

MICROSAR RTE

Die optimierte Ablaufumgebung für Softwarekomponenten nach dem AUTOSAR-Standard

Die Vorteile im Überblick

- > Leicht konfigurierbar und skalierbar
- > Tiefgehende Konsistenzprüfung der Konfiguration
- > Hoch optimierter Code mit intelligenten Synchronisationsmechanismen
- > Schneller Einstieg in AUTOSAR durch z.B. generiert Code Templates für die Softwarekomponenten (SWCs)
- > Für Migrationsprojekte geeignet

Die MICROSAR RTE (Run Time Environment) ist die skalierbare und hoch optimierte AUTOSAR-Laufzeitumgebung von Vector. Die RTE ist ein von AUTOSAR eingeführtes Modul und verwaltet die Kommunikation zwischen den Softwarekomponenten (SWCs). Sie stellt dabei die Konsistenz des gesamten Informationsflusses sicher und bildet die Schnittstelle zwischen Funktionssoftware, Basis-Software (BSW) sowie Complex Device Drivern (CDD). Die SWCs, RTE, CDD und BSW-Module bilden zusammen die gesamte AUTOSAR-Steuergerätesoftware.

Anwendungsgebiete

Wenn die Funktionssoftware eines Steuergeräts über AUTOSAR-konforme SWCs implementiert ist, benötigt der Anwender die RTE als Laufzeitumgebung. Dieser modulare Aufbau der Steuergerätesoftware bietet dem Anwender maximale Flexibilität: Eine manuell entwickelte oder durch modellbasierte Werkzeuge entworfene SWCs kann in mehreren Steuergeräte-Projekten verwendet werden.

Hierzu ist lediglich für das konkrete Steuergerät das Umkonfigurieren und die neue Generierung der RTE sowie ggf. der BSW-Module erforderlich. Darüber hinaus ist auch die Mehrfachverwendung einer SWC auf einem Steuergerät möglich.

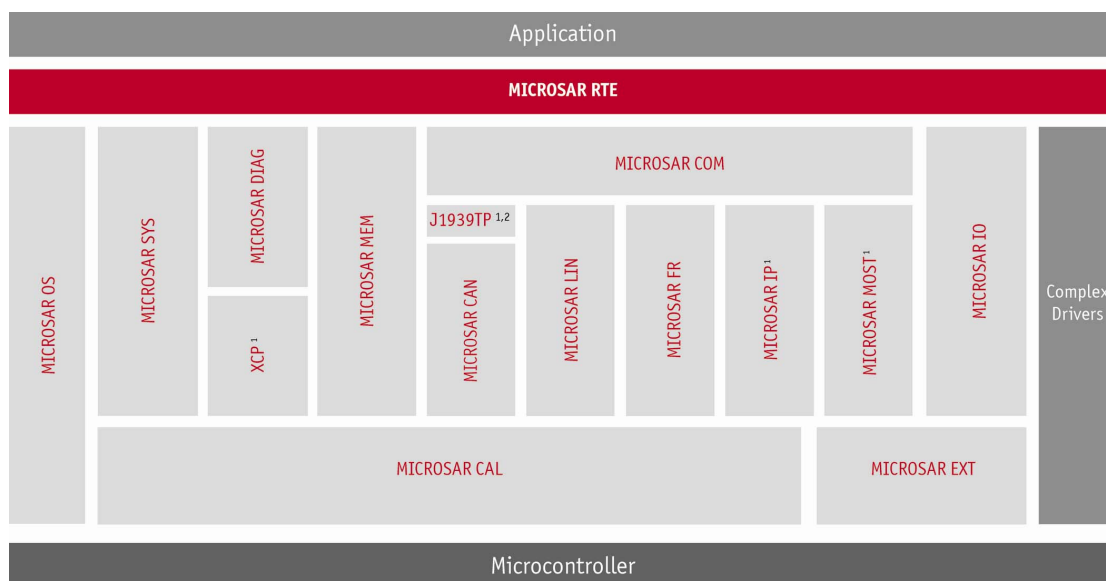
Beim Generieren der MICROSAR RTE kann der Anwender zwischen zwei Modi wählen:

- > Contract Phase Generierung für die Entwicklung einzelner SWCs in einer frühen Phase. Hierbei erstellt der Generator anstelle der gesamten RTE nur eine Header-Datei für jede SWC. Damit ist es möglich die SWC einzeln zu kompilieren um sie z. B. als Objektcode an den Entwicklungspartner weiterzugeben.
- > RTE-Generierung für die gesamte Steuergerätesoftware. Der in diesem Modus generierte Code ist hoch effizient und benötigt wenig Speicherplatz. Er ist auf die gesamte Steuergerätekonfiguration optimiert und belastet aufgrund geringer Laufzeit und minimaler Interrupt-Sperrzeiten die System-Ressourcen nur wenig. Dies wird z.B. erreicht durch die Verwendung intelligenter Synchronisationsmechanismen, die auf die Eigenschaften der verwendeten Hardware abgestimmt sind.

Funktionen

Die MICROSAR RTE enthält die in der AUTOSAR Releases 3.x definierten Funktionen:

- > Sender/Receiver und Client/Server Kommunikation
- > Mode Management
- > Inter-Runnable Variablen sowie Exclusive Areas



¹ Available extensions for AUTOSAR 3.0

² BAM and CMDT Option available

**MICROSAR RTE
Modul**

Schulungen

Im Rahmen unseres Schulungsangebotes bieten wir Ihnen für MICROSAR verschiedene Schulungen und Workshops in unseren Seminarräumen sowie inhouse bei Ihnen an.

Mehr Informationen zu den einzelnen Schulungen und die Termine finden im Internet unter www.vector-academy.de

Kontakt und Verfügbarkeit

Die Verfügbarkeit der hardwarespezifischen MICROSAR BSW-Module finden Sie unter: www.micosar.de/availability/

Wir informieren Sie gerne über OEM-spezifische Ausprägungen sowie individuelle Unterstützung Ihrer AUTOSAR Projekte.

Email: embedded@de.vector.com

Telefon: +49 711 80670-400

- > Trigger für Runnables
- > Unterstützung von Kalibrier-Parametern und Per-Instance Memory
- > Mehrfach Instanzierung von SWCs

Darüber hinaus bietet die MICROSAR RTE:

- > Generieren von Code-Templates für die SWCs auf Basis der XML-Datei „SWC Description“. Diese Templates enthalten alle APIs der RTE. Der Anwender füllt sie mit seinem Anwendungscode.
- > Verwenden von Speicherschutzmechanismen, wie im AUTOSAR-Betriebssystem spezifiziert. Besonders optimiert ist diese Unterstützung beim Einsatz von MICROSAR OS, das AUTOSAR-Betriebssystem von Vector.
- > Konfiguration von Initialisierungs-Runnables für das AUTOSAR-Konzept der „Mode-abhängigen Runnables“
- > Generieren einer A2L-Datei zur einfachen Anbindung an bestehende Kalibrier- und Diagnose-Standards
- > Erzeugen eines HTML Reports mit den Eigenschaften der RTE. Darin sind Informationen wie z.B. der berechnete RTE Ressourcenbedarf (RAM + Konstanten) enthalten.

Folgende AUTOSAR 3.x Funktionen sind optional erhältlich:

- > Externe Client/Server Kommunikation (Inter-ECU)
- > Vollständige Unterstützung für Messen und Kalibrieren
- > Empfangsdaten-Filter für die Funktionssoftware

Konfiguration

Sie konfigurieren die MICROSAR RTE mit dem DaVinci Developer. Als Erweiterung zur AUTOSAR-Methode enthält er Validierungsregeln, die die Konsistenz der Konfigurationsparameter der RTE sicherstellt. Die anschließende Generierung der RTE erfolgt direkt aus dem DaVinci Developer oder mit einem separaten Generator“. Ein automatisiertes Generieren der RTE über die Kommando-Zeile ist ebenfalls möglich.

Lieferumfang

- > Kommandozeilen-basierter Generator (für Windows NT/2000/XP/Vista)
- > BSW Module Description
- > Dokumentation und Beispielprogramme mit Makefiles

Die komplette AUTOSAR-Lösung von Vector

Die AUTOSAR-Lösung von Vector besteht aus den DaVinci Werkzeugen, der MICROSAR-Basissoftware und der MICROSAR RTE.

Die Eigenschaften der BSW-Module aus den MICROSAR Paketen sowie den detaillierten Funktionsumfang der einzelnen DaVinci Tools finden Sie in den jeweiligen Datenblättern.

Weitere relevante MICROSAR-Produkte:

Die RTE setzt ein Betriebssystem wie z.B. MICROSAR OS oder osCAN voraus.

```

/*****
 * Runnable Entity Name: Runnable10ms
 *****/
-----
* Executed if at least one of the following trigger conditions occurred:
* - triggered on TimingEvent every 10ms
-----
* Input Interfaces:
* =====
* Explicit S/R API:
* -----
* Std_ReturnType Rte_Read_SensorData_Speed(UInt16 *data)
* Std_ReturnType Rte_Read_SensorData_Temperature(SInt32 *data)
* Client/Server Interfaces:
* =====
* Server Invocation:
* -----
* Std_ReturnType Rte_Call_TemperatureControl_Action(SInt32 temp)
* Synchronous Server Invocation. Timeout: None
*
-----
FUNC(void, RTE_SUCCOMPONENT_APPL_CODE) Runnable10ms(void)
{
/*****
 * DO NOT CHANGE THIS COMMENT! << Start of runnable implementation >> DO NOT CHANGE THIS COMMENT!
 * Symbol: Runnable10ms
 *****/
-----
    UInt16 speed; SInt32 temp; UInt8 action;

    /* read input data from port prototype SensorData */
    Rte_Read_SensorData_Speed(&speed);
    Rte_Read_SensorData_Temperature(&temp);

    action = CalculateAction(speed, temp); /* calculate the action depending on the input data */
    if (action != 0)
    { /* invoke Action operation of TemperatureControl port prototype by a Client-Server call */
        Rte_Call_TemperatureControl_Action(temp);
    }
/*****
 * DO NOT CHANGE THIS COMMENT! << End of runnable implementation >> DO NOT CHANGE THIS COMMENT!
 *****/
-----
}

```

**MICROSAR RTE
ermöglicht die
Generierung von Code
Templates für die SWCs**