel execution of tasks supported by AUTOSAR
 - Preserves task bodies, runnables
- *Intra-task* – Parallel execution of Runnables
 - Distribute Runnables over cores
 - Currently, not supported by AUTOSAR
 - Preserves task schedule, runnables
- *Intra-Runnable* – Parallel execution of functions
 - Currently, not supported by AUTOSAR
 - Preserves task schedule

› *Which level provides the best performance?*



Intra-task – Mapping Tool [1] (in collaboration with BSC)

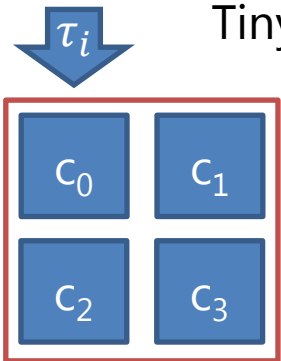
- Definition of a static schedule for every task
 - Runnables of a task are mapped to cores (worst-fit decrease)
 - Core allocation is bound by WCET
 - TDG defines precedence constraints between Runnables
- Advantages
 - Task scheduling remains unchanged
 - No locks for Client-Server calls required (precedence constraints guarantee correct execution order)
 - Low overhead (data consistency provided by execution order / synchronization with advance function)
- Improvement: **supertask**
 - Combine Runnables of tasks scheduled to the same point in time → task interleaving



[1] Panic, Milos; Kehr, Sebastian; Quiñones, Eduardo; Böddeker, Bert; Abella, Jaume and Cazorla, Francisco: RunPar: An Allocation Algorithm for Automotive Applications Exploiting Runnable Parallelism in Multicores . Accepted for the International Conference on Hardware/Software Codesign and System Synthesis (CODES+ISSS), New Delhi, India, October 2014

Results: Mapping Tool

- Setup
 - 4 cores (parMERASA MC / single cluster)
 - 256KB data cache
 - Instruction scratchpad
 - Assume period 1.25 for CrBas
 - WCET estimated with OTAWA
- Static schedule for each task
 - Synchronization of Runnables with advance-instruction from TinyAUTOSAR



WCET speed-up

Task	Sequential	Parallel	Speed-up
τ_1	104260	95845	1.09
τ_4	371453	210032	1.77
τ_5	12426	12426	1.00
τ_8	249165	74842	3.33
τ_{16}	840580	422562	1.99
τ_{20}	65412	32749	2.00
τ_{32}	612863	322600	1.90
τ_{64}	132771	84462	1.57
τ_{96}	102593	82300	1.25
τ_{128}	391206	342665	1.14
τ_{1024}	469303	379605	1.24
τ_{crBas}	833225	437248	1.91

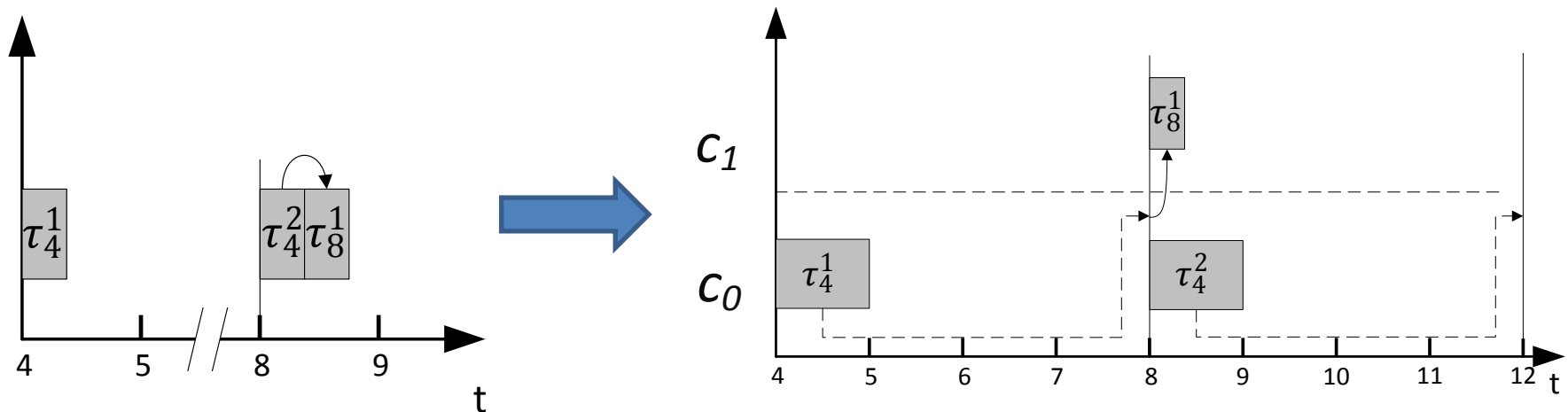
Supertask	Sequential	Parallel	Speed-up
$\tau_1 + \tau_4 + \tau_5 + \tau_8 + \tau_{20} + \tau_{16} + \tau_{32} + \tau_{64} + \tau_{96} + \tau_{128} + \tau_{1024}$	3352032	2000071	1.68
$\tau_1 + \tau_4 + \tau_8 + \tau_{16} + \tau_{32} + \tau_{96}$	2280914	857946	2.66

Inter-task – Timed Implicit Communication

- Decoupled tasks with asynchronous communication

Implementation

- Data elements are redirected to a buffer + publication timestamp
- Publication at end of producer period (e.g. $t = 8$ for τ_4^1)
- Data flow is determined based on the single-core configuration

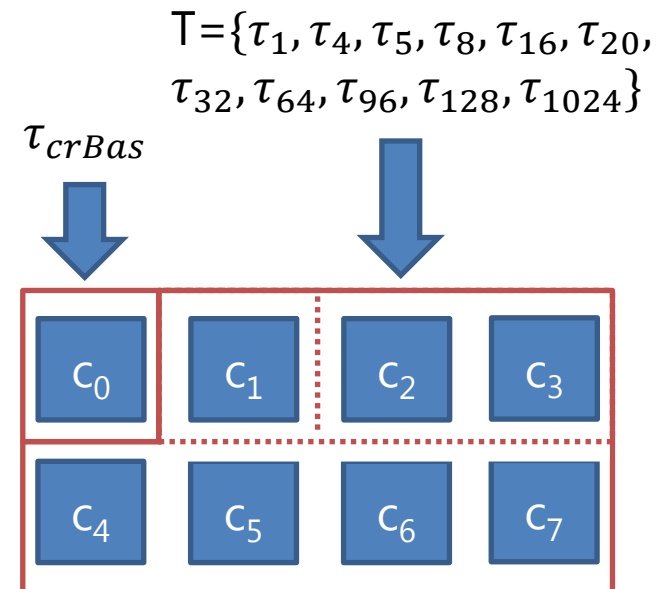


› *Scheduling becomes independent from the data flow!*

Results: Inter-task p. w/ Timed Implicit Communication

- Setup: Schedule for worst-case scenario
 - Perfect buffer mechanism is assumed
 - 256KB data cache / instruction scratchpad
 - 1 cluster / crank-angle task fix on core 0
 - Task WCET estimated with OTAWA
 - Scheduling with parMERASA mapping tool
 - Client-Server calls encapsulated in ticket locks
- WCET speed-up

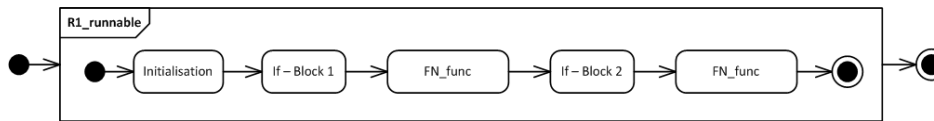
Cores	WC scenario
2	0.59
4	2.56
8	4.46



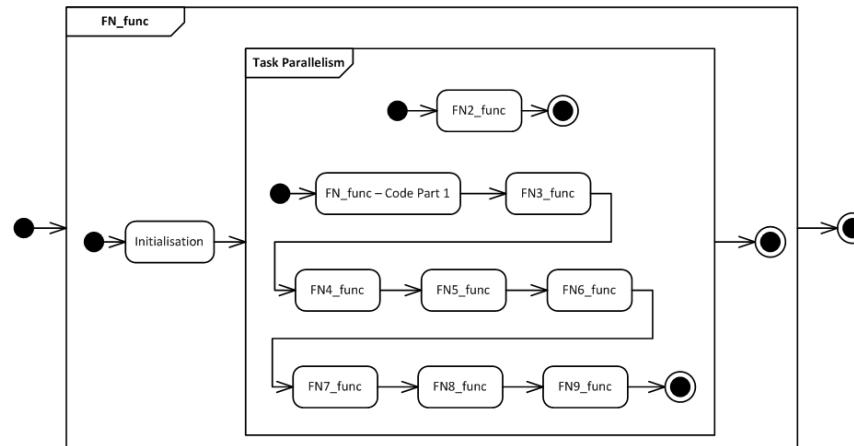
- Speed-down on 2 cores from dedicated use of crank-angle task on a core

Intra-Runnable – ADP and TAS (in collaboration with UAU)

- Runnable FR1_runnable can be split for execution on multiple cores



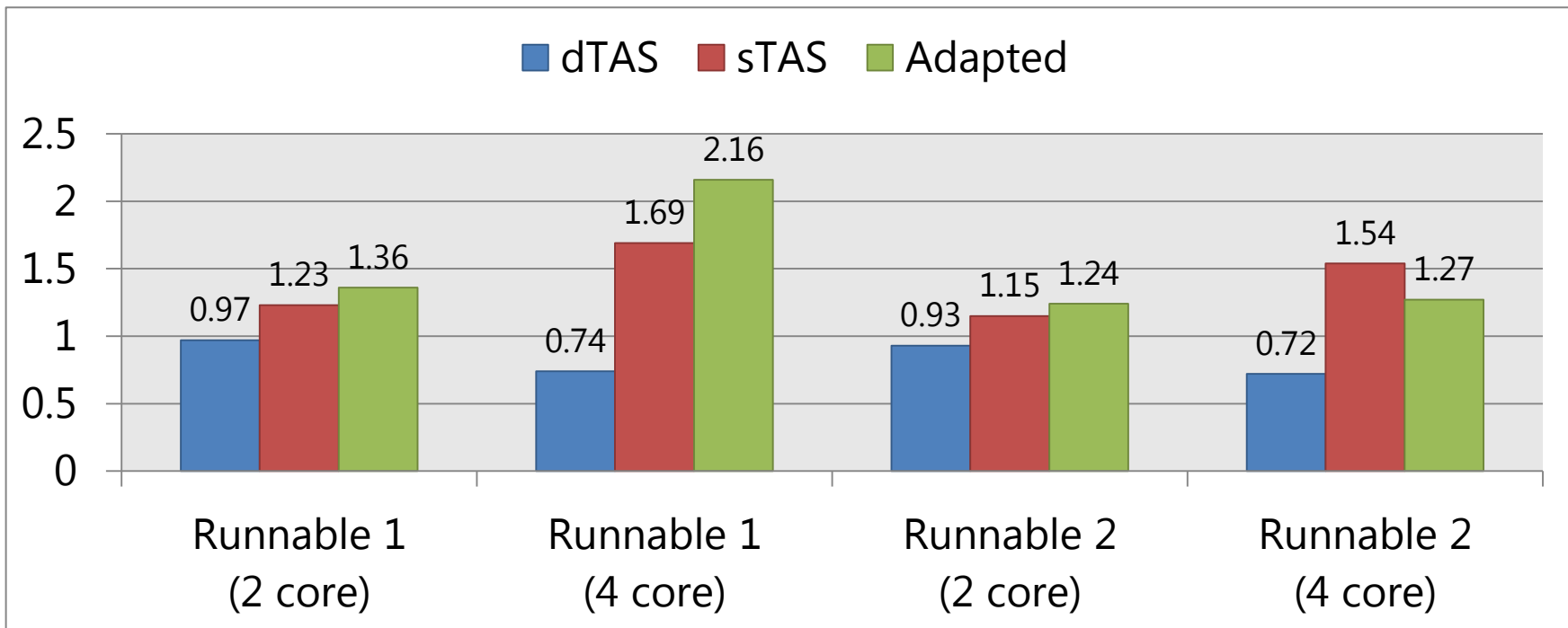
Activity and Pattern Diagram for R1_runnable



Activity and Pattern Diagram for FN_func

Results: Intra-Runnable Parallelism

- Timing Analysable Algorithmic Skeleton (TAS) used for implementation
 - Static allocated TAS (sTAS) assigns workload before execution
 - Dynamic allocated TAS (dTAS) assigns workloads during execution
 - Adapted is optimized for low overhead



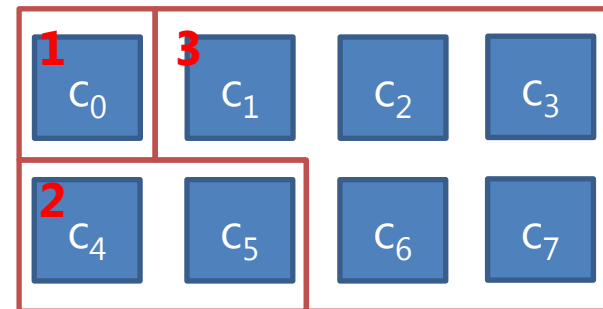
Combining approaches for WC scenario

- 8-core parMERASA processor
- Three groups using TIC (Inter-task)
- WCET of τ_{16} is greater than τ_{crBas}
 - Runnables in τ_{16} are scheduled with Mapping Tool on 2 cores (Intra-task)
- Tasks in T are scheduled with the Mapping Tool on cores 3 to 7 (Inter-task)

Group	Core	Tasks
1	0	τ_{crBas}
2	1 + 2	τ_{16}
3	3 – 7	$T = \{\tau_1, \tau_4, \tau_5, \tau_8, \tau_{20}, \tau_{32}, \tau_{64}, \tau_{96}, \tau_{128}, \tau_{1024}\}$

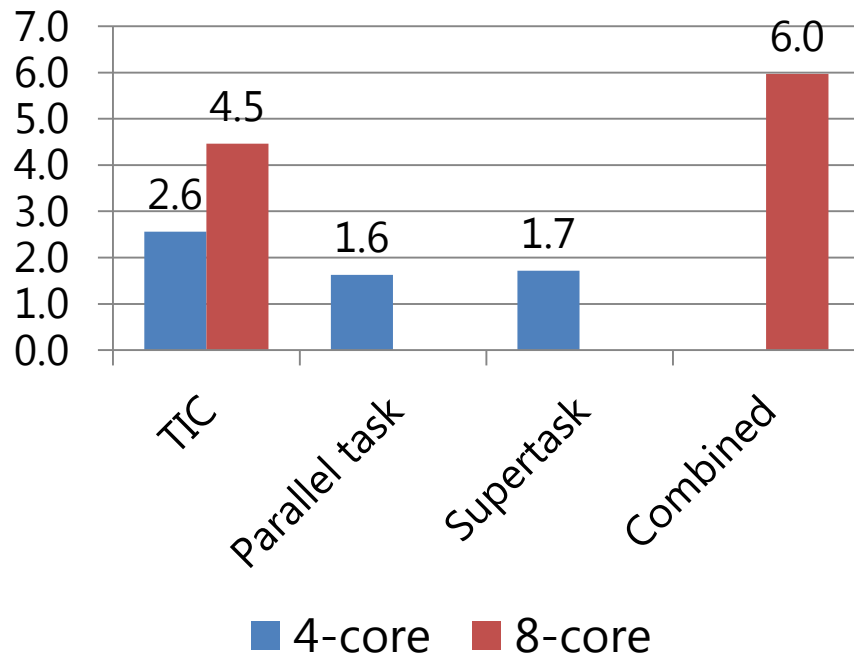


- WC scenario speed-up: 5.97*

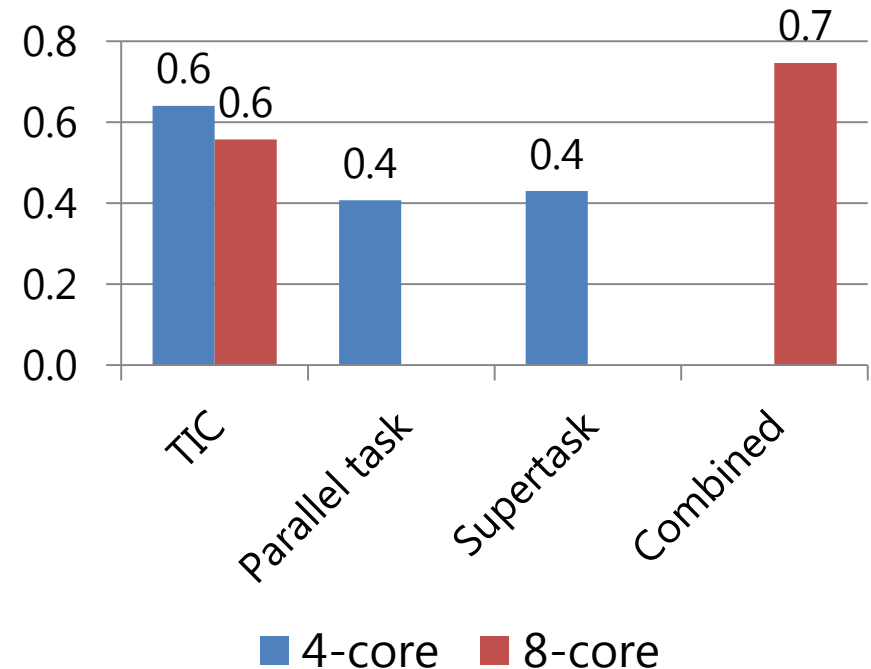


Summary of results for WC scenario

speed-up



efficiency



- Parallelization of the used EMS scales up to 8 cores with a combined approach
- Highest speed-up for WC scenario with combined approach (TIC + Mapping Tool)
 - Speed-up for worst-case scenario is **5.97** on 8-core parMERASA processor
 - Efficiency can potentially be further improved
- Parallelization of single Runnables suitable for frequently used server-Runnables
- Future work
 - Continue towards evaluation on real platform in the EMC² project
 - Uncover bottlenecks on real platforms
 - Investigate efficient implementations of TIC
 - Optimize holistic approach to reach a maximum WCET speed-up