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

Activity Progress Report for Year 4

Timing Analysis

Cluster:

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	Feb 1st 2012

Table of Contents

1. Overview of the Activity (2008-2011)

1.1 ArtistDesign participants and their role within the Activity

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, Compilation with Support for Timing Analysis, Measurement-Based Timing Analysis
Within the Timing-Analysis Activity TU Vienna focuses on WCET analysis, compilation and on hardware and software architectures that provide time-predictability and composability. Strategies for measurement-based WCET analysis are explored.

Prof. Dr. Peter Marwedel – TU Dortmund (Germany)
Architecture-Aware Compilation, Low-Power Code Generation, Development of Optimizations for WCET Minimization.

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

Research on Scheduling and Timing Analysis in the presence of interrupts

-- Changes wrt Y3 deliverable --

Dr. Raimund Kirner has left TU Vienna. His activities have been taken over by Peter Puschner.

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; improvement and extension of UCB analysis.

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 Y3 deliverable --

No changes.

1.3 Starting Date, and Expected Ending Date

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

-- Changes wrt Y3 deliverable --

No changes.

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 Y3 deliverable --

No changes.

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, first executed in 2006, 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 Y3 deliverable --

No changes.

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. Some research problems are:

- 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.

-- Changes wrt Y3 deliverable --

The statement about the foreseen activities for the first 18 months has been removed, since these 18 months have since long passed.

2. Work Achieved in the NoE

2.1 Synthesis View of the Main Overall Achievements

The achievements of the activity can be grouped into four main categories: **timing predictability**, **measurement-based/hybrid timing analysis**, **tool interfacing and tool infrastructure**, and **promotion of timing analysis research community activities**.

The achievements regarding timing predictability can, in turn, be divided into three groups:

- *foundational issues*,
- *design principles for timing-predictable systems*, and
- *advanced analysis methods* that can predict the timing of complex systems accurately, including methods to improve the predictability.

We can furthermore divide the achievements into results for *multicore/parallel architectures*, and results for *single-core architectures*.

Timing predictability

On the foundational side, we have defined notions of predictability: first for cache replacement policies, and later as a general template for predictability definitions. Such definitions are necessary if one is to use the concept predictability in a scientifically rigorous way.

A number of important results have been achieved regarding timing analysis and -predictability of multicore architectures. Important design principles for predictable multicore architectures have been devised: these principles build on the concept of *de-sharing*, meaning that activities on different cores should be localized as far as possible. Remaining shared resources, such as shared data, buses, and shared memories, should be put under strict scheduling control. This achieves temporal isolation, and thus high predictability. A totally different design principle, which builds on randomization, has been the target of an initial investigation. By independent randomization, predictability can be regained as an *emergent* property.

The issue of timing-composable hardware/software architectures has been thoroughly investigated. Timing composability aids timing predictability by allowing analysis results of system parts to be easily composed for the full system. Results apply to both multicore and single-core architectures: they concern memory hierarchies, code transformations to avoid timing anomalies, and coding policies for timing predictability.

Important achievements have been made regarding WCET analysis of standard shared memory multicore architectures. This is a hard topic, since these architectures have poor timing composability: thus, an analysis must consider the full system with all ongoing activities. First results concerned a shared cache analysis for multicores. Later, this has been extended into a unified WCET analysis for multicores taking more advanced architectural features into account.

The aforementioned analysis concerns single-core tasks executing on a multicore. In the future computationally demanding tasks will be executed as parallel programs, on several cores. We have investigated how to perform WCET analysis for such programs. An initial approach was based on model checking over timed automata: later, an analysis based on abstract interpretation has been outlined.

For single-core architectures, we have improved considerably on different analyses as well as methods to improve the timing predictability. Several novel cache analyses have emerged: analyses for replacement policies occurring in practice such as FIFO and PLRU, and precise and scalable cache analyses that are based on a combination of abstract interpretation and

model checking. We have also obtained interesting results regarding predictability of replacement policies, which show that for some policies the initial cache state can have a lasting influence on the cache contents and thus the timing. Furthermore, improved analyses for cache-related preemption delay have been developed, as well as methods for cache-aware memory allocation.

A method for parametric WCET analysis has been developed. This method utilizes several advanced symbolic analysis techniques. For such techniques, scalability is an issue: a method to improve the scalability of the so-called calculation phase, with limited loss of precision, has been developed. Lately, the analysis has been modified to be sound in the presence of arithmetic overflows and wraparounds. Such situations appear frequently in code running on small embedded processors with short word length.

A key achievement is the integration of timing analysis and compilers. In cooperation with AbsInt and Saarland University, TU Dortmund has implemented the WCET-aware compiler WCC. WCC incorporates a tight integration of timing analysis into compiler optimizations. The potential for making standard optimizations WCET-aware has been explored in depth. It turned out that the largest potential is in exploiting the memory hierarchy. It was demonstrated that scratchpads offer a large potential for WCET-improvements, but even WCET-aware register allocation can contribute toward WCET-efficiency. Recent extensions include code generation beyond the TriCore architecture, the support of multi-processing and multi-processors as well as multiple objectives.

Measurement-based/hybrid timing analysis

Timing predictability is of utmost importance for time- and safety-critical systems where both safe and tight timing estimates are required. For soft real-time systems, or for early timing estimation guiding system design, one can make do with unsafe estimates. Such estimates can be provided by measurement-based or hybrid timing analyses. This is especially interesting for off-the-shelf multicores, which exhibit poor timing predictability and for which a safe timing analysis easily can give very large overestimations.

One issue that has been tackled is test vector selection for measurement-based timing analysis. Approaches based on different coverage criteria have been investigated. The problem of reducing the test vector set, for streaming applications with huge data sets, has also been attacked, with promising results.

Another strand of research is the detection of timing models, and program flow constraints, from measurements. Several approaches have been investigated, from classical model identification to machine learning methods. An application is to identify approximate timing models for source code; this enables a rough source-level WCET estimation by static analysis techniques early in the design process, even before the full code compiles. Initial experiments indicate that deviations from the real WCET in the range 0-30% can be achieved.

Tool interfacing and tool infrastructure

Partners have cooperated around tool interfacing and tool infrastructure. The aforementioned work on WCET-aware compilation was made possible by an integration of WCC with aiT. Work has been initiated towards an open platform for timing analysis research, based on an open compiler framework and an open source processor. The WCET analysis tool SWEET from MDH has been interfaced in different ways with aiT, the compiler framework LLVM, and the WCET analysis tool Bound-T from Tidorum in order to give them access to the powerful program flow analysis of SWEET.

Promotion of timing analysis research community activities

Finally, ArtistDesign has supported the *WCET Workshop* and the *WCET Challenge*. The workshop, held annually as a satellite workshop to ECRTS, has become the major event of the timing analysis research community attracting increasing numbers of submissions and

attendees. ArtistDesign has sponsored the proceedings, and the invited speakers. The WCET Challenge is a bi-annual event held for the first time in 2006, and again 2008 and 2010/11. The purpose of this event is to provide an arena where different timing analysis tools can be evaluated, with respect to different criteria, compared, and showcased. For each instance of the Challenge a set of benchmark codes, target processors, and problems to be solved have been defined. Each participating tool has then attacked a selection of problems, for some target processor(s). The collected results have been reported, and published on the website of the Challenge. For each instance of the Challenge more demanding benchmark codes and problems have been used, including real-time production code. ArtistDesign has supported the work with evaluating the tools.

-- The above is new material, not present in the Y3 deliverable --

2.2 ***Work achieved in Year 1*** *(Jan-Dec 2008)*

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 Y3 deliverable --

This section was already presented in the Y3 deliverable, in section 1.7.

2.3 ***Work achieved in Year 2*** *(Jan-Dec 2009)*

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 architectures, 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.

-- No changes wrt Y3 deliverable --

This section was already presented in the Y3 deliverable, in section 1.8.

2.4 ***Work achieved in Year 3*** *(Jan-Dec 2010)*

Within the PREDATOR project, we continued the advances in various fields. The work on foundations of predictability has found success in the proposal of an adaptable template for predictability definitions. We also continued the work of timing analysis of cache replacement strategies, tackling FIFO and PLRU replacement strategies.

Timing analysis of tasks under preemptive scheduling was also investigated further. A new approach based on resilience further improved the bounds on cache-related pre-emption delay context-switch costs for preempted tasks.

Experiments have shown the cache-aware memory allocator to work successfully in practice, and our algorithm for transforming dynamic into static allocation has been extended as well. Both approaches allow for timing analysis of tasks with dynamic memory allocation.

Concerning the goal of reconciling timing analysis with compilation, the WCET-aware compiler WCC developed at TU Dortmund has matured in the past 12 months. WCC is currently evaluated at Bosch and is now able to generate and optimize industrial code representing an engine control system. First results indicate that WCC produces code with a WCET of a factor of 2 below that of the GCC compiler. In ArtistDesign year 3, WCC has been extended towards code generation and optimization for multi-process systems. This included porting a real-time

operating system to the TriCore architecture mainly supported by WCC. In ArtistDesign year 3, the collaboration between TU Dortmund, Saarbrücken, AbsInt, ETHZ and Pisa has been strengthened significantly. Furthermore, cooperation between TU Dortmund and NUS Singapore led to a joint effort for bus-aware timing analysis for multi-core systems.

TU Dortmund's work on WCET-aware compilation has been awarded several times during ArtistDesign year 3. Timon Kelters master's thesis on WCET-aware superblock formation and optimization has been awarded three times. It received the Hans-Uhde award and was awarded as best thesis in the field of real-time systems by the German Society of Informatics (GI) and as nationally best computer science thesis by Germany's congregation of computer science faculties. Furthermore, Dortmund's former group member Paul Lokuciejewski received TU Dortmund's dissertation award.

The work on parametric WCET analysis has continued. A main problem to be tackled is the complexity of the analysis, and how to find the best tradeoffs between efficiency and precision. During Year 3 the partners developed new, more efficient methods for the symbolic calculation phase, as well as for symbolic path (program flow) analysis. The use of parametric WCET estimates to obtain a more precise schedulability analysis has also been studied.

The problem of analyzing explicitly parallel programs with respect to execution time has been the focus of an initial study. Such programs cannot be analyzed using methods and models for purely sequential tasks. Real-time scheduling theory for parallel systems is not applicable either, since parallel programs may synchronize in a much more general manner than assumed by this theory.

Several partners have worked on the issue of timing predictability. Besides the work on timing analysis of cache replacement strategies mentioned above, we have studied how to use code transformations to avoid the unpredictability caused by timing anomalies. We have also introduced the concept of timing barrier. The work on coding policies for timing predictability has been forwarded by extending the allowed coding policy to partially input data-dependent programs. The results indicate that the more relaxed policy still allows a fully automated loop bounds analysis. Finally the problem of mapping source-level program flow annotations to binary level, in the presence of code optimizations, has been attacked by developing an approach where the annotations are transformed in parallel with the code.

The problem of early WCET analysis has been attacked. Early WCET approximations are useful to direct the system design work in order to avoid costly system redesigns at late stage due to timing and resource bottlenecks. A tool chain combining a fast and approximate WCET analysis with a source-level program flow analysis has been implemented.

The work on hybrid- and measurement-based approaches has been continued. An important part is test-case generation, where advances have been made regarding suitable coverage criteria as well as for search-based approaches to find representative trace samples. We have also applied learning techniques to approximate cache modelling from test data.

-- No changes wrt Y3 deliverable --

This section was already presented in the Y3 deliverable, in sections 1.8 and 3.1.

2.5 ***Work achieved in Year 4*** *(Jan-Dec 2011)*

The central issue of timing predictability has been further studied in Y4. Foundational work has been made regarding the formal definition of predictability. On the practical side earlier developed cache analyses, for replacement policies such as FIFO and PLRU, are now being commercialized in the aiT WCET analysis tool from AbsInt. The cache-aware memory

allocation and the cache-related preemption delay (CRPD) analysis, both developed in Y3, have been refined and improved. Furthermore, a scalable and precise cache analysis based on a combination of abstract interpretation and model checking has been developed. We have also seen advances in WCET analysis for multicores, with a closer modelling of the microarchitecture lifting the previous restriction to timing anomaly-free cores.

Timing composability is a strong enabler of timing predictability. The work on timing-composable software/hardware architectures has continued in Y4. We have compared a dynamic and a static strategy for prefetching memory contents to local memory. The evaluation showed the dynamic strategy's inferiority due to the impossibility of predicting the behaviour of conditional branches.

The T-Crest project (Time Predictable Multi-Core Architecture for Embedded Systems), started in October 2011, will further develop architecture and compiler concepts to support time predictability of applications and simplify WCET and overall timing analysis.

An alternative approach is to consider predictability as an emergent behaviour. If system components exhibit independent, random behaviour, then the system will be predictable with very high probability. An initial study of this approach has been made.

The work at TU Dortmund considering WCET-aware basic block reordering has been finished. Unconditional branches are avoided and the prediction of conditional branches is supported by the developed techniques. A genetic approach applies evolutionary algorithms considering the WCET of the program to optimize as fitness value with the costs of high optimization times. Thus, an integer-linear programming-based approach has been developed which determines the optimal order of basic blocks and also takes the branch prediction into account [PKFM11]. Furthermore, WCET-aware cache locking and code positioning has been improved. The integration and enhancement of a framework for the static analysis of software and hardware as announced in last year's report has been advanced. The extension of WCC's native analysis capabilities allows for novel approaches especially in the domain of multitask- and multicore-aware compilation. A much higher degree of control over system states directly affected by optimization decisions can thus be achieved. The primary effort was made in the direction of tightening timing estimations and the evaluation and improvement of cache analysis techniques with a focus on improving compiler optimizations [KFM11].

Several partners have worked on measurement-based/hybrid approaches to WCET analysis. Regarding test-case generation, work has been done on generation of test cases using pair coverage metrics as well as on reducing the test vector sets for streaming applications with potentially huge input data spaces. Furthermore a hybrid approach to early WCET source-level analysis has been developed, where a timing model for source code is automatically derived from measurements.

For parametric WCET analysis work in Y4 has focussed on the soundness of the relational value analyses, in the presence of arithmetic wraparounds, which are used in the parametric analysis. MDH and Tidorum have improved on the precision of a sound polyhedral analysis.

During Y4 the work on WCET analysis for explicitly parallel programs has concerned foundational issues. A model parallel language has been defined, together with formal semantics. The purpose is to verify the correctness of a WCET analysis for such programs.

Finally, work on tool infrastructure and tool interfacing has picked up. Work has begun to develop an open and extensible evaluation platform for timing analysis, which is centred around a mainstream open-source compiler framework and open computer architectures. A possible candidate under investigation is the LLVM compiler framework, and an open-source processor. A link to MDH's WCET analysis tool SWEET has been built, adding a backend for the "ALF format" to LLVM, which allows LLVM to be used as a C frontend to SWEET. In a

similar vein, the tool Bound-T has been extended to export ALF code that can be analyzed by SWEET.

-- The above is new material, not present in the Y3 deliverable --

3. Detailed view of the progress in Year 4 *(Jan-Dec 2011)*

3.1 Technical Achievements

Timing analysis and timing predictability (USaar)
We continued our efforts towards a formal definition of predictability. We proposed a template for predictability definitions, which captures important aspects of predictability, reviewed pertinent literature of the past years, and were able, in most of the cases, to cast the authors' intuitions on predictability in terms of our template. The most severe disagreement between our opinion on predictability and those of others concerns the question whether predictability should be an *inherent* property.

Our work on static cache analysis has been continued. The dissertation of D. Grund comprehensively treats the cache analyses developed in the last years. These analyses are now subject to technology transfer to AbsInt GmbH, who has indicated great interest in the results.

We further investigated predictable, cache-aware dynamic memory allocator designs in order to handle uncertainty of cache effects arising from dynamic memory allocation. Our new designs implement cache-aware techniques for area allocations, i.e., small blocks can be efficiently allocated in bulks, as well as improved placement policies for their internal management structures.

We proceeded to the integration of the cache-related preemption delay (CRPD) developed in Year 3 into the schedulability analysis of fixed priority preemptive systems. Beside a thorough review of existing methods, we developed a new and more precise CRPD aware response time analysis and also presented a corrected analysis in case of shared resource taking into account the effect of the preempted and the effect of the preempting task.

Highly variant embedded systems require a specialized timing analysis: a bound valid for all variants at once is too pessimistic while a separate analysis for each variant is computationally intractable. We thus developed a variant-aware timing analysis based on an ILP formulation of the variant dependencies. As a by-product, the new analysis also delivers the worst-case variant, i.e., the variant causing the highest execution-time bound.

Combination of model checking and abstract interpretation for timing analysis (NUS)
In this work, we study scalable and precise timing analysis methods which combine the power of model checking and abstract interpretation. Unlike previous works which employ model checking for path analysis and abstract interpretation for micro-architectural analysis – our framework offers a more fine grained integration. We show how scalable and precise cache timing analysis can be achieved by using abstract interpretation as the base analysis, and refining the cache conflicts detected by abstract interpretation using model checking. We have employed this analysis framework in different architectural settings, such as single core cache timing analysis, multi-core shared cache timing analysis and cache related pre-emption delay (CRPD) analysis. More details about our approach appear in the recent work [CR11].

WCET analyzer for multi-cores (NUS, TU Dortmund)
With the advent of multi-core architectures, worst-case execution time (WCET) analysis has become an increasingly difficult problem. In this work, we propose a unified WCET analysis framework for multi-core processors featuring both shared cache and shared bus. Compared to other previous works, our work differs by modeling the interaction of shared cache and shared bus with other basic micro-architectural components (e.g. pipeline and branch predictor). In addition, our framework does not assume a timing anomaly free multicore architecture for computing the WCET. A detailed experimental methodology suggests that we

can obtain reasonably tight WCET estimates in a wide range of benchmark programs. More details about our approach appear in the recent work [CKR+12].

Bit-precise relational domains for WCET flow analysis (MDH, Tidorum)
For small embedded processors, arithmetic overflows and wraparounds are an issue. They can appear either by mistake or intentionally when optimizing the code. A sound program flow analysis must take them into account. Unfortunately many program analyses do not, which means that their use to constrain possible program flows for WCET analysis can yield unsound results. Relational abstract numerical domains are of particular interest to program flow analysis. We have improved on an existing approach to make the polyhedral abstract domain sound in the presence of overflows. Our approach uses type information to restrict values whenever possible, together with a judicious placement of widening points, to improve the precision of the analysis [BLH11].
http://www.aparts.se/

WCET analysis for explicitly parallel programs (MDH)
WCET analysis for parallel programs is a challenging topic. Even disregarding the difficulties on the hardware level, stemming from timing side effects through shared resources, waiting times for synchronizations and possible race conditions necessitate an integrated analysis where timing and program flows are analyzed together. We have outlined such an analysis. In order to prove its correctness, we have defined a small parallel language. We have defined both a dynamic semantics, and a collecting semantics, for this language.
http://www.mrtc.mdh.se/index.php?choice=projects&id=0327

Early WCET analysis (MDH, TU Vienna)
WCET analysis is typically applied late in the development process, when verifying the timing properties. There is a need to estimate timings earlier in the development process, to support the time budgeting, and the dimensioning of the hardware. Such WCET estimates need not be safe, as long as a final verification with a safe method is done. One way to estimate WCET's early is to create approximate timing models for source code: with such models, WCET analysis can be done on source code, or parts of it, early in the process. We have developed a method to identify such models automatically from test runs of a suit of specially crafted training programs. The derived timing model is then used in a source-level WCET analysis. Initial experiments indicate a deviation of estimate from real WCET mostly in the range 0-30%.
http://www.mrtc.mdh.se/index.php?choice=projects&id=0017

Measurement-based WCET analysis (TU Vienna)
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 have explored methods for generating test cases that provide a good coverage by investigating methods for partitioning and analyzing control-flow graphs. In year 4, the use of pair coverage metrics as a criterion for generating input data for hybrid and measurement-based execution time analysis was explored. The experiments showed that the use of this additional information helps to improve the results of the analysis in many applications with input-data dependent control flow. A PhD thesis that will present the results in detail is expected to be finished in spring 2012.

Composable software/hardware architectures (TU Vienna)
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. In year 4 we explored the use of dynamic preview strategies for prefetching and compared them to a fully static approach that relies on pre-runtime analysis and planning. The evaluations showed the superiority of the static approach due to the limitation of the dynamic approach with respect to predicting conditional control-flow branches at runtime. At the time of this writing we are putting together a proposal for a national project to continue the work on time-predictable memory hierarchies.

Support of Tool Evaluation and RT Design Process (MDH, TU Vienna)
We have implemented a first prototype of an open and extensible platform for the evaluation of architectures, compilers, and timing analysis strategies, centred around a mainstream open-source compiler framework and open computer architectures. We investigated a possible candidate built around the LLVM compiler framework and an open-source LEON processor, and incorporated the SWEET tool for timing analysis to demonstrate that the effort for tool integration into the framework is in the order of a few days. Besides providing a platform for (a) investigating the effects of code generation methods and hardware architecture features on timing and timing analysis and (b) testing different approaches to WCET analysis, the platform shall in the future be used to explore methods for architecture independent exploration of the timing behaviour of software. The connection to SWEET makes it possible to use LLVM as a C frontend for SWEET, thus enabling SWEET to perform timing analysis on source level.

Time-Predictable Multi-Core Architectures (AbsInt, DTU, TU Vienna, York)
Based on the observation that the speedup features of current processor and computer architectures are an impediment to constructing time-predictable systems, members of the NoE are now involved in the T-Crest project consortium working on a multi-core computer architecture and compiler concepts that support the development of time-predictable applications, thus simplifying the WCET and overall timing analysis. T-CREST is funded by the European Commission under grant agreement 288008.

WCET-aware cache locking and code positioning (TU Dortmund, AbsInt)
Based on techniques developed in the past, a new fine grained technique for WCET-aware static locking of instruction caches has been developed at TU Dortmund. In order to improve the performance and the predictability of programs, memory blocks are locked into the cache. Time consuming repetitive WCET analyses are avoided by employing integer linear programming and modelling the worst-case execution path of the program to optimize. By taking the effect of locked memory on the runtime of basic blocks into account, the overall WCET can be efficiently minimized. It was also evaluated in how far existing WCET-driven memory-architecture-based optimizations can be combined in a sophisticated fashion for the optimization of multi-task sets. WCET-driven techniques such as scratchpad allocation, memory content selection and cache partitioning were examined for their interaction and possible synergetic effects. A heuristic for multi-task scratchpad allocation has been developed and a useful application order has been determined. A new WCET-driven cache-aware code positioning which optimizes the WCET of applications and reduces cache conflicts has been devised at TU Dortmund. The code positioning is applied both to basic blocks and to entire functions. The algorithm works by construction a cache conflict graph (CG), meaning a weighted graph describing if and how much certain code fragments are overlapping in the cache. The initial version of the CG is build using the control flow graph annotated with the number of cache misses and the execution frequencies. The CG is then refined using the results of a control flow analysis, a cache analysis and the memory layout of the program. With these refinement steps, the graph describes the potential cache conflicts. Using the refined graph, the actual code positioning is performed by moving those code fragments which offer the most optimization potential, i.e. which have the most conflicts. This avoids overlapping of cache sets and thus decreases the number of cache conflicts [FK11].

http://ls12-www.cs.tu-dortmund.de/research/activities/wcc

Reduction of test vector sets for measurement-based WCET analysis (York)
York presented at the 6th IEEE International Symposium on Industrial Embedded Systems (SIES'11) a paper on choosing appropriate input signals as part of WCET analysis. A key problem with real systems is they take a stream of data and a set of parameters as input. The search space for the data can be potentially huge so a key challenge is taking a representative subset of it for scalability reasons. The subset is also guaranteed to contain the longest

measurable frame in terms of execution time. In this paper a large set of data was analysed and then a representative subset chosen using non-parametric statistical techniques. The work was evaluated using the VP8 codec from ffmpeg and a long film was subset to a manageable clip using the approach developed.

Predictability as an emergent behaviour (York)
One means of obtaining time predictable systems is to build them from predictable components which are themselves built from predictable subcomponents. But this is not the only approach. Benefits are possible if components are designed to exhibit independent random behaviour. The worst-case execution time of application code can be estimated (with low probability of failure) to be not much greater than what one would expect the average to be. Unfortunately this approach of obtaining predictability as an emergent property of the code's execution cannot be delivered by today's hardware. We have made an initial exploration of the means by which hardware can be made more accommodating to this objective.

-- The above is new material, not present in the Y3 deliverable --

3.2 Individual Publications Resulting from these Achievements

USAAR

E. Althaus, S. Altmeyer and R. Naujoks. *Symbolic Worst Case Execution Times.* Proceedings of the 8th International Colloquium Theoretical Aspects of Computing (ICTAC), 2011.

P. Montag, and S. Altmeyer. *Precise WCET Calculation in Highly Variant Real-Time Systems.* Proceedings of Design, Automation, and Test in Europe (DATE), 2011.

D. Grund, J. Reineke, and R. Wilhelm. *A Template for Predictability Definitions with Supporting Evidence.* Proceedings of Bringing Theory to Practice: Predictability and Performance in Embedded Systems (PPES), 2011.

D. Grund. *Static Cache Analysis for Real-Time Systems.* PhD Thesis, Saarland University, 2011.

J. Herter, P. Backes, F. Haupenthal, and J. Reineke. *CAMA: A Predictable Cache-Aware Memory Allocator*. Proceedings of the 23rd Euromicro Conference on Real-Time Systems (ECRTS), 2011.

T. Dudziak, and J. Herter. *Cache Analysis in Presence of Pointer-Based Data Structures*. Proceedings Work-In-Progress Session of the 23rd Euromicro Conference on Real-Time Systems, 2011.

NUS

[CR11] Sudipta Chattopadhyay and Abhik Roychoudhury*. Scalable and Precise Refinement of Cache Timing Analysis via Model Checking.* Proc. IEEE Real-time Systems Symposium (RTSS) 2011.

Sudipta Chattopadhyay and Abhik Roychoudhury. *Static Bus Schedule aware Scratchpad Allocation in Multiprocessors.* Proc. ACM SIGPLAN/SIGBED Conference on Languages, Compilers and Tools for Embedded Systems (LCTES) 2011.

MDH

Peter Altenbernd, Andreas Ermedahl, Björn Lisper, and Jan Gustafsson. *Automatic Generation of Timing Models for Timing Analysis of High-Level Code*, Sébastien Faucou (ed) Proc. 19th International Conference on Real-Time and Network Systems (RTNS2011), Nantes, France, Sep. 2011.

Andreas Gustavsson. *Worst-Case Execution Time Analysis of Parallel Systems*, Dag Nyström and Thomas Nolte (eds) Proc. Real Time in Sweden 2011 (RTiS2011), pp. 104-107, Västerås, Sweden, June 2011.

AbsInt

D. Kästner, S. Wegener, C. Ferdinand. Safety Standards and WCET Analysis Tools. Proceedings of the 11th International Workshop on Worst-Case Execution-Time Analysis (WCET 2011), Porto, 2011.

D. Kästner, C. Ferdinand, R. Heckmann, M. Jersak, P. Gliwa. An Integrated Timing Analysis Methodology for Real-Time Systems. SAE World Congress 2011, doi:10.4271/2011-01-0444, available at http://papers.sae.org/2011-01-0444/.

D. Kästner, C. Ferdinand, R. Heckmann, M. Jersak, P. Gliwa. An Integrated Timing Analysis Methodology for Real-Time Systems. Embedded World Congress 2011, Nürnberg, 2011.

D. Kästner, C. Ferdinand, R. Heckmann, M. Jersak, P. Gliwa. An Integrated Timing Analysis Methodology for Real-Time Systems. Design und Elektronik, Embedded Software Engineering, 2011. (As presented at Embedded World 2011, on invitation.)

D. Kästner, C. Ferdinand. Using Code Analysis Tools for Software Certification. Embedded World Congress 2011, Nürnberg, 2011.

D. Kästner. Echtzeitfähigkeit von Multicores. E&E Magazine, volume 3/2011. Publish-industry Verlag, 2011.

D. Kästner. Nicht-funktionale Softwareanforderungen im Kontext aktueller Sicherheitsstandards: Timing, Speicherverbrauch, Laufzeitfehler (Extended Abstract). Euroforum Jahrestagung ISO-26262.

D. Kästner, C. Ferdinand. Efficient Verification of Non-Functional Safety Properties by Abstract Interpretation: Timing, Stack Consumption, and Absence of Runtime Errors. Proceedings of the 29th International System Safety Conference ISSC2011, Las Vegas, 2011.

D. Kästner. Verifikation nicht-funktionaler Sicherheitseigenschaften durch Abstrakte Interpretation (Extended Abstract). Safetronic 2011.

G. Gebhard, C. Cullmann, R. Heckmann. Software Structure and WCET Predictability. In P. Lucas, L. Thiele, B. Triquet, T. Ungerer, R. Wilhelm (eds.), Bringing Theory to Practice: Predictability and Performance in Embedded Systems, DATE Workshop PPES 2011. OASICS, volume 18, Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik, Germany, 2011.

C. Cullmann. Cache Persistence Analysis: A Novel Approach - Theory and Practice. In J. Vitek, B. de Sutter (eds.), Proceedings of the ACM SIGPLAN/SIGBED 2011 conference on Languages, compilers, and tools for embedded systems, LCTES 2011, ACM, 2011.

TU Vienna

Benedikt Huber, Wolfgang Puffitsch, and Peter Puschner. *Towards an Open Timing Analysis Platform*. In Proc. 11th Euromicro Workshop on WCET Analysis, p. 5-14, 2011.

Michael Zolda, Sven Bünte, and Raimund Kirner. *Context-Sensitive Measurement-Based Worst-Case Execution Time Estimation*. In Proc. 17th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications, p. 243-250, 2011.

Christian El Salloum and Peter Puschner. *Mapping Safety-Critical Applications to Multi-Processor Systems-on-a-Chip* (abstract). In Proc. 4th Workshop on Mapping of Applications to MPSoCs, 2011.

Bekim Cilku and Peter Puschner. *Using a Local Prefetch Strategy to Obtain Temporal Time Predictability*. In Proc. 14th IEEE International Symposium on Object/component/service-oriented Real-time Distributed Computing Workshops, 2011.

TU Dortmund

[PKFM11] Sascha Plazar, Jan C. Kleinsorge, Heiko Falk and Peter Marwedel. WCET-driven Branch Prediction aware Code Positioning. In *Proceedings of the International Conference on Compilers, Architectures and Synthesis for Embedded Systems (CASES)*, pp. 165-174, 2011.

[KFM11] Jan C. Kleinsorge, Heiko Falk and Peter Marwedel. A Synergetic Approach To Accurate Analysis Of Cache-Related Preemption Delay. In *Proceedings of the International Conference on Embedded Software (EMSOFT),* pp. 329-338, 2011.

[FK11] Heiko Falk and Helena Kotthaus. WCET-driven Cache-aware Code Positioning. In *Proceedings of the International Conference on Compilers, Architectures and Synthesis for Embedded Systems (CASES),* pages 145-154, Taipei, Taiwan, October 2011.

Heiko Falk, Norman Schmitz and Florian Schmoll. WCET-aware Register Allocation based on Integer-Linear Programming. In *Proceedings of the 23rd Euromicro Conference on Real-Time Systems (ECRTS)*, pages 13-22, Porto, Portugal, July 2011.

York
Sitsofe Wheeler, Iain Bate, and Mark Bartlett. Video Subset Selection for Measurement Based Worst Case Execution Time Analysis. *Proc. 6th IEEE International Symposium on Industrial Embedded Systems (SIES'11)*, 2011.

A. Burns and D. Griffin. Predictability as an Emergent Behaviour. Robert I. Davis and Linh T.X. Phan (eds.) *Proc. 4th Workshop on Compositional Theory and Technology for Real□Time Embedded Systems*, pages 27-29, Vienna, Austria, Nov. 2011.

- The above are new references, not present in the Y3 deliverable --

3.3 Interaction and Building Excellence between Partners

Interaction between TU Vienna, and MDH
TU Vienna and MDH investigate the connections and cooperation between the flow-fact analysis and its realization in the SWEET tool (focus at MDH) and the work on writing time-predictable software (focus at TU Vienna). After a first integration of SWEET into the open timing analysis framework the partners are now seeking to use the timing analysis platform for less hardware dependent early-stage timing estimations. See Section 3.1.

Interaction between MDH, and Tidorum

MDH and Tidorum have cooperated on the topic of bit-precise relational domains, which is described more closely in Section 3.1. Furthermore, Tidorum has extended its WCET tool, Bound-T, to allow the program under analysis to be exported in ALF format, for further analysis by MDH's SWEET tool.

Interaction between AbsInt, DTU, TU Vienna, and York
Within T-CREST partners have started to work on the integration of timing analysis and compilation for time-predictable multi-core platforms, where the hardware architecture is being designed with the goal to support the generation of time-predictable code. See Section 3.1.

Interaction between USAAR, and AbsInt

Results on the relative competitive analysis of cache replacement policies achieved by Saarland University have been used in AbsInt's WCET analyzer aiT to increase the performance and precision of the analysis of caches with PLRU replacement policy.

A prototype implementation of the UCB computation as developed by Saarland University has been integrated by AbsInt into the aiT Timing Analyzer. The analysis was originally implemented for the ARM7 and has been tested on smaller benchmark programs. In Year 4, the implementation has been optimized greatly (much higher performance and reduced memory consumption), and the analysis has been extended to other target processors including MPC55xx, MPC603e, and Leon2.

After the successful end of the PREDATOR project, AbsInt became partner of the new projects T-CREST (FP7 ICT-288008) and CERTAINTY (FP7 ICT-288175). AbsInt is importing the results of PREDATOR on the predictability of hardware configurations and its experience from the participation in the set-up of a WCET-aware compiler into these new projects.

Interaction between TU Dortmund, and NUS

TU Dortmund cooperated with the National University of Singapore (NUS). In this cooperation, WCC's capabilities were advanced in the direction of multicore WCET analysis and optimization. As presented in Section 3.1, multicore architectures pose new problems for timing analysis through usage of shared resources. TU Dortmund developed a new analysis which is able to determine bounds on the delay incurred by accesses to TDMA-arbitrated shared resources. Experiments with an implementation based on the Chronos WCET analyzer show that the new analysis is able to outperform previously published approaches [KFMCA11]. This enhanced precision can now be used to effectively guide WCET-directed multicore optimizations in WCC.

WCET Challenge

The third WCET Challenge, initiated in 2010 and described in the activity report for 2010, was successfully completed in 2011. Eight tools participated in the Challenge, including commercial and academic tools from both ArtistDesign partners and tool providers outside the NoE. The results were reported at the WCET Workshop 2011.

http://www.mrtc.mdh.se/projects/WCC/2011/

-- Changes wrt Y3 deliverable --

Several new interactions have emerged in Y4.

3.4 Joint Publications Resulting from these Achievements

S. Altmeyer, and C. Maiza. *Influence of the Task Model on the Precision of Scheduling Analysis for Preemptive Systems – Status Report*. Proceedings of the 2nd International Real-Time Scheduling Open Problems Seminar (RTSOPS), 2011

S. Altmeyer, R. I. Davis, and C. Maiza. *Cache Related Pre-emption Aware Response Time Analysis for Fixed Priority Pre-emptive Systems* Proceedings of the 32nd Real-Time Systems Symposium (RTSS), 2011.

[CKR+12] Sudipta Chattopdhyay, Chong Lee Kee, Abhik Roychoudhury, Timon Kelter, Peter Marwedel, Heiko Falk. *A Unified WCET Analysis Framework for Multi-core Platforms.* Submitted to: 18th IEEE Real-time and Embedded Technology and Applications Symposium (RTAS) 2012.

[KFMCA11] Timon Kelter, Heiko Falk, Peter Marwedel, Sudipta Chattopadhyay and Abhik Roychoudhury. *Bus-Aware Multicore WCET Analysis through TDMA Offset Bounds.* Proc. 23rd Euromicro Conference on Real-time Systems (ECRTS) 2011.

Stefan Bygde, Björn Lisper, and Niklas Holsti. *Static Analysis of Bounded Polyhedra*, Paul Pettersson and Cristina Seceleanu (eds) Proc. Nordic Workshop of Programming Theory (NWPT), pp. 83-85, Västerås, Sweden, Oct. 2011

[BLH11] Stefan Bygde, Björn Lisper, and Niklas Holsti. *Fully Bounded Polyhedral Analysis of Integers with Wrapping*, Proc. International Workshop on Numerical and Symbolic Abstract Domains , Venice, Italy, Sep. 2011

Reinhard von Hanxleden, Niklas Holsti, Björn Lisper, Erhard Ploedereder, Armelle Bonenfant, Hugues Cassé, Sven Bünte, Wolfgang Fellger, Sebastian Gepperth, Jan Gustafsson, Benedikt Huber, Nazrul Mohammad Islam, Daniel Kästner, Raimund Kirner, Laura Kovacs, Felix Krause, Marianne de Michiel, Mads Christian Olesen, Adrian Prantl, Wolfgang Puffitsch, Christine Rochange, Martin Schoeberl, Simon Wegener, Michael Zolda, Jakob Zwirchmayr. *WCET Tool Challenge 2011: Report,* Chris Healy (ed) Proc. 11th International Workshop on Worst-Case Execution Time (WCET) Analysis (WCET 2011), OCG, Porto, Portugal, July 2011

D. Grund, J. Reineke, G. Gebhard. Branch target buffers: WCET analysis framework and timing predictability. Journal of Systems Architecture - Embedded Systems Design, volume 57, number 6, 2011.

-- The above are new references, not present in the Y3 deliverable --

3.5 Keynotes, Workshops, Tutorials

Keynote (Björn Lisper): Parametric WCET Analysis
Nordic Workshop of Programming Theory
Västerås, Sweden – Oct 28, 2011

The purpose of Worst-Case Execution Time (WCET) analysis is to compute a safe upper bound to the execution time of a sequential program executing uninterrupted on some given hardware. Such bounds are important when verifying the timing requirements on hard real-time systems. WCET analysis has been an active research topic for the last 20 years, and today there exists a large body of theory, methods, and algorithms. Both academic and commercial tools have emerged during the last decade, and the technique is becoming established in industrial use.

Traditional WCET analysis computes a single number. For programs whose execution time varies strongly with the inputs, a single upper bound may provide very large overestimations in most situations since it has to take the program executions for all possible input values into account. It may then be advantageous to have a parametric WCET analysis, which computes the WCET bound as a symbolic formula in the unknown inputs rather than as a single number. When the formula is instantiated for the specific inputs at hand, the resulting number is likely to provide a much tighter bound for the actual WCET. Thus, it is highly interesting to develop good methods and tools for parametric WCET analysis.

In this talk we first give a short primer to WCET analysis. We then give an account for the past, present, and planned future research at Mälardalen University regarding parametric WCET

analysis.
http://www.mrtc.mdh.se/nwpt2011/

Workshop : Bringing Theory to Practice: Predictability and Performance in Embedded Systems
Conference on Design, Automation & Test in Europe (DATE)
Grenoble, France – March 18, 2011

The PPES workshop is concerned with critical hard real-time systems that have to satisfy both efficiency and predictability requirements. The aim of this workshop was twofold: to present the results achieved and tools developed by various researchers, in particular to industrial end users; and to present the industrial viewpoint on needs and challenges which need to be tackled for applicability.

To this end, the workshop comprised an invited presentation by Ottmar Bender (Cassidian Electronics) on Predictability and Performance Requirements in Avionics Systems, a panel discussion on Predictability and Performance in Industrial Practice, and 7 regular paper presentations. PPES was supported by ArtistDesign, the European Network of Excellence on Embedded Systems Design; the PREDATOR project (Design for Predictability and Efficiency); and the MERASA project (Multi-Core Execution of Hard Real-Time Applications Supporting Analysability).

The workshop was organised by Philipp Lucas (Universität des Saarlandes), Lothar Thiele (ETH Zürich), Benoît Triquet (Airbus), Theo Ungerer (Augsburg University), and Reinhard Wilhelm (Universität des Saarlandes; chair).

http://ppes2011.cs.uni-saarland.de/

Workshop : 11th Int'l Workshop on Worst-Case Execution Time Analysis (WCET'11)
Porto, Portugal – July 6, 2011

Objectives for the meeting: To present and discuss recent work in WCET analysis issues, dealing with various kinds of systems, as well as static and dynamic approaches

*Organize*r: Chris Healy (Furman University, USA)

Other participants: About 30

Conclusions: Presenters came from both academia and industry. There was considerable discussion on control-flow techniques to yield tighter timing bounds. In addition, some presentations dealt with high-level issues such as how to effectively create research platforms for robust WCET testing, as well as industrial applications of WCET analysis techniques. The workshop also featured a keynote talk by Francisco Cazorla of the Barcelona Supercomputing Center on the subject of hardware support features.

http://www.artist-embedded.org/artist/-WCET-2011-.html

-- The above is new material, not present in the Y3 deliverable --

4. Internal Reviewers for this Deliverable

- **Heiko Falk** (Univ. Ulm)
- **Thomas Nolte** (MDH)