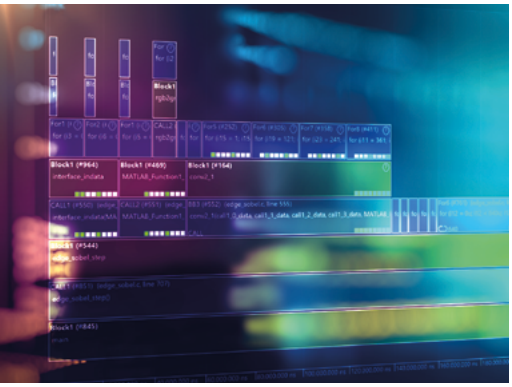
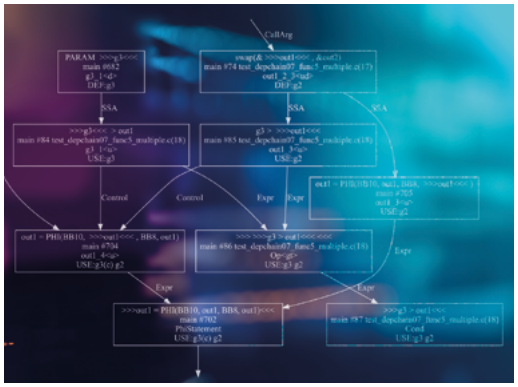
emmtrix
Technologies





Automation Tools for Better Code

Content

Welcome to emmtrix Technologies

We are an innovative company in the field of software development for embedded systems from Karlsruhe, Germany, founded in 2016.

As experts in compiler technology and static source code analysis, we help our customers analyze and optimize their code by offering software-tools and services.

Our aim is to provide comprehensive insight into the runtime performance of an application (emmtrix Performance Estimator) as well as the complexity/ interdependency of the code (emmtrix Dependency Analyzer) early on in the development process. Armed with this information, engineers can make informed decisions right from the start of the project. Our analysis solutions can be integrated into existing workflows and run in continuous integration set-ups on a defined schedule complete with versioning and changelogs.

Going beyond analysis we also help with automated software parallelization (emmtrix Parallel Studio and emmtrix Code Vectorizer) to optimize performance on multicore and manycore processors, as well as accelerators such as vector processors and DSPs. This parallelization can be qualified according to ISO 26262/DO-178C upon request.

Your emmtrix team

What Our Clients Say

> We have been working with emmtrix for a couple of years now and we find their technology of great interest. Their expertise in the field of source-to-source compiler technology and their tool suite emmtrix Parallel Studio help us in developing and improving our high-performance hardware solution. Sadahiro Kimura, Manager of Advanced Technology, NSITEXE

> ePS shows where and how performance can be increased. As a „side-effect“ of using ePS the developer quickly learns to design his application to be suitable for multicore HW.
>
> Arndt-Michael Meyer,
> Solution & Partner Manager, ETAS GmbH

emmtrix Tool Overview

Some Supported Platforms

- AURIX™ TC2x-TC4x
- arm
- RISC-V
- PowerPC
- x86
- R-Car
- DSP
- STM32

Performance Estimator

The Tool to GET the Performance of Your Applications

emmtrix Performance Estimator (ePE) provides static timing analysis of C code. Compared to simulation or measurement on hardware, static performance analysis can be applied significantly earlier in the development process and will deliver results on average 6 months earlier compared to a typical automotive HIL setup. The analysis only takes a few minutes at most and runs on the developer's PC independently of any target hardware. Function developers can analyze their runnables or SWCs without the need of a fully integrated program.
A unique feature is the combination with TargetLink or Embedded Coder generated code. Without any measurement overhead, our static performance estimation can analyze even the smallest code snippets. This allows us to map the timing analysis to Simulink blocks, giving function developers insight into the timing behavior of their models.

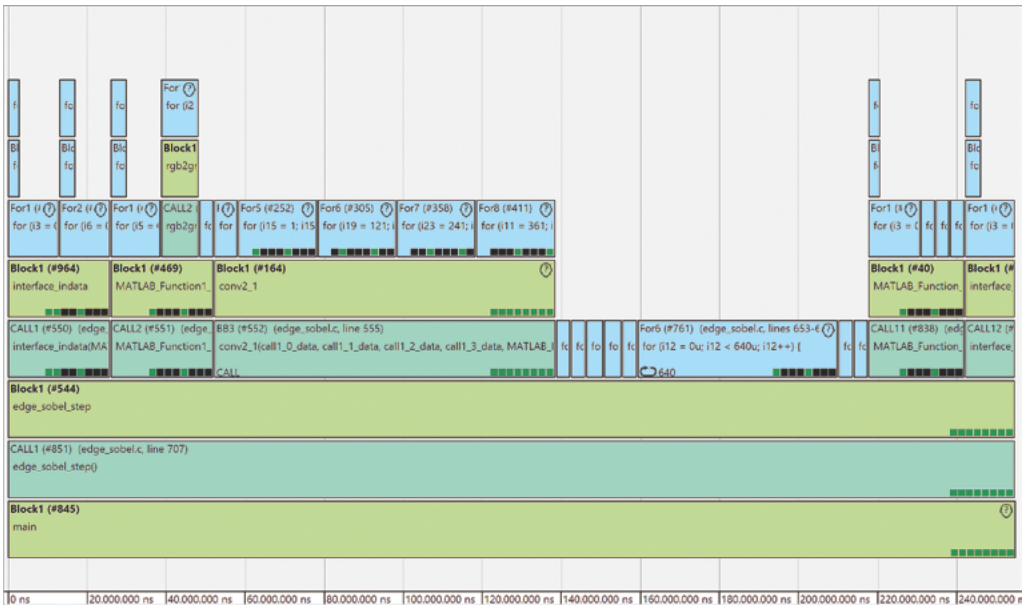
Features

- Automatic generation of reports and visualization for more detailed information
- Confidence levels for classification of results
- Easy to integrate into the development workflow
- Fast evaluation for different target platforms
- Static performance estimation based on C or assembly code
- Integration of simulators or hardware profiling into your workflow

Benefits

- Detailed information to better understand the timing behaviour of your application early in the development process
- Continuous monitoring of performance changes during the development
- Comparison of performance for different or heterogeneous target platforms
- Detect high-runners or critical parts of your software application
- Estimate the core utilization to optimize runnable/ task to core mapping

Visualization of the Performance Estimation

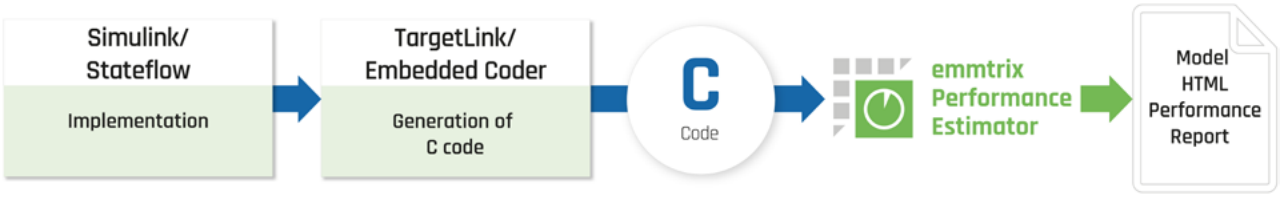


TargetLink Support

Get to Know the Performance of Your Simulink Models

ePE analyzes TargetLink generated C code and maps the performance values to the original Simulink blocks. This enables function developers who work with Simulink models to get a better understanding of the performance impact of design decisions on the embedded device (for more details see next page). Results can be accessed directly from generated HTML reports or viewed in the ePE GUI, helping to faster identify the bottlenecks/hotspots of the application.

ePE Workflow with TargetLink



Features

- Analysis of TargetLink generated C code
- Min and max execution time
- Number of time blocks of subsystems are called
- Interactive HTML report for results
  - Filtering
  - Searching
  - Sorting

Benefits

- Continuous monitoring of runtime with CI flow integration
- Runtimes for subsystems and modules without instrumentation of source code
- Exploration of different architectures without the need for hardware

HTML Report of Results

Platform: Aurix TC397 Starter Kit Processor: Infineon Aurix TC397 Clock Frequency: 300.000.000 Created: 11.5.2023, 14:44:52

Display runtime in Clock Cycles

Path	Simulink Block Type	Average Runtime (total runtime of a...)	Average Runtime (per execution)	Minimum Runtime	Maximum Runtime	Unknowns [Functions,...	Confidence
wristCtrl	Subsystem	588,13 cycles	588,13 cycles	405 cycles	703,87 cycles	[1,0,28,0]	69%
bahnplaner	Subsystem	190,5 cycles 32.4%	190,5 cycles	114 cycles	259,25 cycles 36.8%	[1,0,11,0]	60%
Subsystem1	Subsystem	83,75 cycles 44.0%	83,75 cycles	38 cycles	133,75 cycles 51.6%	[1,0,6,0]	45%
Subsystem	Subsystem	59,75 cycles 31.4%	59,75 cycles	31 cycles	92,75 cycles 35.8%	[0,0,5,0]	52%
OR	Logical	14,5 cycles 7.6%	14,5 cycles	14,5 cycles	14,5 cycles 5.6%	[0,0,0,0]	100%
OR1	Logical	14,5 cycles 7.6%	14,5 cycles	14,5 cycles	14,5 cycles 5.6%	[0,0,0,0]	100%
DetectChange_TL2	Subsystem	10 cycles	10 cycles	10 cycles	10 cycles	[0,0,0,0]	100%

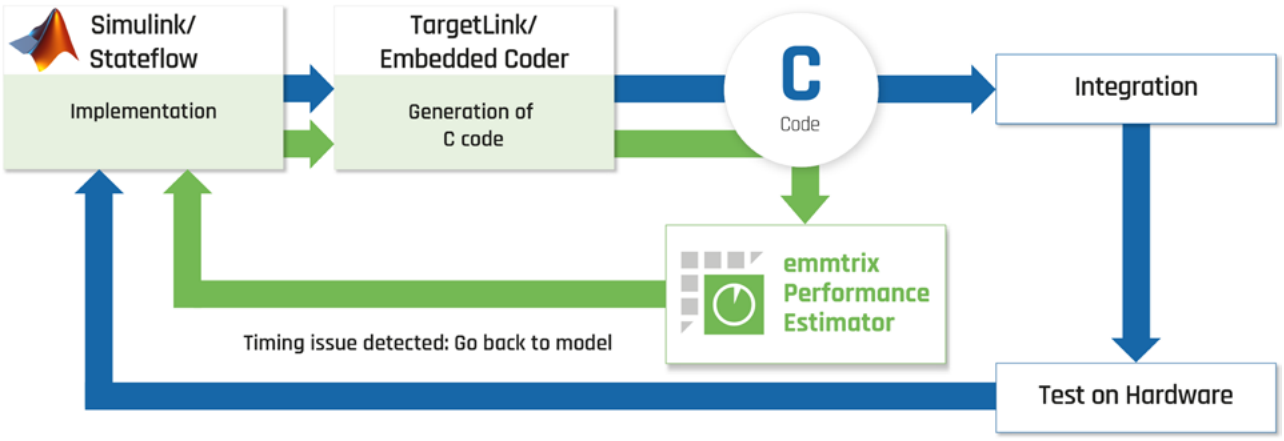
ePE for Automotive Function Developers Using Simulink and TargetLink

Use Case ePE

ePE calculates the runtime of Simulink blocks on a specified hardware platform based on static analysis of the source code generated from these blocks.

Here is a typical workflow in automotive:
- Functions are developed in Simulink
- TargetLink or Embedded Coder is used to generate C code for the target platform
- Function is integrated into the whole system and test setup
- Timing is measured on the hardware or a simulator
- In case of timing issues, the models have to be adapted and the steps are repeated

Today's Typical Model-Based Workflow in Automotive



Addressing timing issues in this way is expensive and time-consuming, as the issues are detected only after integration and measurements. Manually mapping these results back to Simulink is complicated as the granularity of the information is on the level of runnables or software components as specified in AUTOSAR and not on the actual model.

ePE can be integrated after code generation from the model as shown on the figure when following the green arrows. The user immediately receives feedback on the performance impact of implementation decisions without the need for integration or execution on hardware. Results of the automated analysis are presented in a hierarchical HTML report using the block names from the model. For all blocks, runtimes are provided for average, minimum, and maximum execution frequencies. This gives the function developer a much better idea of which parts of the application are taking the most time on the hardware. Optimizations are much easier and cheaper to implement at this stage than after integration on the hardware.

Success Stories

emmtrix Performance Estimator for Automotive Development

Introduction

Customers in the automotive industry face critical challenges when optimizing software performance under strict timing constraints. In AD/ADAS systems, ensuring that algorithms meet these demands is essential for commercial success. Traditional approaches often show timing issues very late during integration, making changes cumbersome and expensive.

Challenge

Developers frequently struggle to pinpoint inefficiencies, such as unnecessary computations and time-wasting operations that impact software runtime. Without actionable insights, optimization efforts are delayed, leading to higher costs and extended development cycles.

Solution

ePE analyzes C code generated from Simulink models via TargetLink or Embedded Coder. By providing accurate predictions of minimum, average, and maximum runtimes, ePE enables developers to identify and address performance bottlenecks. Its integration into CI/CD pipelines ensures continuous monitoring and immediate feedback on software changes.

> The HTML report generated by ePE helped us to identify inefficiencies in our models. In the report, the relative runtime between two modules was completely different to what we expected and this helped us remove some unnecessary executions in loops. This saved us a lot of time as we didn't have to measure everything on the hardware, but could improve our models directly.
>
> German OEM

> We were developing a new module with completely new functionality. The timing required the module to be completed in under 1ms. With ePE we could see immediately that our first approach was taking far too long. ePE also helped us to quickly pinpoint the places where too much time was being spent, so that we could optimise our implementations. As a result, we were able to reduce the expected runtime by a factor of 10 and get to hardware testing much faster.
>
> German First Tier

Dependency Analyzer

Identify the Event Chains of Your Applications

emmtrix Dependency Analyzer (eDA) assists you with the safety analysis of your applications: Results from a data dependency analysis of the source code can be used to:

- Verify freedom from interference
- Propagate different safety levels of variables
- Detect mixed-criticality dependencies (variables with different safety levels)
- Verify event chains between input and output signals of the system

A further use case is the optimization of your testing strategies by identifying which subsystems are affected by a change in the source code and, more importantly, which subsystems are not and therefore do not have to be tested again.

The tool can easily be integrated into CI flows offering automation, versioning and logging.

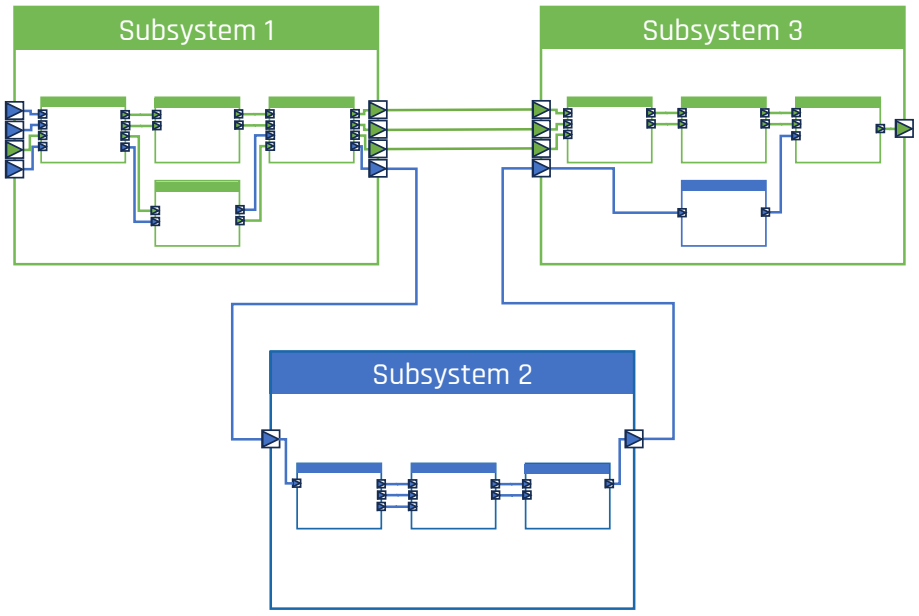
Features

- Analysis takes all possible paths of the control flow graph into consideration to ignore dependencies that can never occur
- Data and control dependencies between variables are calculated
- All calls to sub-functions are taken into account
- Supports analysis of programs consisting of multiple compilation units (source files)
- Supports analysis of delayed dependencies where values are stored in a variable and only fed to the output when the function is called again

Benefits

- Verify your expected dependencies
- Ensure that there are no unwanted connections between input and output signals
- Track down and document all modules affected by an input signal
- Identify which code will be affected by code changes to minimize re-testing effort
- Document all dependencies for any required certification process

Signal-Tracing across Subsystems



Use Case eDA

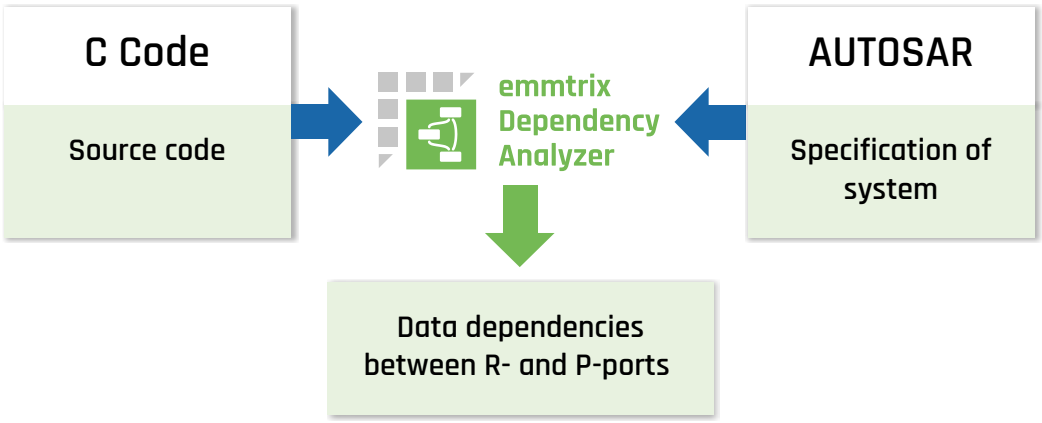
Enhancing Emissions Compliance

Background: In response to the 2015 emissions scandal, regulatory bodies like the EPA, CARB, and the EU have tightened vehicle emissions and On-Board Diagnostics (OBD) standards. Automobile manufacturers must now prove their OBD software is tamper-free and meets emission norms, supported by detailed documentation of algorithms used in emissions monitoring. This documentation fosters accountability and helps rebuild trust in automotive compliance.

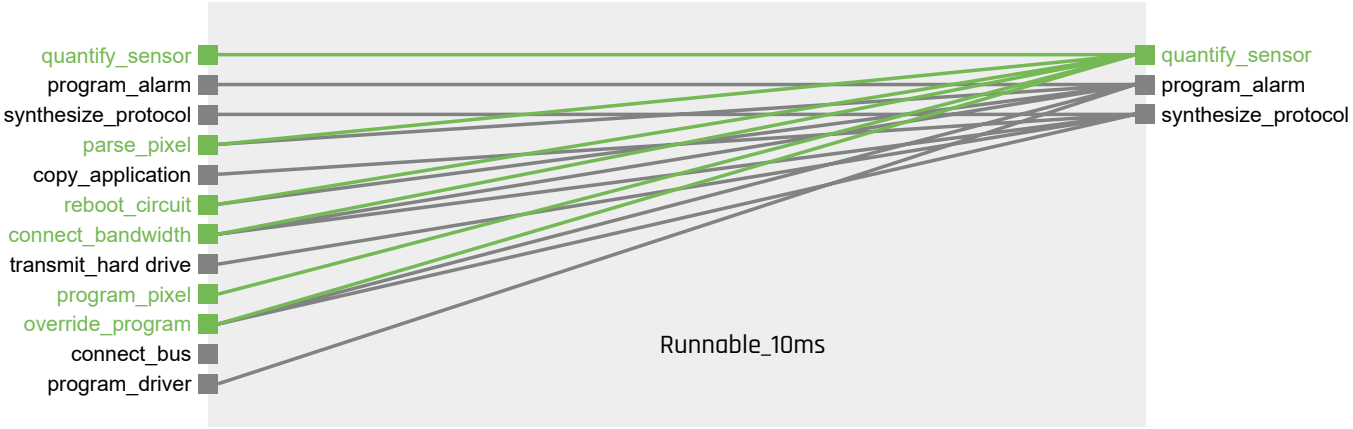
Challenge: Auto manufacturers face the challenge of ensuring their OBD software is fully compliant with stringent emission regulations. This entails not only adhering to legal requirements but also maintaining transparency and traceability in how the software manages and influences emissions. With regulatory scrutiny intensifying, manufacturers need a reliable method to analyze and document the behavior of their OBD systems under various conditions.

Solution: emmtrix Dependency Analyzer offers a groundbreaking solution for auto manufacturers seeking to navigate the complexities of emissions compliance. This static code analysis tool delves into the source code of OBD software, mapping out how specific input variables affect emission outputs. By identifying the intricate relationships between different components and variables, emmtrix Dependency Analyzer provides an unparalleled level of insight into the software's operational logic.

Typical AUTOSAR Workflow with Integrated eDA



Internal Data Dependencies of a Runnable



Using AUTOSAR with emmtrix Tools

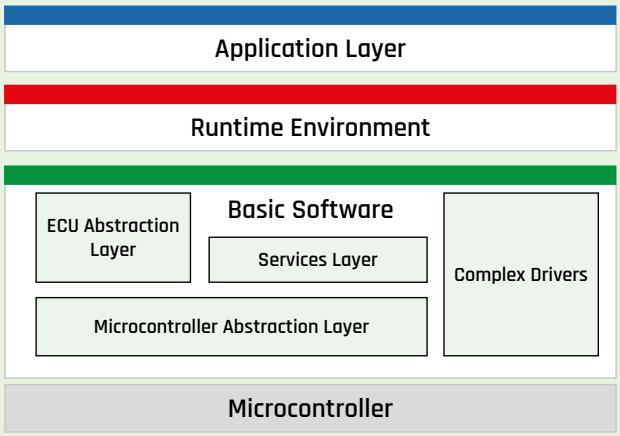
Enhanced Support for Automotive

The AUTOSAR classic platform is a standard specification that defines several abstraction layers from the microcontroller to the applications. The Basic Software (BSW) Layer consists of services such as operating system, communication, diagnostic, and hardware abstraction layers that provide standardized interfaces to the application layer. The Runtime Environment (RTE) acts as the middleware that enables communication between the application software and the basic software layer, abstracting the complexity of the hardware and the BSW modules. Finally, the Application Layer contains functional software components developed by OEMs or suppliers, which can be easily ported across different hardware platforms due to the abstraction provided by the RTE.

Features in an AUTOSAR Environment

- Automatic extraction of relevant runnables based on trigger events
- Performance estimation of individual runnables on different embedded architectures using, for example, Infineon AURIX, ARM Cortex or RISC-V processors
- Extraction of data dependencies with individual runnables to verify and obtain more detailed information on dependencies as specified in AUTOSAR
- Interactive dependency visualization per runnable
- Enables event chain analysis across multiple SWCs
- Tracking of signals and their safety levels throughout the system, e.g. to detect OBD-relevant parts

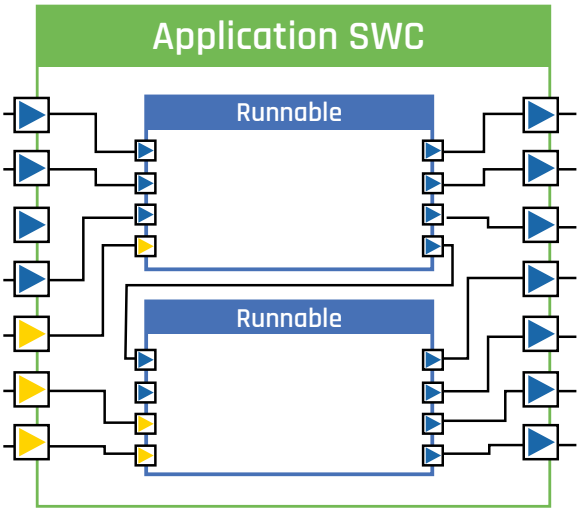
AUTOSAR Software Stack



The emmtrix analysis tools ePE and eDA are designed to analyze C code on the Application Layer in the form of runnables that are part of Application Software Components (SWCs) and make up the functional part of applications. They can be considered as running isolated from the rest of the system as all interaction with other runnables is handled through function calls to the RTE.

The entry point for analysis or optimization is typically a runnable, specifically the C function that implements the runnable's functionality. As the RTE can come from different vendors and not all developers in the supply chain have access to it, we generate our own custom RTE from the AUTOSAR specification to facilitate the analysis of the runnables. This covers all relevant topics such as data types, required functions and macro definitions to ensure correct analyzability of the source code.

Structure of a Software Component



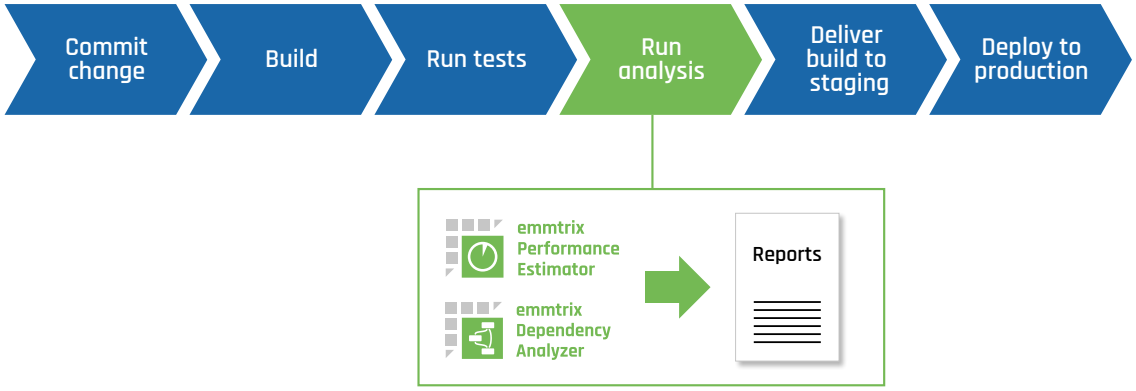
Automated Analysis in Your CI/CD Workflow

Enhanced Support for Automotive

Many modern software development methodologies rely on the use of CI/CD workflows (Continuous Integration/Continuous Delivery). The aim of such a workflow is to automate the process of building, testing and deploying software, so that changes to the source code in the version control system either fail quickly or are quickly pushed to test and staging environments for thorough testing.

The emmtrix tools ePE and eDA are designed to integrate directly into such flows to provide developers with additional information throughout the development process. A setup in a CI/CD pipeline based on tools like Jenkins might look like this:

Automated Analysis Integrated in CI/CD Workflow



An additional 'run analysis' pass can be added after the build step. Compilable source code is a prerequisite for analysis in both tools.

Available Reports

- Interactive HTML reports, which can be viewed directly as results of Jenkins in the browser
  - For ePE: searchable and sortable with options to hide or show items
  - For eDA: interactive visualisation that highlights relevant dependencies
- Results in XML and JSON formats for automated processing, such as ingestion into databases or use in other tools
- Detailed log files that record all analysis steps with warnings and other important information that can be used for further insight into the analysis

Some Benefits of Using emmtrix Tools in CI/CD Workflows

- Can be used early in development to track projects through their lifecycle and monitor how performance or dependencies change over time. This helps to pinpoint high-impact code changes and get a handle on application timing.
- (Function) developers can simply access the results in their browsers
- Automated monitoring of thresholds can automatically notify designers or architects when runtimes become too high
- Reporting can be focused on parts of the application to provide immediate feedback on the impact of code changes

Code Vectorizer

The Tool to Vectorize Your Application

The vector units of high-performance microcontrollers promise to speed up the execution of data-parallel applications based on linear algebra by factors greater than 10. Programming such accelerators manually is challenging because it requires deep knowledge of their instruction set and microarchitecture. emmtrix Code Vectorizer (eCV) is your solution to simplify this task significantly.

AURIX™ TC4x

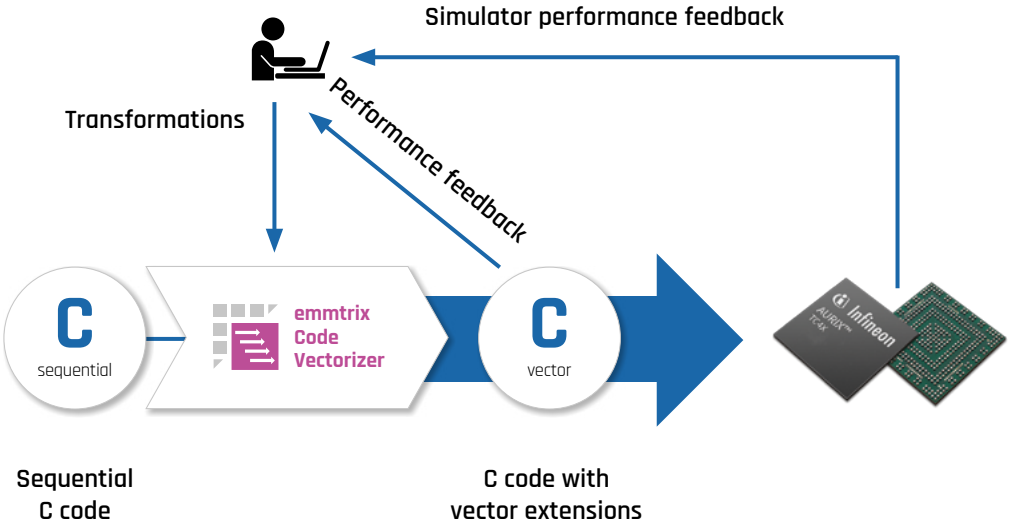
Features

- Functional testing of vector code independent of target platform
- Code transformations improving data level parallelism and optimizing code for vectorization
- Integration of target platform simulators for performance estimation
- Vectorization-aware code generation from Simulink® models
- Code Fusion: block-crossing vectorization of Simulink® models
- Generation of C code with vector extensions using generic libraries or target specific intrinsics

Benefits

- No need to write vectorized code manually
- Easy exploitation of parallel vector hardware
- Limited hardware knowledge required
- Reduced testing effort
- Functional testing without hardware
- Short development cycles

Vectorization Workflow

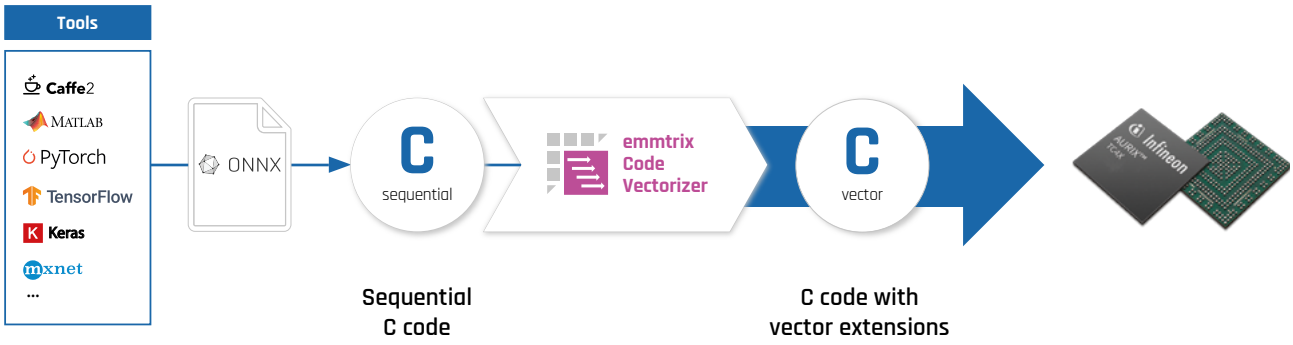


AI Workflow: Vectorization of ML Models

Automated Optimization of ONNX Models

The vectorization flow is particularly useful for AI applications. Machine learning (ML) models are often implemented as matrix operations, which are well-suited for vectorization. The emmtrix vectorization flow allows to accelerate these operations on the target hardware. You can implement your ML model in your favorite framework like TensorFlow, Caffe2, MATLAB, PyTorch etc. and export it to the ONNX format. The ONNX model is then converted into sequential C code which is used as input for the emmtrix Code Vectorizer. The resulting vectorized C code can be either simulated or run on the target hardware to get feedback on the performance and try out different vectorization strategies. Finally, the resulting code can be integrated into your application.

Optimize AI with emmtrix - Vectorize ML Models for Hardware Acceleration



Features

- Automated handling of ONNX2C output
- Automatic use of code transformations to improve performance
- Support for arithmetic functions like sine, tan, tanh, log, etc. Continuous updates in development
- Vectorization-aware code generation from Simulink® models
- Automated handling of unsupported constructs (masking, moving, fallback solutions)
- Configurable accuracy of calculations (ulp)

Benefits

- Easy exploitation of parallel vector hardware or similar accelerators
- Easy switch to different platform
- No need to write vectorized code manually
- No hardware knowledge required

Parallel Studio

The Tool for Parallel Programming

emmtrix Parallel Studio (ePS) helps you to optimize the performance of your embedded applications on multicore, manycore and DSP architectures as well as any combination of these processing units. Our tool automates and radically simplifies the parallelization process to the point where you simply need to take a few decisions to get good results. The patented graphical user interface (GUI), together with a number of reports, provides full transparency and leaves you in complete control at every step of the process.

Use your existing C code as starting point for the parallelization in ePS. Together with ePS Qualification Kit the parallelization can be performed for applications with functional safety requirements like ISO 26262 or DO-178C.

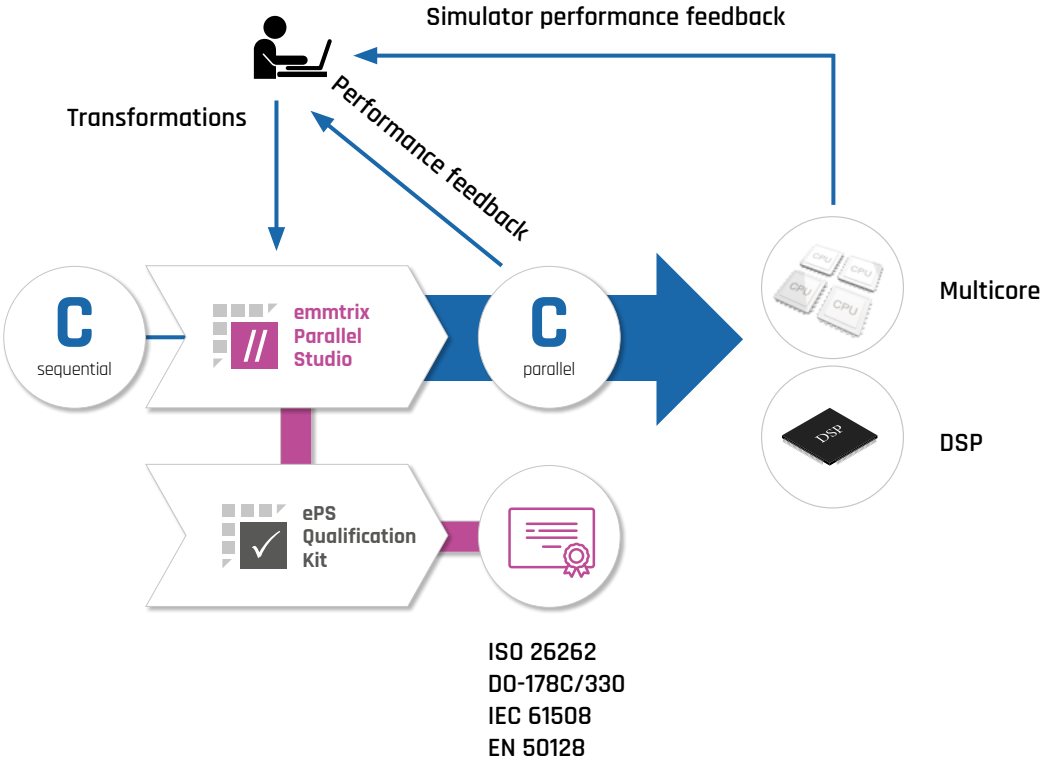
Features

- Automated generation of parallel code
- Interactive optimization with user-friendly Eclipse-based GUI
- Interactive code transformations to optimize parallel code
- Parallelization on runnable-level or function-level and sub-function-level
- Direct deployment of the parallelized program to evaluation boards

Benefits

- Improved application response time and processing throughput
- Correct-by-design approach
- Integrated functional tests for sequential and parallel code (ISO 26262 and DO-178C)
- Easy workflow integration

Parallelization Workflow



C++ to C Compiler

Support for C++ Code

emmtrix C++ to C Compiler (eCPP2C) automatically translates your C++ source code into analyzable C code. The design goal was to keep the binary compilation of the original C++ code and the binary compilation of the translated C code mostly identical. This guarantees the functional correctness of the generated C code. eCPP2C utilizes the LLVM/Clang compiler technology to enable support of the latest features of the fast evolving C++ standard. In combination with emmtrix Parallel Studio, eCPP2C enables software parallelization of C++ applications.

Features

- Translation of C++ to C source code
- Utilizes latest LLVM/Clang compiler technology
- Gurantees functional correctness of generated C code by verification tool
- eCPP2C Qualification Kit (ISO 26262, DO-178C/330 or any comparable standard) can be provided on request
- Demystifies how your C++ code is compiled to assembler
- Can be used in combination with (certified) C compilers and C code analysis tools
- Is integrated into emmtrix Parallel Studio GUI to enable C++ code parallelization

Our Services

Technical Consulting

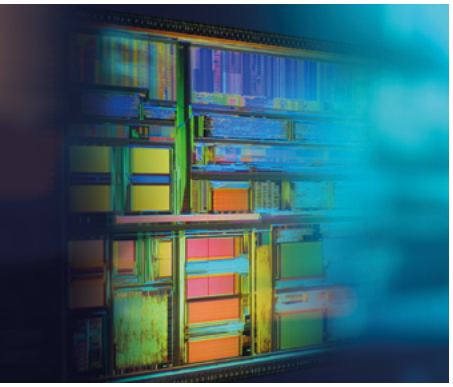
- Performance optimization for single-core architectures (e.g. cache optimization, floating-to-fixed-point conversion)
- Deployment of applications on multicore architectures and DSPs
- Evaluation and selection of appropriate single- and multicore architectures individually and with DSP accelerators if applicable

Integration & Tool Customization

- Customization of emmtrix tools for your target domain's requirements
- Individual interfaces for the seamless integration of emmtrix tools into your existing workflow
- New product features on demand
- Support for your target architecture of choice (i.e. multicore and DSP)

Trainings & Support

- User training for all our software packages
- Customisable installation and integration support
- Individual trainings upon request



emmtrix Technologies GmbH
Erbprinzenstraße 4-12
Entrance A
76133 Karlsruhe / Germany

Phone: +49 721 1803-2880
E-Mail: contact@emmtrix.com

www.emmtrix.com

emmtrix Technologies

Last updated: March 2025