IST-214373 ArtistDesign
Network of Excellence
on Design for Embedded Systems

Activity - Progress Report for Year 2

Timing Analysis

Clusters:

SW Synthesis, Code Generation and Timing Analysis

Activity Leader:

Prof. Björn Lisper (Mälardalen University)
http://www.idt.mdh.se/~blr/

Policy Objective (abstract)

The activity gathers the most prominent groups in the timing analysis area. They have all previously worked together in the ARTIST2 NoE, and therefore have well established links. The theme of the activity, timing analysis of MPSoC systems, is basically a new field scientifically, and also very timely from an application perspective as MPSoC and Multicore architectures rapidly are becoming mainstream. A research effort in this area will thus establish European dominance in a field that rapidly is becoming very important. ArtistDesign also provides a close to perfect environment for this research due to the relevant competence in other activities and clusters, such as the compiler groups in the local cluster, and the MPSoC cluster.

Versions

number	comment	date
1.0	First version delivered to the reviewers	December 18th 2009

Table of Contents

1. Overview of the Activity

1.1 ArtistDesign Participants and Roles

Prof. Dr. Reinhard Wilhelm – Saarland University (USaar; Germany)

Compiler Design, Static Program Analysis, Timing Analysis

Saarland University has developed much of the timing-analysis technology that is further developed and commercialised by the spin-off company AbsInt.

Dr. Iain Bate – University of York (UK)

Research on Timing Analysis

Prof. Dr. Björn Lisper – Mälardalen University (MDH; Sweden)

Activity Leader, Timing-AnalysisTools

Mälardalen University is working on automatic flow analysis, WCET analysis case studies on industrial code, the maintenance of a WCET-benchmark suite, the definition of interface formats for timing analysis, and the use of WCET tools in education. Mälardalen University is coordinating the integration activity.

Dr. Jan Gustafsson – Mälardalen University (MDH; Sweden)

Timing Analysis Research and Tools, WCET Analysis Case Studies

Dr. Andreas Ermedahl – Mälardalen University (MDH; Sweden)

Timing Analysis Research and Tools, WCET Analysis Case Studies

Prof. Dr. Peter Puschner – TU Vienna (Austria)

Timing-Analysis Tools and Temporally Predictable HW-SW Architectures

Within the Timing-Analysis Activity TU Vienna focuses on measurement-based WCET analysis and on hardware and software architectures that provide time-predictability and composability.

Prof. Dr. Peter Marwedel – TU Dortmund (Germany)

Architecture-Aware Compilation, Low-Power Code Generation, Development of Optimizations for WCET Minimization.

Dr. Claire Burguière – Saarland University (USaar; Germany)

Research on Scheduling and Timing Analysis in the presence of interrupts

Dr. Raimund Kirner – TU Vienna (Austria)

Timing-Analysis Tools and Compilation with Support for Timing Analysis, definition of Annotation Language for WCET Analysis, Measurement-Based Timing Analysis.

Changes wrt Y1 deliverable --

PhD students were removed from the list of participants.
They were erroneously listed in the Y1 report.

1.2 Affiliated participants and their role within the Activity

Dr. Christian Ferdinand – AbsInt GmbH (Germany)

Tool Supplier

AbsInt provides advanced WCET analysis tools for a wide variety of targets. The work within ArtistDesign focuses on the advance of WCET analysis techniques and its integration in other tools.

Gernot Gebhard – AbsInt GmbH (Germany)

Generic specification of processor components for generation and validation of timing analyses.

Dr. Niklas Holsti – Tidorum Ltd. (Finland)

Timing-Analysis Tools

Tidorum supplies the WCET analysis tool Bound-T for several targets. In ArtistDesign Tidorum aims to make the tool applicable to single cores of suitably decoupled multi-core systems, integrated within some multi-core scheduling tool.

Prof. Abhik Roychoudhury – National Univ. of Singapore (NUS; Singapore) – International affiliate

Timing Analysis

The group at National University of Singapore (NUS) has studied methods for micro-architectural modelling in timing analysis, leading to the Chronos timing analyzer. The work on Chronos has led to related works on supporting predictable execution, e.g., by developing scratchpad memory allocation schemes for sequential as well as concurrent embedded software. Recently the NUS group has focused on linking timing analysis with model-driven software development flows, with the goal of model-level performance debugging.

-- Changes wrt Y1 deliverable --

No changes with respect to Year 1.

1.3 Starting Date, and Expected Ending Date

January 1st, 2008 until there is a framework for timing analysis of MPSoC systems.

-- Changes wrt Y1 deliverable --

No changes with respect to Year 1.

1.4 Policy Objective

The activity gathers the most prominent groups in the timing analysis area. They have all previously worked together in the ARTIST2 NoE, and therefore have well established links. The theme of the activity, timing analysis of MPSoC systems, is basically a new field scientifically, and also very timely from an application perspective as MPSoC and multicore architectures rapidly are becoming mainstream. A research effort in this area will thus establish European dominance in a field that rapidly is becoming very important. ArtistDesign also provides a close to perfect environment for this research due to the relevant competence in other activities and clusters, such as the compiler groups in the local cluster, and the MPSoC cluster.

-- Changes wrt Y1 deliverable --

No changes with respect to Year 1.

1.5 Background

All the partners in this activity have participated in the NoE Artist2. They developed a common tool architecture, and exchanged tool components. They have created the WCET Tool Challenge, executed in 2006 and 2008, to evaluate the existing commercial tools and academic prototypes. The Tool Challenge is planned to be executed every second year with improved conditions and more challenging benchmarks.

-- Changes wrt Y1 deliverable --

No changes with respect to Year 1.

1.6 Technical Description: Joint Research

Traditional static timing analysis has three parts: the flow analysis, which finds constraints on the possible program flows, the low-level analysis, which applies hardware timing models to obtain timing estimates for short execution paths, and the calculation which combines the result of the two previous analyses to obtain an estimate of the WCET for the full code.

Timing analysis on code level has so far dealt almost exclusively with sequential programs running in isolation. For MPSoC and multi-core architectures, these assumptions will no longer be valid: tasks might be parallel, and different tasks will run in parallel on different sets of cores. Timing analysis of parallel code, running on parallel hardware, is a new research area, and the aim of this activity is to initiate research in this area. Due to its novel nature, a large part of the research, at least for the first 18 months, will consist of initial investigations, paving the way for future in-depth research.

Some research problems:

- Flow analysis needs to be extended from single-threaded programs to multi-threaded programs with possible synchronization between threads.
- Current low-level analysis is restricted to synchronous processor models: only [Thesing06] has modelled processor periphery. Hence, hardware modelling must be extended to include asynchronous systems, ultimately including full MPSoC and multicore architectures.

- Novel WCET calculation methods must be developed, which take into account that several interacting threads may have to complete in order for a task to complete.
- Methods to handle common resources must be devised. For single-processor systems, interference between tasks through shared resources like caches can be dealt with on the scheduling level, by bounding the number of preemptions and calculating a maximal timing penalty. For parallel processors, common resources can potentially be accessed at any time by totally unrelated activities. This renders traditional scheduling theory useless to estimate costs from interference with other tasks running in parallel.

Some ideas how to tackle the research problems are given below:

For flow analysis, there are several possibilities. One is to consider restricted parallel programming models, like Bulk Synchronous Programming, which have been developed in the parallel programming area in order to ease the task of parallel programming. These programming models have simple cost models, which should translate into more predictable timing models. Another possibility may be to use timing analysis to derive a Timed Automaton modelling the parallel code, and use the TA to analyze its synchronization properties. A third possibility is to use information from a parallelizing compiler. Such compilers sometimes use internal representations describing the computation in an abstract way, like an explicit task graph, or a polyhedral index set for sets of loop body executions, which is allocated and scheduled. The compiler then actually has considerable knowledge about where and when different computations are performed, which can be used to help predicting the timing.

For low-level analysis, the necessary hardware modelling should start with a formal specification of the architecture, and should be based on sound methods of abstraction, analysis, and transformation. The attainable accuracy of the models will be critically dependent on the hardware architecture: thus, research is necessary to find suitable MPSoC architectures which are amenable to timing analysis.

The calculation methods will depend on the program execution model. Thus, research to find appropriate such methods will be strongly connected to the flow analysis research.

The common resources issue is a matter of both hardware and system design. As for low-level analysis, research into MPSoC architecture and systems is necessary to reduce the interference between tasks. In particular on-chip networks and memories are crucial components, which have to be designed as to allow predictable timing. A hypothesis is that the ability to dynamically partition the resources, like assigning different parts of the network to different tasks, is helpful in this regard.

In the first 18 months, we foresee the following activities and potential results:

1. Derivation of timing models from MPSoC designs given in a language like Verilog or VHDL.
2. Meetings with researchers in Timed Automata (Modelling & Validation Cluster) to discuss the possible connections between timing analysis on code level and timing analysis on model level. Possible outcome: a report describing one or several combined approaches to the problem of analyzing parallel software with respect to timing properties such as WCET.
3. An investigation whether restricted models for parallel programming can make the problem of WCET analysis easier to solve for programs adhering to these models. Possible outcome: a survey of potentially interesting parallel programming models, with an assessment of their respective amenability to WCET analysis.
4. A joint activity with the MPSoC cluster, where TA expertise is fed back to MPSoC architecture level. Task: to identify features of MPSoC architectures that are critical to the predictability of timing properties, and to suggest possible designs which make the

architectures more predictable with respect to these properties. Evident targets are shared resources like on-chip networks and shared memories. Possible outcome: a report describing the problem and some possible solutions, with their respective pros and cons.

-- Changes wrt Y1 deliverable --

No changes with respect to Year 1.

1.7 Work achieved in Year 1

We studied the problem of timing predictability further. On the level of individual cores, work at Saarland University gave for the first time ever a precise and useful definition of predictability of cache replacement policies, competitiveness and sensitivity. This work can pave the ground for the analysis of cache predictability in the presence of multiple cores and of sensitivity to disturbances from outside influences.

Additionally, work on quantifying the influence of preemptive scheduling on cache contents allows to safely bound effects of preemption on WCET analysis. Furthermore, new work aims at guiding developers to select preemption points which minimise the preemption costs.

Work has started on the elaboration of design principles for architectures with predictable and composable timing behaviour. Various kinds of side effects, interferences that inhibit composability and impair predictability of single- and multi-processor systems have been identified. A small set of mechanisms that rely on pre-planned control to protect the time-relevant state from unforeseen changes has been proposed.

Work has been carried out in the area of measurement-based timing analysis. Such methods can be appropriate when timing behaviour is very complex and hard to model, such as for MPSoC systems, and when the real-time requirements are soft. Problems tackled include the adaptive control-flow segmentation to control the effort of test-case generation, and the use of learning techniques to identify timing model features.

Previous work in parametric WCET analysis has been continued. Parametric WCET analysis calculates a formula for the WCET, in some input parameters, rather than a single number. There are strong relations between the techniques used for this analysis and techniques used in parallelizing compilers, especially the polytope method.

-- No changes wrt Y1 deliverable --

This section was already presented in the Y1 deliverable, in sections 1.7 and 3.1.

1.8 Problem Tackled in Year 2

In Year 2, the partners continued to tackle the problem of poor timing predictability of current and emerging multicore and MPSoC architectures. This problem is complex, and the partners have pursued a number of different approaches to cope with it. The approaches can be classified as follows:

- Design principles for timing-predictable parallel architectures
- Static analysis for timing predictability
- Machine learning, model identification, and test case generation

The first approach includes work on timing-composable hardware/software acrhitectures, where constraints on the design of software and hardware yield systems for which a composable timing analysis of parallel tasks is possible. Furthermore, two technical meetings with the MPSoC design cluster have been held where the issue of timing-predictable MPSoC architectures has been investigated.

Static analysis for timing predictability includes work to better understand and predict the effect of hardware features such as caches. This includes the development of a method to analyze shared caches in multicore systems, a method to bound the context-switch penalties in preemptive systems, and improved microarchitectural analyses. The work on parametric WCET analysis has also continued: recent advances include a method that improves considerably on the complexity of the analysis.

For current multicore/MPSoC systems, a safe static analysis can yield very pessimistic timing estimates. The third approach above is more pragmatic, and yields measurement-based or hybrid methods where models are built from observations. The obtained timing estimates are not safe in general, but the methods are often quite simple to use and tend to give timing estimates closer to the typical case. For applications where some overruns can be accepted, such methods can be appropriate. The partners have worked on methods to identify timing models from observed data, and on methods for test case generation.

Finally, the partners have continued to collaborate on common tool infrastructure and common formats for annotations and analysis results.

-- The above is new material, not present in the Y1 deliverable --

2. Summary of Activity Progress in Year 2

2.1 Technical Achievements

Parametric WCET Analysis (MDH, USaar)

The work on parametric WCET analysis, initiated with USaar, has been carried on further. A new method for parametric WCET analysis has been developed, which uses a symbolic technique for counting the number of states in different program points. A major achievement is a novel symbolic computation method, which for a minor loss of precision gives orders of magnitude of speedup and reduced memory consumption as compared with earlier methods. This breaks a barrier as regards the scalability of the analysis. The method has also been applied in a case study of a robotics control system.

Model Identification for WCET Analysis (MDH)

Hybrid methods for WCET analysis use measured data to estimate the hardware timing effects. Existing methods have relied on a fine-grain instrumentation of the code, which is problematic since the instrumentation causes probe effects and is time-consuming. We have developed a method based on model identification that allows the derivation of the timing model from end-to-end measurements. This reduces the need for instrumentation to a minimum. Furthermore, the identified model will never underestimate any observed execution times. We have also obtained novel theoretical results regarding the test coverage needed to identify an exact timing model.

Using Learning to Support the Development of Embedded Systems (YORK)

The University of York have been engaged in an EPSRC funded project "Using Machine Learning to Support the Development of Embedded Systems", ref: EP/F00334X, since 2007. This supports a post-doctoral researcher and a PhD student. The main thrust of the post-doctoral research is to reflect the issue that much of our current analysis is based on models that may not be available, like for instance detailed knowledge of hardware configurations or the flow of the software including loop bounds. Therefore we have addressed how such information can be inferred from measured data. The papers we have produced show how the approach can be applied to the problems of flow analysis and determining hardware models, e.g., for branch predictors. We have also developed a strategy combining testing and learning that we have proved gives safe and exact results. This is an important result as it means the evidence gained through measurement and learning is at least as safe as that gained from any static analysis approach. In addition we have been investigating how a multi-criteria optimisation approach to the generation of test data can help find the WCET using search. In recent work, we have cooperated with Mälardalen University to consider how parametric WCETs can be derived using the TIMES model checker: a joint paper will appear in early 2010. Future work will consider the scalability of this approach in more depth.

http://www-users.cs.york.ac.uk/~bartlett/rts/

Integrated Instruction and Data cache analysis (NUS)

In this work, we consider the modeling of a generic cache architecture which is most common in commercial processors — separate instruction and data caches in the first level and a unified cache in the second level (which houses code as well as data). Our modeling is used to develop a timing analysis method built on top of the Chronos WCET analysis tool. Moreover we use our unified cache modeling to develop WCET-driven code and data layout optimizations — where the code and data layout are optimized simultaneously for reducing WCET.

Shared cache analysis in multi-cores (NUS)
Multi-core architectures often involve a second level cache shared across cores. In this work, we have developed a timing analysis method for concurrent software running on multi-cores with a shared cache. Communication across tasks is by message passing. Our method progressively improves the lifetime estimates of tasks that execute concurrently on multiple cores.

WCET Analysis in the Presence of Context Switches (USaar, AbsInt, SSSA)
In preemptive systems, not only the WCET, but also the context switch costs need to be bounded. In case of preemption, cache memories may suffer interferences between memory accesses of the preempted and of the preempting task. These interferences lead to some additional reloads that are referred to as cache-related preemption delay (CRPD). This CRPD constitutes the major part of the context switch costs. The original concept to bound the CRPD uses the concept of useful cache blocks (UCB). These are memory blocks that may be in cache before a program point and may be reused after it. In recent work, we tightened the CRPD bound by using a modified notion of UCB: Only cache blocks that are definitely cached are considered useful by our approach. Such blocks are called DC-UCBs. The computed CRPD based on DC-UCBs (when used in combination with the bound on the WCET) delivers a safe bound on the execution time in case of preemption. Experimental results show that our approach provides up to 90% tighter CRPD bounds. A prototype implementation has been integrated into the aiT Timing Analyzer by AbsInt and shipped to Pisa for further collaboration on the analysis of the intertask timing behaviour.

Abstraction of VHDL Processor Models (USaar and AbsInt)
Manual derivation of timing models of hardware on a level suitable for precise and safe timing analysis is a tedious and error-prone process. Because modern processors are synthesised out of formal hardware description languages such as VHDL, in which their behaviour (including the timing) is exactly specified, the timing model could be semi-automatically derived from them. But VHDL models of real world processors are usually very big and complex. Therefore, we have investigated how to derive timing models out of formal VHDL specifications semi-automatically. To this end, we have introduced a two step process: In a first step, we automatically discard all features which do not affect timing properties of the system. Because the size of the resulting model typically does not allow a computationally feasible WCET analysis, in a second step we introduce abstractions of the processor state space iteratively until the state space if small enough. Just like abstractions in program analyses, these abstractions have to be safe. Therefore we build this work on existing techniques of abstract interpretation.

Timing Analysis and Timing Predictability (USaar and AbsInt)
We have improved the microarchitectural analysis in several ways. Whereas classical research work often used LRU replacement policy to achieve good predictability, FIFO replacement is wide-spread in practice. We have invented a generic policy-independent framework for cache analysis that couples may- and must-analyses by means of domain cooperation. With this framework we achieved a precise may-analysis for caches with FIFO replacements, increasing both the predicted hits and misses.

Regarding pipeline behaviour, WCET analysis has to consider many possible hardware states. Classical representations of the pipeline domain enumerate all reachable states explicitly. For modern, complex pipelines, the analysis can become infeasible due to memory and computation time constraints. To alleviate this problem, we have devised a symbolic representation of pipeline states and their transitions as binary decision diagrams. This compact representation leads to significant savings in memory consumption and execution time during WCET analysis.

On the software side, we worked on the derivation and exploitation of operating modes in embedded software. We identified heuristics to derive operating mode candidates from source code, and devised a procedure that exploits mode information to arrive at mode-specific WCET bounds. For the source code analysis, we use the ICD-C compiler framework developed at TU Dortmund. We also worked on improving timing analysis for applications generated from high-level Simulink models. Using relations between variables arising from the behaviour of Stateflow automata, we can derive infeasibility constraints on the control-flow graph. This work is at an early experimental stage. Furthermore, we developed two methods to deal with the additional uncertainty of cache effects arising from dynamic memory allocation. In a first approach, we have developed a predictable dynamic memory allocator. In a second approach, we developed a tool to automatically transform dynamic allocation into static allocation with comparable memory consumption. For both approaches, the necessary information about the dynamically allocated memory is derived by an adapted shape analysis together with appropriate abstraction techniques.

Timing Analysis and Timing Composability (TU Vienna)

As the complexity of real-time hardware and software is increasing, a central concern in real-time systems development is the lack of sufficient methods and tools to effectively reason about the timing of these systems. To alleviate this problem it is important that software systems can be constructed hierarchically from components while still guaranteeing the timing properties. We therefore discussed the deficiencies in current real-time embedded hardware and software structures with respect to achieving our goal of composable and compositional timing behavior. To address these deficiencies, we identified the needs for programming methods, code generation techniques, and ideas about hardware and software architectures that should help us in achieving a truly timing-composable and compositional engineering process for real-time software systems. As a feasibility study, we implemented a sample application on a chip-multiprocessor system. The software of this application was implemented in single-path code. Resource conflicts were avoided by time-sliced arbitration for all shared resources.

Test-Case Generation for WCET Analysis (TU Vienna, York, MDH)

Within this activity, the efficient coverage of program models for measurement-based timing analysis is explored. One of the main issues covered this year was the segmentation of control flow to tune the effort for test-case generation and execution time measurements, which was published in a paper (Zolda, Bünte, Kirner). Further, research is going on for the combination of vertical and horizontal control-flow segmentation techniques to speed up the learning rate of the timing model by execution time measurements. Another achievement was a novel method, based on static analysis methods, to find an input vector that causes a piece of code to execute its longest path.

A Common Annotation Language for WCET Analysis (TU Vienna, AbsInt, MDH)

This is an ongoing activity of the timing analysis community to define a common annotation language for worst-case execution time analysis. This annotation language is meant to be flexible enough to include manually provided descriptions of the program behavior as well as automatically generated annotations. Within this year two journal articles have been submitted, one giving an overview of existing annotation languages for timing analysis, and the other presenting essential ingredients for a universal common annotation language for WCET analysis.

Integration of Timing Analysis and Compilation (TU Dortmund, AbsInt)
The work on Dortmund's Worst-Case Execution Time-aware C Compiler WCC has been continued in ArtistDesign Year 2.

Related to the compiler's core infrastructure, a sophisticated fully automatic loop analyzer based on polyhedral models has been added to WCC. It is able to determine loop iteration counts and flow facts required for WCET analysis for broad classes of loops. Furthermore, a Back-Annotation module is able to map WCET-related timing data from low-level assembly code back to high-level C code such that a WCET timing model now is available a C code level.

On top of the WCC infrastructure, several compiler optimizations known from standard literature have been made WCET-aware. This includes Function Inlining, Loop Unswitching and Loop Unrolling. Since the impact of many of these optimizations depends on complex characteristics of the entire application currently under compilation, and of the underlying hardware, machine learning techniques steering these optimizations have been integrated into WCC.

Finally, WCC is currently extended to support code generation and optimization for multiple tasks of multi-process applications. In this area, a multi-task cache partitioning optimization has been developed.

Overall, the WCC compiler can be considered the leading WCET-aware compiler. The integrated tool set allows studying the impact of optimizations for WCET minimization.

-- The above is new material, not present in the Y1 deliverable --

2.2 Individual Publications Resulting from these Achievements

TU Dortmund

P. Lokuciejewski, F. Gedikli, P. Marwedel and K. Morik. *Automatic WCET Reduction by Machine Learning Based Heuristics for Function Inlining*. Proceedings of SMART '09: 3rd Workshop on Statistical and Machine Learning Approaches to Architectures and Compilation, January 2009, pp. 1-15.

P. Lokuciejewski, D. Cordes, H. Falk and P. Marwedel. *A Fast and Precise Static Loop Analysis based on Abstract Interpretation, Program Slicing and Polytope Models*. Proceedings of CGO '09: International Symposium on Code Generation and Optimization, March 2009, pp. 136-146.

P. Lokuciejewski, F. Gedikli and P. Marwedel. *Accelerating WCET-driven Optimizations by the Invariant Path – a Case Study of Loop Unswitching*. Proceeding of SCOPES '09: 12th International Workshop on Software & Compilers for Embedded Systems, April 2009, pp. 11-20.

S. Plazar, P. Lokuciejewski and P. Marwedel. *WCET-aware Software Based Cache Partitioning for Multi-Task Real-Time Systems*. Proceedings of WCET '09: 9th International Workshop on Worst-Case Execution Time Analysis, June 2009, pp. 84-94.

P. Lokuciejewski and P. Marwedel. *Combining Worst-Case Timing Models, Loop Unrolling, and Static Loop Analysis for WCET Minimization*. Proceedings of ECRTS '09: 21st Euromicro Conference on Real-Time Systems, July 2009, pp. 35-44.

MDH
Stefan Bygde, Andreas Ermedahl, and Björn Lisper. An Efficient Algorithm for Parametric WCET Calculation. in Patrick Kellenberger, ed. *Proc. 16th International Conference on Real-Time Computing Systems and Applications (RTCSA'09)*, pages 13-21. Beijing, China, Aug 2009. **Best paper award.**

Andreas Ermedahl, Johan Fredriksson, Jan Gustafsson, and Peter Altenbernd. Deriving the Worst-Case Execution Time Input Values. *Proc. 21st Euromicro Conference of Real-Time Systems (ECRTS'09)*, pp. 45-54, Dublin, Ireland, July 2009

Björn Lisper and Marcelo Santos. Model Identification for WCET Analysis. *Proc. 15th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS 2009)*, IEEE, San Francisco, CA, April 2009

Yue Lu, Antonio Cicchetti, Mikael Sjödin, Jukka Mäki-Turja, Stefan Bygde, and Christer Norström, Towards Response-Time Analysis of Complex Real-Time Systems by using Parametric Worst-Case Execution-Time Estimate on Tasks – A Case Study for Robotic Control Systems. *Proc. 21st Euromicro Conference on Real-Time Systems (ECRTS 09)*, Work-In-Progress (WIP) session, Dublin, Ireland, July 2009

NUS
Sudipta Chattopadhyay and Abhik Roychoudhury. Unified Cache Modeling for WCET Analysis and Layout Optimizations. *IEEE Real-time System Symposium (RTSS)* 2009.

Yan Li, Vivy Suhendra, Yun Liang, Tulika Mitra and Abhik Roychoudhury. Timing Analysis of Concurrent Programs Running on Shared Cache Multi-cores. *IEEE Real-time System Symposium (RTSS)* 2009.

Abhik Roychoudhury. *Embedded Systems and Software Validation*. Elsevier (formerly Morgan Kaufmann), Systems on Silicon series, 2009.

Samarjit Chakraborty, Tulika Mitra, Abhik Roychoudhury, and Lothar Thiele. Cache-aware Timing Analysis of Streaming Applications. Real-time Systems Journal, 41(1), 2009. (Special issue for selected papers from ECRTS 2007)

USAAR
J. Herter and J. Reineke. *Making dynamic memory allocation static to support WCET analyses*. Proceedings of the 9th International Workshop on Worst-Case Execution Time (WCET) Analysis, June 2009.

S. Altmeyer and C. Burguière. *A new notion of useful cache block to improve the bounds of cache-related preemption delay*. Proceedings of the 21st Euromicro Conference on Real-Time Systems (ECRTS), July 2009.

D. Grund and J. Reineke. *Abstract interpretation of FIFO replacement*. In Proceedings of the 16th International Symposium on Static Analysis (SAS), August 2009

P. Lucas, O. Parshin, and R. Wilhelm. *Operating mode specific WCET analysis*. Proceedings of the 3rd Junior Researcher Workshop on Real-Time Computing (JRWRTC), October 2009.

L. Tan, B. Wachter, P. Lucas and R. Wilhelm. *Improving timing analysis for Matlab Simulink/Stateflow*. In Proceedings of the 2nd International Workshop on Model Based Architecting and Construction of Embedded Systems (ACES-MB), October 2009.

VIENNA

Peter Puschner, Raimund Kirner, and Robert G. Pettit. Towards Composable Timing for Real-Time Software. In *Proc. 1st International Workshop on Software Technologies for Future Dependable Distributed Systems*, IEEE Computer Society CPS, p. 1-5, 2009.

Martin Schoeberl, Peter Puschner, and Raimund Kirner. . A Single-Path Chip-Multiprocessor System. In *Software Technologies for Embedded and Ubiquitous Systems, 7th IFIP WG 10.2 International Workshop*, Springer, LNCS, 2009.

M. Zolda, S. Bünte, R. Kirner. Towards Adaptable Control Flow Segmentation for Measurement-Based Execution Time Analysis, *Proceedings ofthe 17th International Conference on Real-Time and Network Systems (RNTS)*, 2009

A. Prantl, J. Knoop, R. Kirner, A. Kadlec, M. Schordan. From Trusted Annotations to Verified Knowledge. *Proceedings of the 9th International Workshop on Worst-Case Execution Time Analysis (WCET)*, 2009.

YORK

M. Bartlett, I. Bate, D. Kazakov, Guaranteed Loop Bound Identification from Program Traces for WCET, *Proceedings of the 15th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pp. 287-294, 2009.

U. Khan, I. Bate, WCET Analysis of Modern Processors Using Multi-Criteria Optimisation, *Proceedings of the 1st International Symposium on Search Based Software Engineering (SSBSE '09)*, pp. 103-112, 2009.

-- The above are new references, not present in the Y1 deliverable --

2.3 Interaction and Building Excellence between Partners

Interaction between MDH and AbsInt:

AbsInt and Mälardalen are cooperating on methods for early stage timing analysis, and works together to interface the advanced program flow analysis methods of Mälardalen's WCET analysis tool SWEET with the commercial WCET analysis tool aiT from AbsInt. This work continues the tool integration work undertaken in Artist2.

Interaction between Saarland, Dortmund and AbsInt

For their mode analysis on C code, Saarland University uses the ICD-C compiler infrastructure developed at TU Dortmund. That infrastructure also is part of Dortmund's WCET-aware C Compiler (WCC), which integrates AbsInt's aiT timing analysis tool.

Interaction between TU Vienna, MDH, and AbsInt:

TU Vienna, MDH, and AbsInt work jointly on flow-fact information categorization and flow-fact description languages for WCET analysis.

Interaction between MDH and Tidorum:

A cooperation has been initiated in the area of bit-precise abstract domains for program flow analysis. Such domains are necessary to accurately model the control flow behaviour for embedded software, in particular on processors with short wordlength.

-- Changes wrt Y1 deliverable --

The interactions between MDH and AbsInt; TU Vienna, MDH, and AbsInt; and MDH and Tidorum are novel to the Y2 deliverable.

2.4 Joint Publications Resulting from these Achievements

R. Wilhelm, D. Grund, J. Reineke, M. Schlickling, M. Pister and C.Ferdinand: *Memory Hierarchies, Pipelines, and Buses for Future Architectures in Time-Critical Embedded Systems*, IEEE TCAD, July 2009

D. Grund, J. Reineke, and G. Gebhard. *Branch target buffers: WCET analysis and timing predictability*. In Proceedings of the 15th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA), August 2009.

S. Wilhelm and B. Wachter. *Symbolic state traversal for WCET analysis*. Proceedings of the International Conference on Embedded Software (EMSOFT), October 2009.

M. Abdel Maksoud, M. Pister, and M. Schlickling. *An Abstraction-Aware Compiler for VHDL Models*. In Proceedings of the International Conference on Computer Engineering and Systems, December 2009.

-- The above are new references, not present in the Y1 deliverable --

2.5 Keynotes, Workshops, Tutorials

Invited Talk: Predictable Multi-Cores
Verimag – February 13, 2009
Reinhard Wilhelm was invited to give this talk at a local colloquium at Verimag to explain the PROMPT design principles on predictable multi-core architectures.

Keynote: The PROMPT Design Principles for Predictable Multi-Core Architectures
SCOPES
Nice – April 24, 2009
Reinhard Wilhelm gave at keynote at the SCOPES conference on the PROMPT design principles invented in the PREDATOR project. These principles are designed to facilitate the

development of multi-core architectures to with efficiently predictable good worst-case performance. The talk presented the development process and its connections to established industrial standards and trends such as Integrated Modular Avionics (IMA) and the Automotive Open System Architecture (AUTOSAR).
< http://www.scopesconf.org/scopes-09/keynote.html>

ARTIST Summer School in Europe 2009
Autrans – September 7-11, 2009
The Artist Summer School included invited talks by several researchers, including Luca Benini, Jan Beutel, Jan Reineke, Lothar Thiele and Reinhard Wilhelm.
<http://www.artist-embedded.org/artist/Overview,1633.html>

Workshop: Shape Analysis, Timing Analysis
PUMA Workshop
San Servolo Island,Venice – Oct. 5-6, 2009
Reinhard Wilhelm gave talks on Shape Analysis and on Timing Analysis at the PUMA workshop. This workshop is an annual event held by the PUMA Graduate School of LMU (Ludwig-Maxmilians-Universität, Munich) and TU Munich.
< http://puma.in.tum.de/wiki/Venice_2009>

Keynote: Embedded Systems - Trends, Successes, Challenges
10th Anniversary of the Hasso-Plattner Institute
Potsdam – Nov. 18, 2009
To celebrate its 10th anniversary, the Hasso-Plattner-Instiitute holds a conference "Informatik-Impulse". Reinhard Wilhelm is invited to present an overview of the challenges in embedded systems world and to give an outlook onto future developments.

Keynote: Timing Analysis and Timing Predictability
Tag der Informatik
RWTH Aachen – December 4, 2009
Reinhard Wilhelm is invited to give a talk introducing timing analysis and timing predictability in embedded systems. The current challenges and existing timing analysis algorithms will be discussed as well as the additional challenges posed by multi-core systems and approaches to achieve predictability for them.
< http://www.nets.rwth-aachen.de/content/current_events/tdi/pro/index.html>

Keynote : From Performance to Time-Predictability
9th Architectures and Compilers for Embedded Systems (ACES) Symposium
Edegem, Belgium – September 7-8, 2009

This keynote was given by Peter Puschner (TU Vienna). It outlined the problems of building predictable hardware/software systems and discussed strategies for constructing systems that provide both temporal predictability and performance.
http://www.elis.ugent.be/aces/index.php?page=activities

Invited Talk : Timing Analysis of Real-Time Software
Workshop on Quantitative Analysis of Software
Grenoble, France – June 28, 2009

This invited presentation was given by Raimund Kirner. It gave insights into the principles of worst-case execution time analysis and explained current challenges, including transformation of flow information and hardware modelling in the presence of timing anomalies. Further, our research on measurement-based timing analysis was presented.

http://www.eecs.berkeley.edu/~sseshia/qa09/

Workshop : 9th Int'l Workshop on Worst-Case Execution Time Analysis (WCET'09)
Dublin, Ireland – June 30th, 2009
On June 30, 2009, thirty-five people from nine countries and three continents met in Trinity College, Dublin, to hold the 9th International Workshop on Worst-Case Execution Time Analysis (WCET'09, http://www.artist-embedded.org/artist/WCET-2009.html). The workshop was organised as a satellite event of the 21st Euromicro Conference on Real-Time Systems (ECRTS'09, http://ecrts09.dsg.cs.tcd.ie). ArtistDesign supported the workshop by paying the travel costs of the invited speaker, Prof. Petru Eles (Linköping) and the workshop chair, Dr. Niklas Holsti (Tidorum Ltd),. and the costs of printing and distributing the proceedings ("Worst-Case Execution Time Analysis", ISBN 978-3-85403-252-6, books@ocg.at, Volume 252). ArtistDesign participants presented several of the workshop papers.

Tutorial : Multicore and Hard Real-Time Systems
Swedish Multicore Day
Kista, Sweden – Sept 4, 2009
This tutorial was given by Björn Lisper, and was primarily directed to practitioners in industry. It described the inherent problems with timing predictability for conventional multicore architectures, and gave an account for ongoing research in the area.
http://www.sics.se/multicoredays

Tutorial : WCET Analysis: Problems, Methods and Time-Predictable Architectures
Acaces 2009. Fifth International Summer School on Advanced Computer Architecture and Compilation for Embedded Systems
Terrassa (near Barcelona), Spain – July 12-18, 2009

The annual ACACES summer school is organized by the European Network of Excellence on High Performance and Embedded Architecture and Compilation (HiPEAC). The invited tutorial on WCET analysis and time-predictable hardware/software architectures was given by Peter Puschner (TU Vienna).
http://www.hipeac.net/acaces2009/

-- The above is new material, not present in the Y1 deliverable --

3. Milestones, and Future Evolution

3.1 Problem to be Tackled over the next 12 months (Jan 2010 – Dec 2010)

In the next 12 months, we plan to develop timing models of shared resources in multi-cores, namely shared cache and shared bus. In particular, we plan to build on the shared cache analysis of NUS to develop an integrated timing analysis framework which captures the timing interactions between shared cache and shared bus accesses. To validate the results of the analysis, we have also developed a cycle-accurate simulation infra-structure for multi-core platforms with shared cache and shared bus.

The design of an architecture with predictable behaviour will be further pursued in the PREDATOR project. This architecture will be a multi-core architecture following the principle of de-sharing, i.e., of keeping non-shared parts of applications non-shared when mapped to the target architecture. In the next 12 months, predictability requirements for the scheduling level will be considered. In particular, we will continue to consider the effects of pre-emptive scheduling on the worst-case execution time of tasks as well as means to decrease that effect. A cache-aware memory allocation can minimise the disturbance of caches by preemption, and a cache-aware scheduler can exploit knowledge about maximal disturbance to bound the WCET of tasks more precisely.

We will continue the research on mechanisms that support the timing analysis in a hierarchical development processes for contemporary embedded system architectures. The goal of this investigation is to work out software structures, hardware mechanisms, and appropriate strategies for the use and control of these hardware mechanisms to provide a system timing behaviour that is both predictable and composable. Within the next period the focus of our investigations will be on options of maintaining predictability in the presence of hierarchical memory architectures.

Test-case generation for measurement-based WCET analysis of complex hardware architectures will be further explored. The focus in this area is on new methods for the segmentation of control-flow graphs in order to ease the test-case generation and execution-time measurements. Within the next period we plan to develop strategies for combining vertical and horizontal control-flow segmentation techniques.

We will initiate research on timing analysis of explicitly parallel programs. This activity was planned to start during 2009, but will not start until 2010 due to a delay in the recruitment of a PhD student who will work on the problem.

-- Changes wrt Y1 deliverable --

The planned work on timing models for shared caches and buses, cache-aware memory allocation, and cache-aware scheduling is new. In addition to the work on static analysis and timing predictability, work is now planned in the area of test- and measurement-based timing analysis methods.

3.2 Current and Future Milestones

- **Year 1: PROMPT design principles**
  The PROMPT design principles were developed.

  1. The PROMPT design principles for predictable architectures were developed, based on the principles of de-sharing.
  2. The design principles were developed based on some intuitive assumptions derived from experience with developing timing analyzers for several architectures. These intuitive assumptions were formally defined and proved for the case of cache architectures. For other architectural components we still lack the underlying theory.
  3. The PROMPT architectural design principles will be applied to the design of the PREDATOR multi-core architecture resulting in the first version this design.

     100% achieved. The design principles have been developed.

- **Year 1: Timing Side Effects**
  A clean design and analysis process of embedded systems timing is strongly simplified if the temporal planning and the timing analysis can be performed in a hierarchical way, e.g., in a separation of WCET and schedulability analysis, of single-core analysis from the analysis of the whole multi-core system. Unfortunately many features found in contemporary hardware and software cause phenomena that inhibit a hierarchical decomposition of the planning and analysis process. It is our goal to investigate these phenomena and name the unwanted side effects they have on timing. The results will be used to work out alternative hardware and software structures that provide better support to a hierarchical design and analysis of worst-case timing.
  100% achieved. We have identified various types of side effects, ranging from data-dependent instruction execution times and data paths, over processor-state dependencies between different invocations of a single task, to the side effects that the execution of a task has on the timing of other tasks, where other tasks may reside on the same processor (in a single or multi-processor system) or a different processor of a multi-processor system. The findings of this investigation have been published and presented at the Workshop on Worst-Case Execution-Time Analysis, WCET2008.

- **Year 2: Composable Software/Hardware Architectures**
  Temporal composability is essential for making the factor time an integral part of a meaningful hierarchical development process for embedded real-time applications. Our goal is to work out software structures, hardware mechanisms, and appropriate strategies for the use and control of these hardware mechanisms that support temporal composability.
  50% achieved. In year 1, we started towards this goal, and some ideas on how to obtain composability have been published. In year 2, the topic has been further explored. We investigated the prerequisites for a hierarchical software construction process that provides temporal composability in more detail. A simple example of an application was implemented to show that the use of adequate hardware and software models can provide the framework that avoids temporal interferences between tasks and supports temporal composability.

- **Year 3: Investigation how to extend temporal composability to hardware architectures with hierarchical memory architectures.**

- **Year 2: Initial Timing Model for Parallel Programs**
  We plan to define a formal timing model for explicitly parallel programs. This model will serve as a "reference point", and the correctness of forthcoming timing analyses for such programs will be possible to verify with respect to this model.
  0% achieved. This activity did not get started as planned, due to a delay in the recruitment of a PhD student (now solved) who will work on the problem. The milestone is deferred to Year 3.
  Year 3: Initial Timing Model for Parallel Programs

- **Year 3: Measurement-based WCET analysis**
  Generating the right test data to get a good coverage of relevant test cases is a core issue in measurement-based WCET analysis for complex computer architectures. We will explore methods for generating test cases that provide a good coverage by investigating methods for partitioning and analyzing control-flow graphs.
  This milestone was not present in year 2. However, in year 2 we investigated the segmentation of control-flow graphs to reduce the effort for test generation and measurements for the WCET analysis. In year 3, the combination of vertical and horizontal control-flow segmentation techniques will be explored, with the goal to speed up the learning rate of the timing model by execution-time measurements.

3.3 Main Funding

The ArtistDesign NoE funds integration and building excellence with the partners, and with the European research landscape as a whole. Beyond this "glue" for integration and excellence, during Year2 this activity has benefited from direct funding from:

- **AVACS**
  The AVACS project *(Automatic Verification and Analysis of Complex Systems)* is funded by the DFG and supports Saarland University.
  http://avacs.org
- **PREDATOR**
  The PREDATOR project *(Design for Predictability and Efficiency)* is a research project within the European Commission's 7th Framework Programme on Research, Technological Development and Demonstration. Bologna, Pisa, ETHZ, Saarland University and Dortmund take part in PREDATOR, along with AbsInt, Airbus and Bosch. The goal of PREDATOR is to reconcile performance and predictability, in particular for future multi-core architectures.
  http://predator-project.eu
- **COSTA**
  The COSTA project (*Compiler-Support for Timing Analysis*) is funded by the Austrian Science Fund (FWF) and supports TU Vienna. The project started in July 2006.
  http://costa.tuwien.ac.at
- **FORTAS**
  The FORTAS project (Formales Zeitanalyseframework für Echtzeitsysteme) is a cooperative project that brings together the orthogonal expertise of the TU Vienna real-time systems group and the group on formal methods and systems engineering from TU Darmstadt. Within FORTAS TU Vienna is supported by the Austrian Science Fund (FWF). FORTAS started in January 2007.

  http://www.fortastic.net

- Mälardalen has funding from the Foundation for Strategic Research (**PROGRESS** Strategic Research Centre, duration 2006-2010), the FP7 STREP **ALL-TIMES** (Integrating European Timing Analysis Technology) 2007-2010, and the Swedish Research Council "Worst-Case Execution Time Analysis of Parallel Systems", 2009-2011.
  http://www.mrtc.mdh.se/progress/
  http://www.all-times.org
- The University of York has funding from the EPSRC project **Using Machine Learning to Support the Development of Embedded Systems**, ref: EP/F00334X, since 2007.

-- Changes wrt Y1 deliverable --

The description of the funding for Mälardalen was updated to reflect the current situation. The funding for York was added to the Y2 deliverable.

4. Internal Reviewers for this Deliverable

- **Niklas Holsti,** Tidorum Ltd
- **Damir Isovic**, MDH