T-CREST

TIME-PREDICTABLE MULTI-CORE ARCHITECTURE
FOR EMBEDDED SYSTEMS

Project Number 288008

D 8.2 T-CREST White Paper

Version 2.0
7 October 2013
Final

Public Distribution

Technical University of Denmark with Contributions from all Partners

Project Partners: **AbsInt Angewandte Informatik**, **Eindhoven University of Technology**, **GMVIS Skysoft**, **Intecs**, **Technical University of Denmark**, **The Open Group**, **University of York**, **Vienna University of Technology**

Project Partner Contact Information

AbsInt Angewandte Informatik Christian Ferdinand Science Park 1 66123 Saarbrücken, Germany Tel: +49 681 383600 Fax: +49 681 3836020 E-mail: ferdinand@absint.com	**Eindhoven University of Technology** Kees Goossens Potentiaal PT 9.34 Den Dolech 2 5612 AZ Eindhoven, The Netherlands E-mail: k.g.w.goossens@tue.nl
GMVIS Skysoft João Baptista Av. D. Joao II, Torre Fernao Magalhaes, 7 1998-025 Lisbon, Portugal Tel: +351 21 382 9366 E-mail: joao.baptista@gmv.com	**Intecs** Silvia Mazzini Via Forti trav. A5 Ospedaletto 56121 Pisa, Italy Tel: +39 050 965 7513 E-mail: silvia.mazzini@intecs.it
Technical University of Denmark Martin Schoeberl Richard Petersens Plads 2800 Lyngby, Denmark Tel: +45 45 25 37 43 Fax: +45 45 93 00 74 E-mail: masca@imm.dtu.dk	**The Open Group** Scott Hansen Avenue du Parc de Woluwe 56 1160 Brussels, Belgium Tel: +32 2 675 1136 Fax: +32 2 675 7721 E-mail: s.hansen@opengroup.org
University of York Neil Audsley Deramore Lane York YO10 5GH, United Kingdom Tel: +44 1904 325 500 E-mail: Neil.Audsley@cs.york.ac.uk	**Vienna University of Technology** Peter Puschner Treitlstrasse 3 1040 Vienna, Austria Tel: +43 1 58801 18227 Fax: +43 1 58801 918227 E-mail: peter@vmars.tuwien.ac.at

Contents

Document Control

Version	Status	Date
0.1	Short version for EC workshop	30 January 2012
0.2	Version for Dresden meeting	7 March 2012
1.0	Final version	15 March 2012
1.1	Internal review version for an update	31 August 2013
2.0	Update for the second review meeting	7 October 2013

Executive Summary

This document contains the deliverable *D 8.2 T-CREST White Paper* of work package 8 of the T-CREST project, due 6 months after project start as stated in the Description of Work, with an update after 24 months into the project. The document presents an overview of the T-CREST research project.

Abstract

Safety-critical systems are important parts of our daily life. Those systems have to be reliable, as our lives can depend on them. Examples are controllers in an airplane, braking controllers in a car, or train control systems. Those safety-critical systems need to be certified and the maximum execution time needs to be bounded and known so that response times can be assured when critical actions are needed. Note that just using a faster processor is not a solution for time predictability. Even with high performance processors in our desktop PCs we notice once in a while that the PC is "frozen" for a few seconds. For word processing we accept this minor inconvenience, but for a safety-critical system such a "pause" can result in a catastrophic failure.

The mission of T-CREST is to develop tools and build a system that prevents pauses by identifying and addressing the causes for potential pauses. The T-CREST time-predictable system will simplify the safety argument with respect to maximum execution time, and strive to double performance for 4 cores and to be 4 times faster for 16 cores than a standard processor in the same technology (e.g., FPGA). Thus the T-CREST system will result in lower costs for safety-relevant applications reducing system complexity and at the same time faster time-predictable execution.

1 Introduction

Standard computer architecture is driven by the following paradigm: make the common case fast and the uncommon case correct. This design approach leads to architectures where the average-case execution time is optimized at the expense of the worst-case execution time (WCET). Modeling the dynamic features of current processors, memories, and interconnects for WCET analysis often results in computationally infeasible problems. The bounds calculated by the analysis are thus overly conservative.

We need a sea change and we shall take the constructive approach by designing computer architectures where predictable timing is a first-order design factor. For real-time systems we thus propose to design architectures with a new paradigm: ***make the worst-case fast and the whole system easy to analyze***. Despite the advantages of analyzable system resources, only a few research projects exist in the field of hardware optimized for the WCET.

Within T-CREST we will propose novel solutions for time-predictable multi-core and many-core system architectures. The resulting time-predictable resources (processor, interconnect, memories, etc.) will be a good target for WCET analysis and the WCET performance will be outstanding compared to current processors. Time-predictable caching and time-predictable chip-multiprocessing (CMP) will provide a solution for the need of increased processing power in the real-time domain.

Besides the hardware, a compiler infrastructure will be developed in the project. WCET-aware optimization methods will be developed along with detailed timing models such that the compiler benefits from the known behavior of the hardware. The WCET analysis tool aiT will be adapted to support the developed hardware and guide the compilation.

The T-CREST hardware will be open-source under the industry friendly, simplified BSD license.

1.1 Project Objectives

The main objectives of the T-CREST project are:

- The T-CREST platform shall provide a time-predictable multi-core platform so that reliable WCET analysis can be performed and tighter WCET bounds enable a higher processor utilisation.
- The T-CREST platform shall support a global asynchronous local synchronous system to enable the implementation of large scale multicores. An asynchronous network on chip will connect synchronous processors.
- The T-CREST memory network on chip and memory controller shall support time-predictable access to the shared main memory to allow WCET analysis of accesses to the shared memory.
- The T-CREST compiler and the WCET analysis tool aiT shall be tightly integrated to support the T-CREST processor Patmos and to enable WCET driven compiler optimisation.
- The T-CREST platform shall strive to double the worst case performance for 4 cores and to be 4 times faster for 16 cores then a single processor to provide more processing power for future, more complex embedded real-time systems.

From those main objectives we derived fine grade objectives and requirements for the T-CREST platform, which are listed in Deliverable 1.1 – "Evaluation Requirements".

1.2 Expected Market Impact

Embedded systems are taking over control in ever more demanding environments, including safety- and security-critical systems. The robustness and safety of systems is therefore an ever-growing competitiveness factor. The aptitude to produce robust predictable systems at a competitive price will be key to keeping European companies at the cutting edge of the embedded system market. A large number of European companies will benefit via the project coordinator, The Open Group having close links to more than 300 companies involved in technology evolution. Direct impact on the market position of 3 European companies participating in the project is expected, namely the tools company AbsInt as well as GMV and INTECS offering safety-critical solutions for Aeronautics and Transport. T-CREST will also help European industry to build reliable systems, not only in the areas of air and ground transportation, but also in many other areas where robustness, availability, and safety are important requirements to embedded systems.

2 Project Areas

T-CREST covers technologies from the chip level (processor, memory, asynchronous network-on-chip), via compiler, single-path code generation, and WCET analysis tools, up to system evaluation with two industry use cases. New languages or new operating system concepts for time-predictable real-time systems are not in the scope of the project. Figure 1 shows the individual work packages and interplay between those.

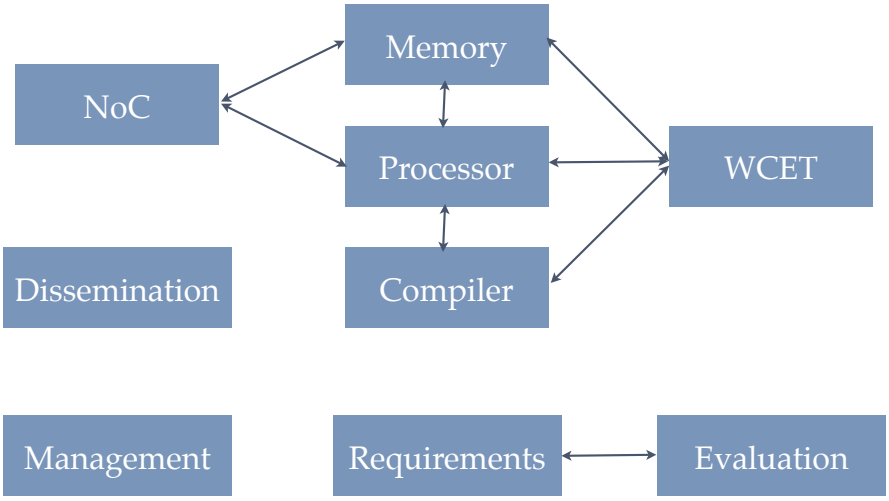

Figure 1: Work packages of T-CREST and their interplay

2.1 The Processor

The basis of a time-predictable system is a time-predictable processor. Within T-CREST we develop a time-predictable processor, named Patmos [29], as one approach to attack the complexity issue of WCET analysis. Patmos is a statically scheduled, dual-issue RISC processor that is optimized for real-time systems. All instruction delays are visible at the instruction set architecture (ISA). This puts more burden on the compiler, but simplifies the WCET analysis tool. A major challenge for the WCET analysis is the memory hierarchy with multiple levels of caches. We attack this issue by caches that are especially designed for WCET analysis. For instructions we adopt the method cache [23], which operates on whole functions/methods and thus simplifies the modeling for WCET analysis. Furthermore, we propose a split-cache architecture [24, 28] for data, offering dedicated caches for the stack area [1], constants, static data, heap allocated objects, as well as a compiler and program managed scratchpad memory.

Accesses to the different types of data areas are explicitly encoded with the load and store instructions. We call this typed load and store instructions, which direct the loads and stores to the relevant cache. This feature helps the WCET analysis to distinguish between the different data caches. Furthermore, it can be detected earlier in the pipeline which cache will be accessed.

Patmos also supports predication of all instructions. This feature reduces the number of conditional branches and supports generation of single-path code [21, 20]. The compiler LLVM is extended with an optimization path that translates normal code into single-path code.

2.2 The Interconnect

In order to build a chip-multiprocessor system out of Patmos processor cores we need a suitable interconnect – a network-on-chip (NoC). The Patmos multi-processor platform will use non-coherent distributed memory and the address space will be populated by smaller memory blocks within the different processor nodes and a large block of off-chip memory. The NoC will support time-predictable

write transactions to memories in other processor nodes. The shared off-chip memory will be supported by a specific memory NoC. The two NoCs are shared resources and must support multiple concurrent read and write transactions.

To enable time-predictable usage of a shared resource the resource arbitration has to be time-predictable. In the case of a NoC, statically scheduled TDM is a time-predictable solution [27]. This static schedule is repeated and the length of the schedule is called the *period*. Like tasks in real-time systems, also the communication is organized in periods. One optimization point of the design is minimizing the period to minimize the latency of delivering data and the size of the schedule tables. The T-CREST NoC uses TDM from end to end, including the network interface. That approach also results in an efficient implementation of the network interface [30].

In the field of embedded systems, multi-processor platforms are typically optimized for a given application or application domain. The NoC structure and/or the routing schedules are then optimized and are then application-specific. In the T-CREST project our aim is to develop a general-purpose platform – a platform that can be configured to optimize the performance of the system or a platform which can be used as is without any configuration.

Different types of data are transferred on the NoC, e.g., message passing data between cores, cache fills from main memory, synchronization operations such as compare-and-swap. In most architectures a single NoC serves all those different types of data. However, the requirements of these different data types with respect to e.g., packet size, address ranges, and flow control are different. Therefore, we will evaluate if several, for the traffic type optimized, NoCs result in a more efficient solution than a single shared NoC.

2.3 Memory Hierarchy

The only memory layer that is under direct control of the compiler is the register file. Other levels of the memory hierarchy are usually not visible – they are not part of the ISA abstraction. The placing of data in the different layers is automatically performed. While caches are managed by the hardware, virtual memory is managed by the operating system (OS). The access time for a word that is in a memory block paged out by the OS is several orders of magnitude higher than a first level cache hit. Even the difference between a first level cache access and a main memory access is in the order of two magnitudes.

Cache memories for instructions and data are classic examples of the paradigm *Make the common case fast*. Plenty of effort has gone into researching the integration of the instruction cache into the timing analysis [3] and the integration of the cache analysis with the pipeline analysis [10]. The influence of different cache architectures on WCET analysis is described in [12].

Caches in general, and particularly data caches, are usually hard to analyze statically. Therefore, we introduce caches that are organized to speed-up execution time and provide tight WCET bounds. We propose a split cache architecture consisting of: (1) an instruction cache for full methods, (2) a stack cache, (3) a cache for static data, constants, and type information, and (4) a small, fully associative buffer for heap access. Furthermore, we will also consider the integration of program- or compiler-managed scratchpad memory for data storage and inter-processor communication to tighten bounds for hard-to-analyze memory-access patterns.

The shared memory abstraction that is prevalent in (embedded) systems introduces contention on the shared memories. Especially off-chip SDRAM and Flash memories, which are performance bottlenecks, are heavily shared between multiple processors. Cache misses or scratchpad memories prefetches therefore have a highly variable response time due to contention between processors. Worse, even without sharing, response times are variable due to effects such as different read/write latencies, and bank open/close effects. In T-CREST we create time-predictable external SDRAM memory controllers [7, 8], as this is an essential ingredient in every embedded system.

The combination of the TDM based NoC and the time-predictable memory controller allows, even on a CMP system, to provide upper bounds on memory transactions. This upper bound enables WCET analysis of individual tasks executing on a CMP system.

2.4 Compiler and WCET Analysis

The performance of the dual-issue processor depends on statically scheduled instructions. We argue that all architectural features of a processor shall be exposed to the compiler to generate time-predictable code. Within T-CREST the LLVM compiler framework will be adapted to target Patmos. Furthermore, we will explore compiler optimizations for the WCET instead of the average case execution time.

The processor is intended as a platform to explore various time-predictable design trade-offs and their interaction with WCET analysis techniques as well as WCET-aware compilation. We propose the co-design of time-predictable processor features with the WCET analysis tool, similar to the work by Huber et al. [13] on caching of heap allocated objects in a Java processor. Only features where we can provide a static program analysis shall be added to the processor. This includes, but is not limited to, time-predictable caching mechanisms, chip-multiprocessing (CMP), as well as novel pipeline organizations.

The WCET analysis tool aiT from AbsInt will be adapted to support the VLIW processor Patmos. It is also the platform for exploration of time-predictable processor features. The WCET oriented optimization in the compiler will be tightly integrated with the WCET analysis tool [22]. The WCET tool will provide information on the worst-case path and basic block timings to guide the optimization process.

2.5 Evaluation

T-CREST will be evaluated by use cases from two industrial partners: GMV and INTECS. GMV will port a set of real-world applications from the aeronautical domain with extreme safety requirements to the T-CREST platform. INTECS will port a specific use case from the railway industry.

The first use case consists of three applications from the avionics domain. One application is intended for the highest criticality and therefore needs a rigorous WCET and scheduling analysis. The demonstrator system is composed of three applications, all compliant to ARINC 653: (1) The Airline Operational Centre (AOC) is the on-board component of an Air Traffic and Trajectory Management System. The application was developed according to DO-178B, it has about 30 LoC and is as such a complex embedded application. (2) A Crew Alert System that is a typical highly critical on-board

application found on civil aircraft of all major manufacturers. The GMV implementation is based on real industrial requirements and is used as a study prototype by GMV customers. (3) The Control Application is a typical closed-loop control application that was developed for training and demonstration purposes. It has strong real-time requirements, but is relatively simple, and appears to be an interesting subject to further studies on thread-level parallelism.

INTECS provides a railway use case, which is an application within the European Railway Traffic Management System (ERTMS). The ERTMS has been conceived by the European railways and by the supplier industry supported by the European Commission to meet the future needs of the European rail transport network. INTECS will provide a specific use case that will focus on the monitoring system of the GSM-R radio-link called GRIDES (GSM-R Integrity Detection System). The GRIDES system performs acquisition and analysis of the GSM-R radio signal within the allocated bandwidth (both in the uplink and downlink directions) in proximity of one (or more) HS/HC railway line(s). GRIDES can receive, process, and analyze both the useful GSM-R signal and the radio interference sources eventually encountered by the HS/HC train during its run, logging information about the quality of the radio link. INTECS will port the GRIDES implementation to the time-predictable T-CREST platform and utilize the supporting tools from the project for analysis and optimizations. The GRIDES system has specific timing constraints and requirements for predictability. It is also computationally intensive where increased parallelism will provide substantial benefits and enable additional system capabilities foreseen in the original conception of ERTMS, but not yet possible due to current limitations in platform architectures and complexities in implementing a time-predictable multi-threaded adaptation of GRIDES and other ERTMS components. Baseline measurements of the current GRIDES implementation will be used to measure the improvements and innovative processing and performance capabilities provided by the T-CREST technologies.

3 Project Partners

The T-CREST consortium consist of the following industrial, academic, technology, and standardization organizations:

AbsInt Angewandte Informatik

Eindhoven University of Technology

GMVIS Skysoft

Intecs

Technical University of Denmark

The Open Group

University of York

Vienna University of Technology

4 Related Projects

The research on time-predictable architectures is a steadily growing research field, which is gaining momentum, partially through EC funded projects. Here we list related projects and how results from former projects are used within the T-CREST project.

4.1 MERASA

The FP-7 project MERASA (Multi-Core Execution of Hard Real-Time Applications Supporting Analysability) [32] investigated a bus-based CMP with a multi-threaded version of the TriCore processor. The memory hierarchy issue for real-time systems was attacked by a dynamic instruction scratchpad [17]. The WCET analysis tool used in MERASA was adapted to support the instruction set of the TriCore. A memory arbiter was designed to support measurement-based WCET analysis with the RapiTime tool [19].

In contrast to MERASA we envision a network-on-chip based multi-core architecture. Furthermore, we attack the time predictability challenge by developing a new processor architecture (WCET optimized ISA) and the supporting compiler.

A followup project, parMERASA, tackles now the parallelization of applications. T-CREST and parMERASA are complementary and we intend to cooperate with the parMERASA team. We have already organized a common workshop, the "1st Workshop on Advanced Real-time Processor Architectures (ARPA 2013)", held on 22 January in Berlin at the HiPEAC conference.[1]

4.2 PREDATOR

The FP-7 project PREDATOR studied the interplay between efficiency requirements and critical constraints in embedded system design, aiming at a design for predictability and efficiency. There have

[1] See: `http://www.parmerasa.eu/index.php?menu=arpa`

been various results on the notion of predictability in general and on the predictability of various hardware and software features, leading to advice what to avoid when developing a predictable system [31].

This work provided one of the foundations for the work in T-CREST. The processor design in T-CREST is based on the principles of predictable architectures elaborated in the PREDATOR project. The T-CREST work on multi-core applications and NoC goes beyond the scope of PREDATOR.

The work on the WCC compiler in PREDATOR represents an important first step towards developing WCET-aware compilation techniques for a single-core architecture by selecting between alternative code sequences. A disciplined approach for the design of WCET-aware optimizations in the context of parallelism and more complex (distributed) memory was, however, not yet developed in PREDATOR and is the topic of research in T-CREST. The LLVM compiler framework used in T-CREST is more powerful than the framework used in PREDATOR.

4.3 SCALOPES

In Scalopes scalable architectures with composability, predictability and dependability properties have been developed. In particular the work performed by the Delft university of technology, in the team of Prof. Goossens who is now at the Eindhoven university of technology, led to the definition of a tile-based system architecture that has inspired the T-CREST architecture. In particular, the network on chip (Aethereal/aelite) used in Scalopes has been improved in T-CREST to be asynchronous, and the architecture of the network interfaces has been completely redesigned in T-CREST for a reduced system-level cost. Similarly, the Scalopes tile architecture that uses a MicroBlaze processor has been simplified and improved by replacing a proprietary processor by the open-source Patmos processor.

4.4 JEOPARD

The FP-7 JEOPARD (Java Environment for Parallel Realtime Development) investigated architectures and tools for real-time Java on CMP systems. JEOPARD considered all levels of the architecture: the hardware, the operating system, the JVM, static analysis tools, and evaluation of the architecture with three use cases.

Within the hardware architecture work package the Java processor JOP was extended to support time-predictable execution of Java applications on a CMP. The TDMA based memory access arbitration was incorporated into the WCET analysis tool of JOP. Research on time-predictable caching for Java started within JEOPARD.

Within T-CREST we will use the results from the memory access arbitration and will adapt the cache solutions from the Java processor to a RISC processor. The time-predictable cache organization that was developed within JEOPARD [24] will be adapted for T-CREST. The analysis methodology developed for an object cache of the Java processor JOP [14] will be adapted within the T-CREST project in the analysis tool Platin [22].

4.5 ALL-TIMES

The main goal of the FP-7 project ALL-TIMES (reference 215068) [9] was to enhance the timing analysis technology for safety-critical embedded systems. The project aimed at combining available timing tools from SMEs and universities into an integrated tool chain using open tool frameworks and interfaces, and at developing new timing analysis methods and tools where appropriate. The XTC framework [2] developed in INTEREST and improved in ALL-TIMES provides an open standard for tool communication.

Specifically, the project was concerned with the integration of static analysis techniques on the code level and on the system level with measurements and with extending timing analysis to the early phases of system design. The latter led to the development of AbsInt's TimingExplorer [18, 6, 11], a variant of aiT particularly suited to explore the timing properties of different hardware layouts, which can be re-used (after adaptation) for the WCET driven design space exploration.

4.6 PRET

Edwards and Lee argue: "It is time for a new era of processors whose temporal behavior is as easily controlled as their logical function" [5]. A first simulation of their PRET architecture is presented in [15]. PRET implements a RISC pipeline and performs chip-level multithreading for four threads to eliminate data forwarding and branch prediction [16]. Scratchpad memories are used instead of instruction and data caches. The shared main memory is accessed via a TDMA scheme, called the memory wheel. A recent version of PRET [4] defines time-predictable access to SDRAM by assigning each thread a dedicated bank in the memory chips. The access to the individual banks is pipelined and the access time fixed. As the memory banks are not shared between threads, thread communication has to be performed via the shared scratchpad memory. Although the PRET architecture is not a CMP system, the concepts used in the multi-threaded pipeline can also be applied to a CMP system. We intend to evaluate the PRET memory controller with our CMP system. Each bank will be assigned to a set of CPUs. Within this set we will perform a TDMA based memory arbitration.

The main difference between our proposal and PRET is that we focus on time predictability [25, 26] and PRET on repeatable timing. Our approach therefore allows run-time dynamism in scheduling and execution, whereas PRET is essentially static, resulting in a much higher implementation cost. In our opinion a time-predictable architecture does not need to provide repeatable timing as long as WCET analysis can deliver tight WCET bounds. Furthermore, PRET implements the ARM instruction set, whereas we will explore an instruction set that supports WCET based optimization and WCET analysis.

5 Future Research

T-CREST is intended as a research start on time-predictable CMP systems. We envision that there remain several research challenges in the mid- and long-term. T-CREST will provide a prototype implementation in an FPGA for evaluation. A mid-term follow-up project would be to explore the T-CREST results in the context of ASICs (maybe together with ARM). WCET-aware compilation

and WCET analysis based co-design are very new research topics. We assume that this direction of research will continue in the mid-term.

More and more systems become safety-critical and some of those systems shall also support mixed-critical applications. Therefore, we envision more importance on time-predictable and analyzable systems. Mixed-critical systems will use time-predictable features to support isolation between the critical and non-real-time parts of the application.

In the long-term we envision that safety-critical and real-time systems will become ubiquitous and therefore we are in desperate need of time-predictable platforms and efficient development methods.

6 Conclusion

The T-CREST project will research time-predictable architectures for future safety-critical systems. It is a 3 year STREP project funded by the EC under grant agreement no. 288008. The end product is a multi-core platform including time-predictable processor elements, network interconnect, and a memory hierarchy. A compiler with WCET based optimization and a version of the WCET analysis tool aiT supports this hardware platform.

The sources of the implementation are available in open source: `https://github.com/t-crest`. Further information can be found at the project web site: `http://www.t-crest.org`.

Acknowledgement

This work was partially funded under the European Union's 7th Framework Programme under grant agreement no. 288008: Time-predictable Multi-Core Architecture for Embedded Systems (T-CREST).

The main author of this document is Martin Schoeberl from DTU. Contributions came from all partners, in particular from Benny Akesson, Neil Audsley, Florian Brandner, Christian Ferdinand, Kees Goossens, Scott Hansen, Reinhold Heckmann, Guido Ioele, Peter Puschner, Tobias Schoofs, Jens Sparsø, and Jack Whitham.

References

[1] Sahar Abbaspour, Florian Brandner, and Martin Schoeberl. A time-predictable stack cache. In *Proceedings of the 9th Workshop on Software Technologies for Embedded and Ubiquitous Systems*, 2013.

[2] AbsInt Angewandte Informatik GmbH. XTC: XML Timing Cookie Language. `http://www.absint.com/xtc/`, 2013.

[3] Robert Arnold, Frank Mueller, David Whalley, and Marion Harmon. Bounding worst-case instruction cache performance. In *IEEE Real-Time Systems Symposium*, pages 172–181, 1994.

[4] Stephen A. Edwards, Sungjun Kim, Edward A. Lee, Isaac Liu, Hiren D. Patel, and Martin Schoeberl. A disruptive computer design idea: Architectures with repeatable timing. In *Proceedings of IEEE International Conference on Computer Design (ICCD 2009)*, Lake Tahoe, CA, October 2009. IEEE.

[5] Stephen A. Edwards and Edward A. Lee. The case for the precision timed (PRET) machine. In *DAC '07: Proceedings of the 44th annual conference on Design automation*, pages 264–265, New York, NY, USA, 2007. ACM.

[6] C. Ferdinand, R. Heckmann, D. Kästner, and S. Nenova. Architecture exploration and timing estimation during early design phases. *Embedded World Congress, Nuremberg*, March 2010.

[7] Manil Dev Gomony, Benny Akesson, and Kees Goossens. Architecture and optimal configuration of a real-time multi-channel memory controller. In *Design, Automation Test in Europe Conference Exhibition (DATE), 2013*, pages 1307–1312, 2013.

[8] Sven Goossens, Benny Akesson, and Kees Goossens. Conservative open-page policy for mixed time-criticality memory controllers. In *Proceedings of the Conference on Design, Automation and Test in Europe*, DATE '13, pages 525–530, San Jose, CA, USA, 2013. EDA Consortium.

[9] Jan Gustafsson, Björn Lisper, Markus Schordan, Christian Ferdinand, Marek Jersak, and Guillem Bernat. ALL-TIMES – a European project on integrating timing technology. In Tiziana Margaria and Bernhard Steffen, editors, *Proc. Third International Symposium on Leveraging Applications of Formal Methods (ISOLA'08)*, pages 445–459. Springer, October 2008.

[10] Christopher A. Healy, Robert D. Arnold, Frank Mueller, David B. Whalley, and Marion G. Harmon. Bounding pipeline and instruction cache performance. *IEEE Trans. Computers*, 48(1):53–70, 1999.

[11] R. Heckmann, F. Ferdinand, D. Kästner, and S. Nenova. Architecture Exploration and Timing Estimation during Early Design Phases. In *International Journal on Software Tools for Technology Transfer (STTT) 2012*. SpringerLink OnlineFirst, July 2012.

[12] Reinhold Heckmann, Marc Langenbach, Stephan Thesing, and Reinhard Wilhelm. The influence of processor architecture on the design and results of WCET tools. *Proceedings of the IEEE*, 91(7):1038–1054, Jul. 2003.

[13] Benedikt Huber, Wolfgang Puffitsch, and Martin Schoeberl. WCET driven design space exploration of an object cache. In *Proceedings of the 8th International Workshop on Java Technologies for Real-time and Embedded Systems (JTRES 2010)*, pages 26–35, New York, NY, USA, 2010. ACM.

[14] Benedikt Huber, Wolfgang Puffitsch, and Martin Schoeberl. Worst-case execution time analysis driven object cache design. *Concurrency and Computation: Practice and Experience*, 24(8):753–771, 2012.

[15] Ben Lickly, Isaac Liu, Sungjun Kim, Hiren D. Patel, Stephen A. Edwards, and Edward A. Lee. Predictable programming on a precision timed architecture. In Erik R. Altman, editor, *Proceedings of the International Conference on Compilers, Architecture, and Synthesis for Embedded Systems (CASES 2008)*, pages 137–146, Atlanta, GA, USA, October 2008. ACM.

[16] Isaac Liu, Jan Reineke, and Edward A. Lee. A PRET architecture supporting concurrent programs with composable timing properties. In *Signals, Systems and Computers, 2010 Conference Record of the Forty-Four Asilomar Conference on*, November 2010.

[17] Stefan Metzlaff, Sascha Uhrig, Jörg Mische, and Theo Ungerer. Predictable dynamic instruction scratchpad for simultaneous multithreaded processors. In *Proceedings of the 9th workshop on Memory performance (MEDEA 2008)*, pages 38–45, New York, NY, USA, 2008. ACM.

[18] Stefana Nenova and Daniel Kästner. Worst-case timing estimation and architecture exploration in early design phases. In Niklas Holsti, editor, *9th Intl. Workshop on Worst-Case Execution Time (WCET) Analysis*, Dagstuhl, Germany, 2009. Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany.

[19] Marco Paolieri, Eduardo Quiñones, Francisco J. Cazorla, Guillem Bernat, and Mateo Valero. Hardware support for WCET analysis of hard real-time multicore systems. In *The 36th International Symposium on Computer Architecture (ISCA 2009)*, pages 57–68, Austin, Texas, USA, 20-24, June 2009. ACM.

[20] Peter Puschner. Transforming execution-time boundable code into temporally predictable code. In Bernd Kleinjohann, K.H. (Kane) Kim, Lisa Kleinjohann, and Achim Rettberg, editors, *Design and Analysis of Distributed Embedded Systems*, pages 163–172. Kluwer Academic Publishers, 2002. IFIP 17th World Computer Congress - TC10 Stream on Distributed and Parallel Embedded Systems (DIPES 2002).

[21] Peter Puschner and Alan Burns. Writing temporally predictable code. In *Proceedings of the The Seventh IEEE International Workshop on Object-Oriented Real-Time Dependable Systems (WORDS 2002)*, pages 85–94, Washington, DC, USA, 2002. IEEE Computer Society.

[22] Peter Puschner, Daniel Prokesch, Benedikt Huber, Jens Knoop, Stefan Hepp, and Gernot Gebhard. The T-CREST approach of compiler and WCET-analysis integration. In *9th Workshop on Software Technologies for Future Embedded and Ubiquitious Systems (SEUS 2013)*, pages 33–40, 2013.

[23] Martin Schoeberl. A time predictable instruction cache for a Java processor. In *On the Move to Meaningful Internet Systems 2004: Workshop on Java Technologies for Real-Time and Embedded Systems (JTRES 2004)*, volume 3292 of *LNCS*, pages 371–382, Agia Napa, Cyprus, October 2004. Springer.

[24] Martin Schoeberl. Time-predictable cache organization. In *Proceedings of the First International Workshop on Software Technologies for Future Dependable Distributed Systems (STFSSD 2009)*, pages 11–16, Tokyo, Japan, March 2009. IEEE Computer Society.

[25] Martin Schoeberl. Time-predictable computer architecture. *EURASIP Journal on Embedded Systems*, vol. 2009, Article ID 758480:17 pages, 2009.

[26] Martin Schoeberl. Is time predictability quantifiable? In *International Conference on Embedded Computer Systems (SAMOS 2012)*, Samos, Greece, July 2012. IEEE.

[27] Martin Schoeberl, Florian Brandner, Jens Sparsø, and Evangelia Kasapaki. A statically scheduled time-division-multiplexed network-on-chip for real-time systems. In *Proceedings of the 6th International Symposium on Networks-on-Chip (NOCS)*, pages 152–160, Lyngby, Denmark, May 2012. IEEE.

[28] Martin Schoeberl, Benedikt Huber, and Wolfgang Puffitsch. Data cache organization for accurate timing analysis. *Real-Time Systems*, 49(1):1–28, 2013.

[29] Martin Schoeberl, Pascal Schleuniger, Wolfgang Puffitsch, Florian Brandner, Christian W. Probst, Sven Karlsson, and Tommy Thorn. Towards a time-predictable dual-issue microprocessor: The Patmos approach. In *First Workshop on Bringing Theory to Practice: Predictability and Performance in Embedded Systems (PPES 2011)*, pages 11–20, Grenoble, France, March 2011.

[30] Jens Sparsø, Evangelia Kasapaki, and Martin Schoeberl. An area-efficient network interface for a tdm-based network-on-chip. In *Proceedings of the Conference on Design, Automation and Test in Europe*, DATE '13, pages 1044–1047, San Jose, CA, USA, 2013. EDA Consortium.

[31] Lothar Thiele and Reinhard Wilhelm. Design for timing predictability. *Real-Time Systems*, 28(2-3):157–177, 2004.

[32] T. Ungerer, F. Cazorla, P. Sainrat, G. Bernat, Z. Petrov, C. Rochange, E. Quiñones, M. Gerdes, M. Paolieri, and J. Wolf. Merasa: Multi-core execution of hard real-time applications supporting analysability. *Micro, IEEE*, 30(5):66–75, 2010.