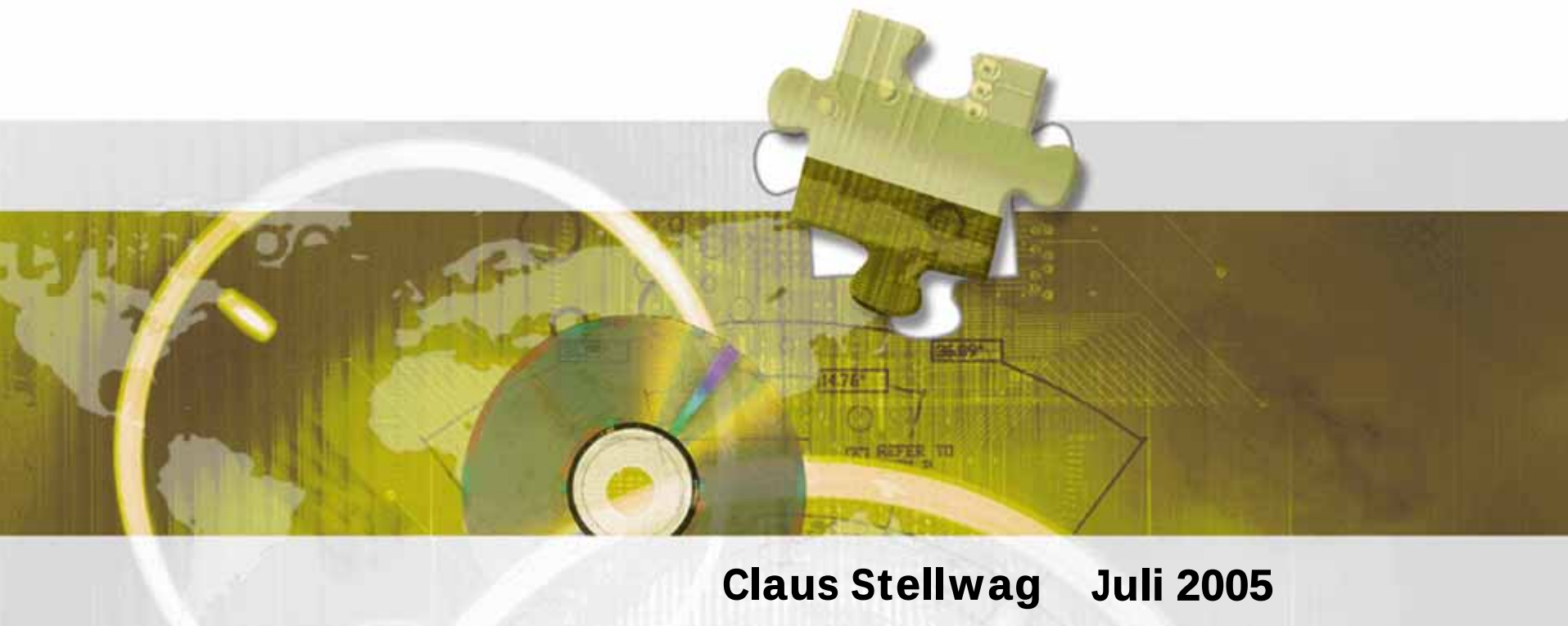
3SOFT

# Sichere Kleinstsysteme mit Speicherschutz (nach IEC 61508)



Claus Stellwag Juli 2005

# Agenda

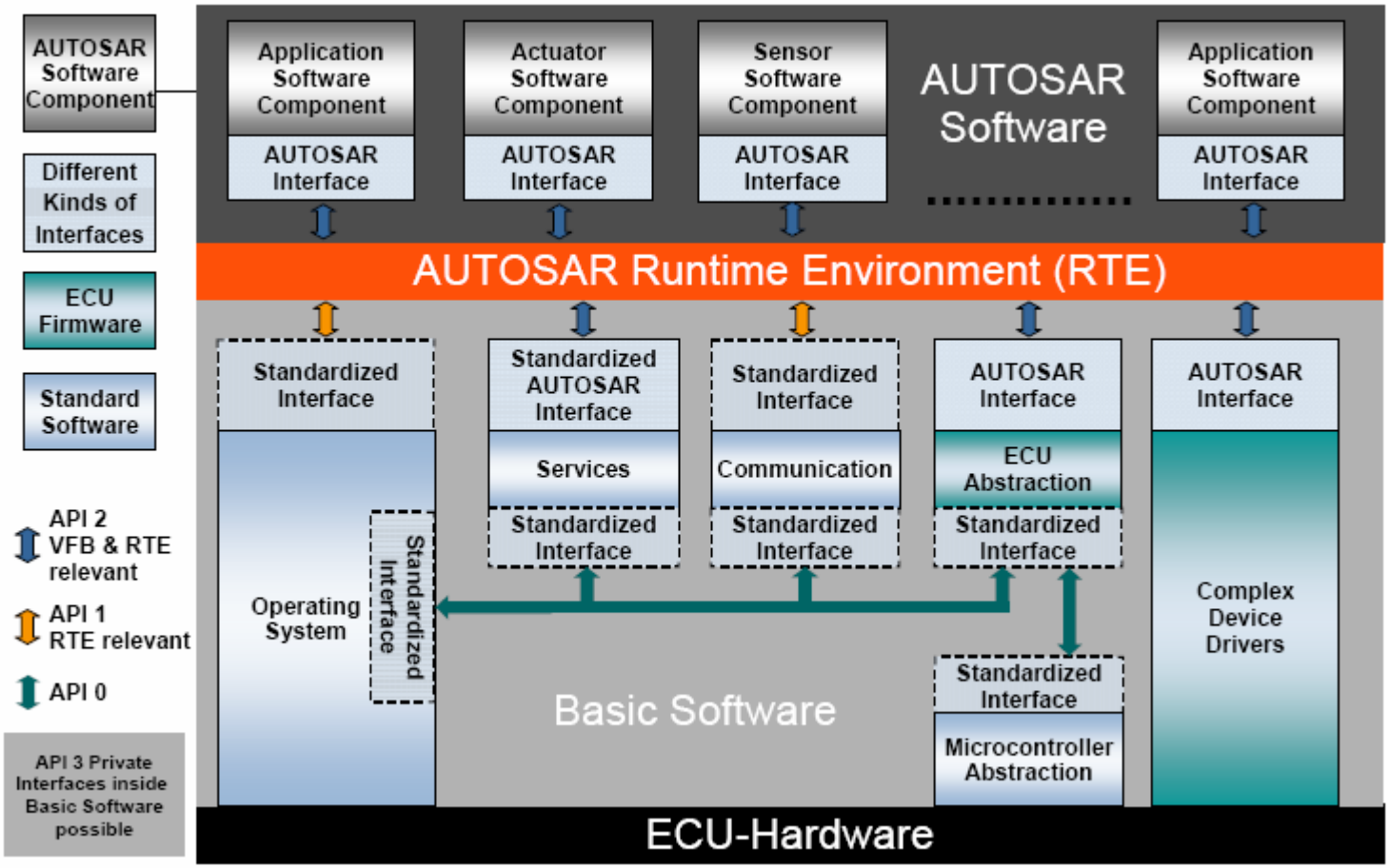
- Entwicklung eines sicheren Echtzeitbetriebssystems (RTOS)
  - Ziele
  - Sicherheit
  - Hardware
  - Umsetzung & Erfahrungen
- Zusammenfassung

Hintergrund: Reales Projekt mit einem Kunden für ein zukünftiges Steuergerät

# Ziele

- RTOS für sicherheitsrelevante Steuergeräte im Automobilbereich - OSEK konform
- Einfache Controller – typischerweise ohne MMU
- Konform zu AUTOSAR
  - Automobilkonsortium mit dem Schwerpunkt der Softwarestandardisierung
  - Definiert eine Standardplattform für In-Car Gebrauch, einschließlich RTOS
  - ... noch in der Entwicklung

# Ziel: AUTOSAR



# Sicherheit

- Was macht ein System sicher?
  - Die Vorgehensweise (Methodik, Prozess)
  - Das Know-how (Reife, Erfahrung) der Projektbeteiligten
  - Die Reife der genutzten Werkzeuge
- Sicherheit wird nicht allein durch das RTOS bestimmt
- Sicherheit ist nur für ganze Systeme realisierbar (HW+SW)
- Sicherheit ist relativ: Wie viele Systemausfälle sind in welcher Zeit tolerierbar?

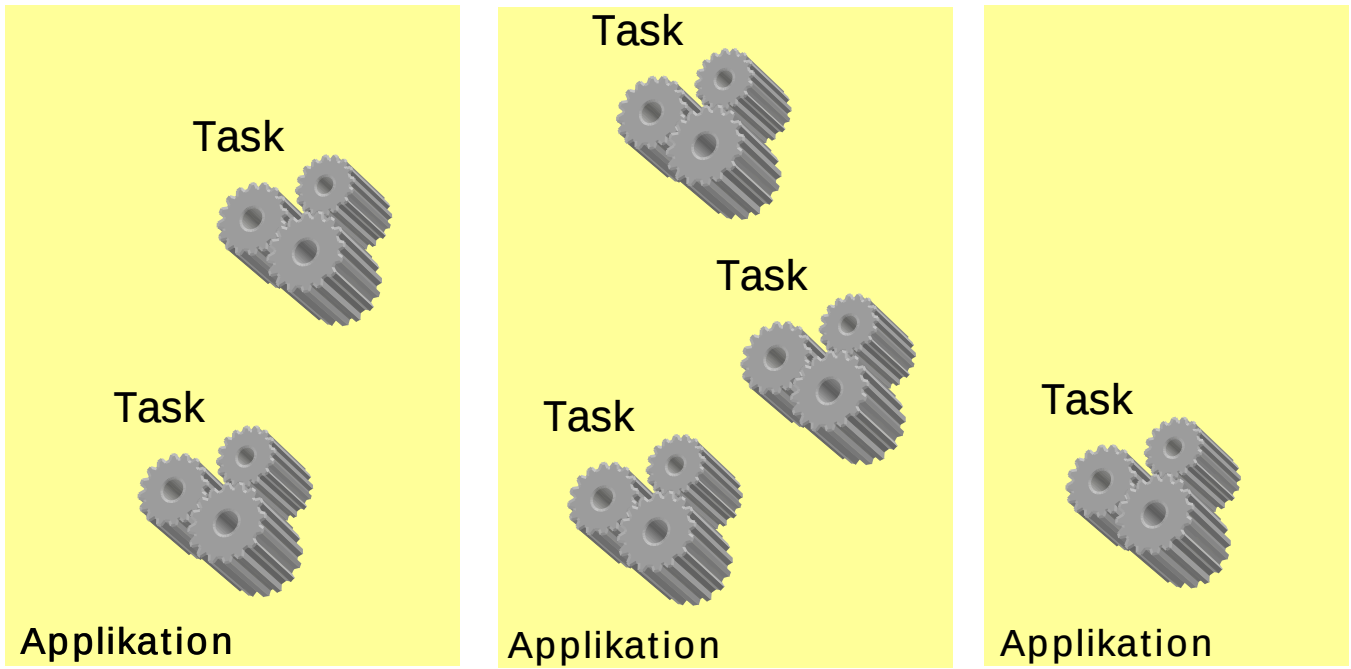
# Sicherheit – IEC61508

- Titel „***Functional safety of electrical/electronic/programmable electronic safety-related systems***“
  - Part 1: General requirements
  - Part 3: Software requirements
- „safety integrity levels“ (SIL 1 - 4) legt die Sicherheitsklassifikation fest.
- Norm deckt den Lebenszyklus des Systems ab (Requirements, Entwurf, Test, Änderungen). Aspekte u. a.:
  - Risikomanagement
  - Change-/Impactmanagement
  - Dokumentenmanagement

# Sicherheit – Problemquellen

- Hardware
  - Verschleiß, Bruch – Lösung: Redundanz, Rückfallebene
  - Bauteilqualität – Lösung: Stringente QM-Prüfung
  - Fertigung
- Software
  - Fehler sind explizit vorhanden: Designfehler, Funktionale Fehler, ...
  - Temporärer Einfluß auf den Programmablauf (Strahlung, Temperatur)
- System: save (silent) fail back

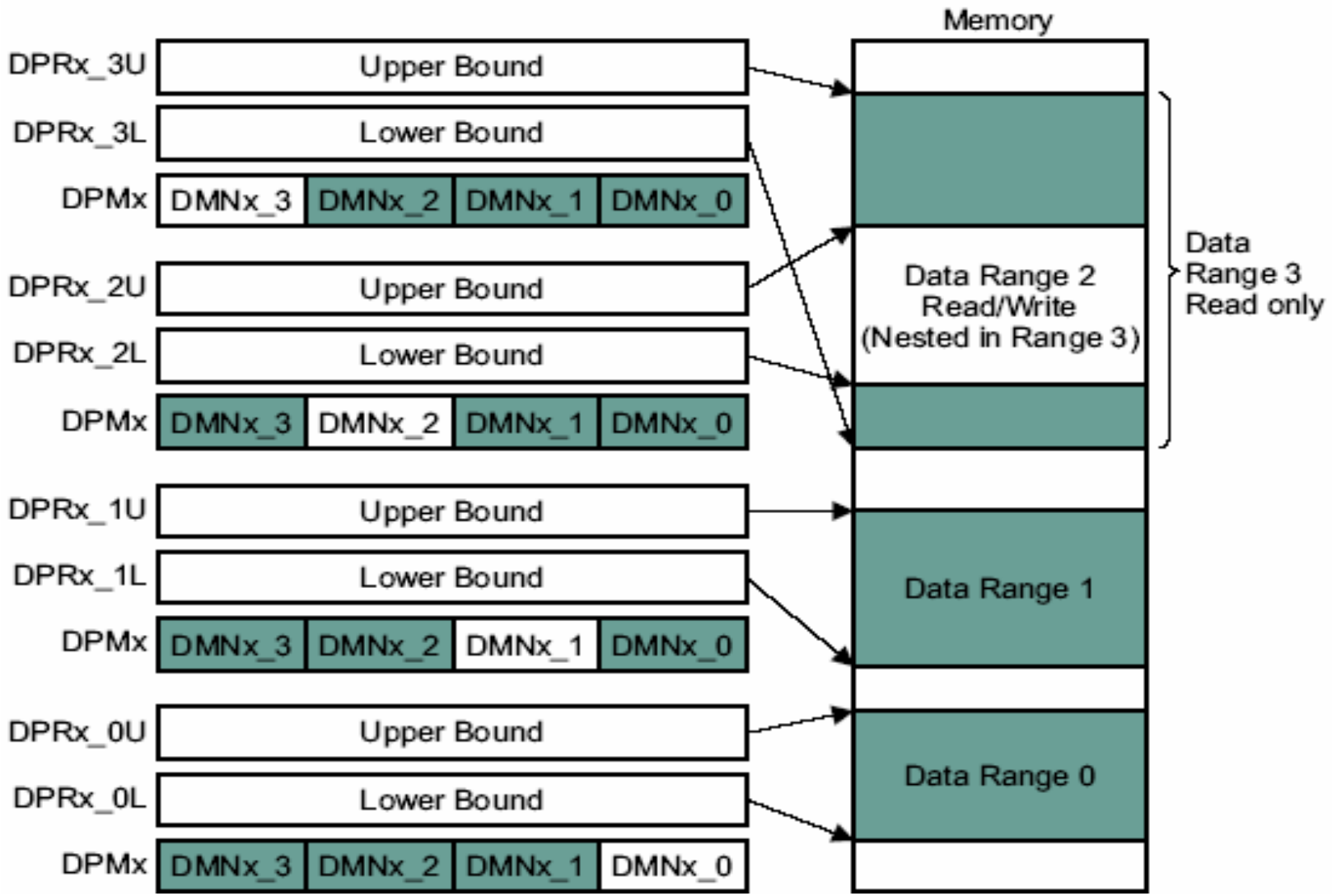
# Sicherheit – OSEK



# Controller – Infineon TriCore

- 32Bit Hochleistungscontroller
- Zielmärkte: Consumer, Automotive, Telekommunikation
- Programmiermodel an 68k angelehnt (Daten- und Addressregister)
- Registersätze werden in Single Linked Lists verwaltet per HW verwaltet
- Controller mit (optionaler) MMU und einer MPU
- 256 Interruptebenen, die auch einen Hardwarescheduler ermöglichen

# Controller – MPU



# ProOSEK/Cert – Konzepte (1)

- Einzelne Applikationen werden gekapselt mit
  - Speicherschutz
  - Überwachung der Laufzeit
  - Überwachung einzelner Services
- Speicherschutz == Schreibschutz im RAM (und Schutz der I/O Register)
- Laufzeitüberwachung mittels Zeitbudgets für Tasks, ISRs, Lockmechanismen
- Trap-Mechanismus anstatt Funktionsaufruf
- Kein virtueller Speicher

# ProOSEK/Cert – Konzepte (2)

- Oberste Priorität: Safety first (nicht RAM, Codesize oder Zeiten)
- SIL Level 3
- *trusted* vs. *non-trusted*
- call trusted functions
- Zuverlässigkeit der Tools (Compiler)
- Library vs. Source? -> Library
- Zertifikat auf
  - Generator
  - Library
- Neuentwicklung

# ProOSEK/Cert – Entwicklung

- Wasserfall-Modell
  - Requirements
  - Design
  - Implementierung (Richtlinien!)
  - Test
  - Änderungszyklen
- Reviews zwischen den einzelnen Stufen
- Einbindung der Prüfstelle (TÜV)
- Hoher Dokumentenaufwand
- Nutzung von Werkzeugen zur statischen Quellcodeananlyse.

# Zusammenfassung

- Sichere Kleinstsysteme sind realisierbar
- Verständnis von Sicherheit klären
- Hardwaresupport entscheidend
- IEC61508
  - Sehr saubere Vorgehensweise erforderlich
  - Prozess sehr aufwendig
  - Festlegung SIL wichtig
- Schreibschutz allein nicht ausreichend

# 3SOFT – Standardizing the Embedded World


**3SOFT** GmbH
Frauenweiherstraße 14 · 91028 Erlangen
Fon +49 9131 7701-0 · Fax +49 9131 7701-333

info@3SOFT.de · www.3SOFT.de