



Safety and Efficiency.

Always on the safe side with AbsInt.

aiT / TimingExplorer / TimeWeaver

timing verification & validation

is your program
always ...

... fast enough?

Efficient Verification of Timing Behavior

aiT Worst-Case Execution Time Analyzer computes safe upper bounds on the worst-case execution time of tasks in real-time systems. The computed bounds are valid for all inputs and each task execution. Cache and pipeline effects are taken into account. aiT has been used successfully for certifying safety-critical software, e.g., according to DO-178B/Level A.

TimingExplorer is based on aiT's static timing analysis technology. It offers a set of parameterizable ECU core models to explore the effects of different ECUs or different ECU configurations on the execution speed. Used in early design phases it helps to avoid late-stage integration problems.

TimeWeaver combines observed code snippet execution times from real-time instruction-level tracing with aiT's method to compute longest execution paths in order to compute task-level WCET estimates. This provides feedback for optimizing the worst-case performance or for investigating the timing behavior of code.



A l w a y s o n t h e s a f e

StackAnalyzer / ValueAnalyzer

memory usage validation

now a thing of
the past:



stack overflow memory corruption

StackAnalyzer automatically determines the worst-case stack usage of tasks and provides feedback for optimizing the stack usage. All potential stack overflow errors can be detected without executing the program. **StackAnalyzer** has been used successfully for certifying safety-critical software, e.g. according to DO-178B/Level A.

ValueAnalyzer automatically determines which memory areas are read or written by the tasks in your application by a static program analysis on binary code. This is useful for verifying the absence of illegal memory accesses.

Astrée

verifying the absence of runtime errors

did you fix all ...



... runtime errors?

Astrée finds all potential runtime errors in C programs. It analyzes safety-critical structured C programs and is also applicable to automatically generated C code.

Astrée is sound. If the analysis does not detect any errors, the absence of runtime errors has been proven.

Astrée's powerful analysis engine, flexible parametrization and comfortable GUI efficiently lead to precise results.

Astrée detects all out-of-bound array accesses, integer division by zero, invalid pointer dereferences, integer arithmetic overflows, floating-point overflows, invalid operations, etc.

Astrée proves user-defined assertions and detects unreachable code.

Astrée has been used successfully to analyze large-scale safety-critical software with zero false alarms.

Abstract Interpretation

verification tool technology

Contemporary safety standards (DO-178B, DO-178C, IEC-61508, ISO-26262, EN-50128, etc.) require identifying potential functional and non-functional hazards and demonstrating that the software does not violate the relevant safety goals.

Especially for non-functional program properties – timing, memory usage, absence of runtime errors – tests and measurements can be ineffective, inefficient and expensive: identifying safe end-of-test criteria is typically undecidable since failures usually occur in corner cases and full test coverage cannot be achieved.

Abstract interpretation based program analysis tools like **aiT**, **StackAnalyzer**, **ValueAnalyzer** and **Astrée** provide a solution to this problem. Static program analysis is a widely used technique to automatically determine runtime properties of a given program without actually executing it. Abstract interpretation is a semantics based framework for static program analysis that enables the systematic derivation of provably correct analyses. Abstract interpretation amounts to performing a program's computations using value descriptions or abstract values in place of concrete values. One reason for using abstract values instead of concrete ones is computability: to ensure that analysis results are obtained in finite time. Another reason is to obtain results that hold for every program execution and all possible inputs.

aiT, **StackAnalyzer**, **ValueAnalyzer** and **Astrée** can not only be applied in the late development stage of validation and verification. They also can be smoothly integrated into the development process to detect bugs and errors early when the costs for a fix are lowest and the benefit highest.

Certification and Qualification

Abstract interpretation based tools like **aiT**, **StackAnalyzer**, **ValueAnalyzer** and **Astrée** are formal verification tools providing 100% complete and reliable results and are therefore perfectly suited to be used for certification.

The tool qualification process is widely simplified by qualification support kits (QSKs). They specify the tool requirements and the verification test plan and contain an extensible test package. They also provide scripts to execute all test cases and to evaluate and document the results. **AbsInt's** tool development and software quality process is detailed in a Qualification Software Life Cycle Data report.

Founded in 1998, **AbsInt** is a privately-held company located in Saarbrücken, Germany.

AbsInt provides advanced development tools and tools for validation, verification and certification of safety-critical software.

AbsInt's tools are designed to:

- enhance software safety,
- speed up time-to-market,
- lower testing and validation costs,
- improve software efficiency.

Consultancy and Services

AbsInt offers consultancy and service in the areas of program analysis, compiler technology, and program validation and verification.

AbsInt's specialists perform code analyses of your software projects as a service according to your requirements.

Customers

Our customers belong to the most respected and innovative companies from avionics, automotive, railway and healthcare technology sectors.

AbsInt's tools are used to validate safety-critical software and to optimize embedded applications. Software products optimized and validated by **AbsInt's** tools are in daily use by millions of people.

AbsInt Angewandte Informatik GmbH

Tel.: +49 681-383-600
Fax: +49 681-383-6020
info@absint.com
www.absint.com

www.absint.com