trustmotion
part of NXP

MotionWise
schedule

Take software integration to the next level and safeguard your critical workloads with our solution

For further information, including price and availability, contact
products@tttech-auto.com

TTTech Auto GmbH
Austria - HQ
Phone: +43 1 585 65 38-5000

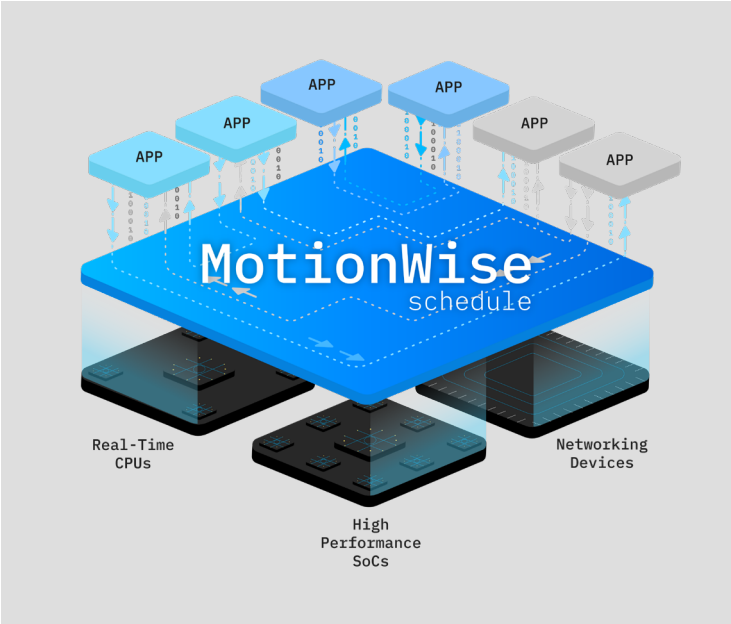
Germany
Phone: +49 841 88 56 47-0

South Korea
Phone: +81 31 707 8895

China
Phone: +86 189 6476 2518

MotionWise Schedule is a software product designed for Software Defined Vehicles (SDVs), with a focus on safety- and time-critical vehicle functions.

- Built on 20+ years of expertise and technologies
- Comes with rich toolset and embedded functions
- Enables safe and efficient execution of mixed-criticality applications within and across MCUs and SoCs
- Automates application integration and schedule configuration
- Designed for highest safety and security (ISO 26262 ASIL D and ISO/SAE 21434 CAL 4)



TOOLING

MotionWise Creator	Global Scheduler	Schedule Visualization

RUNTIME

Applications		
Time-Triggered Scheduling	Dataflow-Driven Execution	Other Middleware SW
Runtime Tracing	Computation Chain Monitoring*	Task Monitoring
Time Synchronization & Monitoring		
ASR OS	Linux	QNX
MCU	SoCs	
Communication Network		

3rd Party SW 3rd Party HW

KEY BENEFITS

Reduce time and cost for software integration

MotionWise Schedule streamlines the configuration and generation process and facilitate the derivation of correct embedded configurations, not only for MotionWise Schedule, but also for other components such as operating systems and network devices.

Ensure end-to-end latencies for global computation chains

MotionWise Global Scheduler creates a consistent and compatible schedule configuration for each CPU in the system. In addition, system requirements such as latency, reliability, deadlines and safety are always met, ensuring timely and accurate system behavior.

Temporal Freedom from Interference between applications

For system in which time determinism is of utmost importance, the scheduler of MotionWise Schedule guarantees freedom from interference for applications in the temporal domain.

Simplifies verification & validation with a correct-by-design approach

With its "correct-by-design" approach, MotionWise Schedule ensures that timing constraints and resource requirements for existing safety and time-critical workloads are fulfilled if new applications are added to the ECU. This significantly reduces the effort required in the system verification and validation phase of new SW releases.

FEATURES

Global Scheduling

MotionWise Global Scheduler follows the correct-by-design approach. It implements scheduling algorithms to automate the integration of complex software functions on high-performance automotive computing platforms. It is designed to create a global deployment solution for the entire set of requirements defined by architects, developers and integrators working on the same vehicle software. Such requirements are:

- System / platform definition (CPUs / cores / network topology)
- Specification of tasks and their dependencies
- Timing requirements (time budget, deadlines, jitter, etc.)

It applies a Time-Triggered Architecture as a basis, shifting the complexity from an explosion of possible runtime states to the solution of the configuration problem. It integrates different computational models into a single, unified framework. Sporadic (event-triggered) workloads that use classical scheduling approaches such as priority-based scheduling are integrated with the time-triggered approach.

It is powered by intelligent, highly parallelized heuristic algorithms and solves more than 80% of complex cases (10^{5000} of possible configurations) within 200 seconds.

Time-Triggered Scheduling

MotionWise Schedule provides a time-triggered scheduling policy for periodic, time-critical workloads with strict timing requirements.

For POSIX systems, MotionWise Schedule implements a time-triggered scheduler. Applications are executed according to the schedule tables that are created with MotionWise Creator at design time according to user requirements.

For Classic AUTOSAR-based microcontrollers, MotionWise Schedule generates the configuration for the operating system scheduler to achieve periodic, time-triggered execution. During runtime, the runnables of the software components are executed according to the schedule tables created during the platform configuration phase.

Activity Sequencing (Dataflow-Driven Execution)

MotionWise Schedule provides a dataflow-driven execution solution on microprocessors, also known as Activity Sequencer. It is a suitable solution for Directed Acyclic Graphs (DAG), enabling multithreading and parallel execution of interdependent activities on multiple CPU cores. This is a typical use case in ADAS/AD systems that process multiple input data from different sensors.

The Activity Sequencer executes application threads according to a predefined data dependency.
The Activity Sequencer is a user space scheduling policy that supplements the time-triggered scheduler. This allows flexibility during development as the execution can be easily customized without affecting the schedule of other applications. Application developers can define and change the behavior of the application without having to consider the higher-level system schedule.

The Activity Sequencer supports a single process or several separate processes, e.g. to accommodate different application architectures or to enable mixed-criticality graph (activities with different safety requirements).

Event-Driven Scheduling*

With the introduction of Event-driven scheduling, MotionWise Global Scheduler allocates resources for both periodic and sporadic workloads at design time. This approach ensures that workloads with real-time requirements meet their deadlines, even if the exact activation time during integration is unknown.

Such event-driven tasks are executed when a user-defined event occurs, e.g. when camera images are received by the ADAS/AD controller. They typically have a minimum interarrival time that enables the associated event-driven tasks to be scheduled.

Computation Chains

A computation chain is a set of time-critical tasks that have data dependencies, where the entire chain must be started and completed within a predefined time interval (end-to-end latency). Such chains are created using MotionWise Creator. The schedule tables are created to meet the sequence and latency requirements.

MotionWise Schedule has no constraints on the length or number of the computation chains. For example, a chain can be triggered by an incoming vehicle signal or any time-triggered task and can span multiple CPU cores. The tasks involved can have different time periods, while the entire Computation Chain itself is executed periodically.

Computation Chain Monitoring*

The execution order and end-to-end latencies are often safety critical. MotionWise Schedule provides a runtime safety mechanism to monitor whether tasks are executed as planned and the specified latency is maintained, otherwise, it raises an error to trigger safety and recovery reactions.

Task Monitoring

MotionWise Schedule provides Task Monitoring on microprocessors that automatically monitors application tasks as they are executed. It includes aliveness and deadline monitoring. The Task Monitoring configuration is derived from the input configuration provided by the user and generated by MotionWise Creator.

On microcontrollers, Task Monitoring is ensured by the correct configuration of the AUTOSAR Watchdog Stack, which is part of the Classic AUTOSAR BSW. MotionWise Creator provides the input configuration for third-party tools to generate the configuration of the Classic AUTOSAR stack.

Time Synchronization (Ethernet, Shared Memory)

The MotionWise Schedule Time Synchronization component provides common, synchronized time bases for:

- All applications within the system in which MotionWise Schedule is deployed.
- Other ECUs, sensors and actuators.

It is implemented according to the AUTOSAR TSP standard with the management of multiple different time bases. It supports gPTP-based synchronization via Ethernet and shared memory (if a microcontroller and an application processor are on the same chip).

Time Synchronization Monitoring

In the scope of an ECU (MCU, SoC, Ethernet switches), MotionWise Schedule provides safe time synchronization (ASIL D). This is achieved through a Time Synchronization Monitoring mechanism and the typical accuracy of time synchronization is less than 100 us.

SOA LogCollector

MotionWise Schedule implements its own log message collection mechanism on top of POSIX OSs, with a simple API for observing debug- and diagnostic information that are provided by the MotionWise Schedule components running on POSIX OS. The release contains pre-implemented adapters for the integration with DLT standard logging stack or printf.

TOOLS

The MotionWise Schedule Toolset consists of the software applications used to configure, generate and test the behavior of the in-vehicle stack.

MotionWise Creator

MotionWise Creator is an essential tool for the configuration, creation, integration and deployment of MotionWise Schedule in a project-specific system. It consists of frontend tools (client) and a cloud-based backend. To support the needs of all software developers, the frontend provides a Graphical User Interface (GUI)* as well as the ability to use it directly via the Command Line Interface (CLI) on Windows and Linux.

While the CLI is intended for advanced users and integration into the Software Factory (CI/CD) on the user side, the GUI guides first-time users and ensures correct configuration through live validation checks and notifications on-demand notifications.

MotionWise Schedule is designed for collaboration between team members: Users can work simultaneously on the front-end and back-end and regenerate the schedule configuration on either the branches or the main development line.

Schedule Visualization

Schedule Visualization is a feature/set of tools that measure and analyze the runtime of customer applications executed by MotionWise Schedule in-vehicle stack.

Once the schedule tables have been generated, the toolset allows the user to view and analyze the expected (planned) execution of the specified tasks and visualize the computation chains.

Schedule & Computation Chain Viewer

The Schedule & Computation Chain Viewer provides valuable input for system and software integrators, enabling the refinement of schedule configuration and the optimization of resource consumption.

TracePoint Converter

MotionWise Schedule collects trace events from the system during runtime. This data is used to visualize runtime behavior.

Users can compare the actual execution with the planned schedule (i.e. the schedule tables generated at design time) in the Schedule Viewer tool.

SUPPORTED ARCHITECTURES

Typical Deployment Scenarios

MotionWise Schedule supports architectures with one or more hosts. A “host” is a combination of hardware and software (e.g. the Operating System). MotionWise Schedule distinguishes between two main types of hosts:

- Microcontroller-based: a stand-alone MCU or the safety island of a complex SoC. It executes the Classic AUTOSAR software stack and an OSEK operating system.
- Microprocessor-based: The performance CPU core(s) of an SoC. It executes a POSIX operating system and provides more computational capabilities than a microcontroller. Microprocessor-based hosts can be deployed with or without Adaptive AUTOSAR.

Example deployment scenarios of Domain Control Units (DCU) and High-Performance Compute (HPC):

- 1 uC + 1 uP
- 1 uC + 2 uPs
- Single uP

Communication between hosts may be Ethernet or shared memory.

Supported 3rd-party Components

MotionWise Schedule supports a variety of widely adopted 3rd party components.
It integrates well with standard automotive software stacks, as presented in the table below.

In addition, MotionWise Schedule offers the flexibility for customers to choose other out-of-the-box options.

	MICROCONTROLLER / MCU	MICROPROCESSOR / SOC
AUTOSAR COMPATIBILITY	Vector MICROSAR Classic stack	Adaptive AUTOSAR as well as non-AUTOSAR based POSIX systems (both are supported)
OPERATING SYSTEM COMPATIBILITY	Vector MICROSAR OS	QNX OS 7.1 / QOS 2.2x Linux
SEMICONDUCTORS	▪ ARM Cortex M7 (NXP S32G2 / G3) ▪ R5F (e.g. TI TDA4) ▪ TriCore (e.g.: TC39x)	ARMv8 and x86-64, e.g.: ▪ ARM Cortex A53/A55/A72 (e.g. S32G, HR J5, TDA4) ▪ Kryo Gen6 (e.g. Qualcomm SA8xxx) ▪ Intel x86-64

* *Coming in 2026 on Microprocessors/SoCs*