

# Treffen des Arbeitskreises 'Test Eingebetteter Systeme' Erlangen/Tennenlohe 8.Juni 2001

## OSEK -Testprobleme

DAIMLERCHRYSLER

Roman Pitschinetz

Methods and Tools (FT3/SM)

e-mail:

Roman.Pitschinetz@DaimlerChrysler.com

DaimlerChrysler AG

Research and Technology

Alt-Moabit 96 A

D-10559 Berlin

Phone +49-(0)30 - 39982-231

Fax +49-(0)30 - 39982-107

# OSEK - Testprobleme

- Gegenüberstellung Sequentielle/Nebenläufige (OSEK) Programme
- Testaktivitäten:
  - Schnittstelle ermitteln (Parsieren der C-Quelle)
  - Testdaten (Generierung, Import, Editor, ggf. Sollwerte)
  - Testtreiber
  - Instrumentierung (Monitoring)
  - Teststand
  - Teststeuerung, Testdurchführung auf der Zielplattform
  - Testauswertung und Testdokumentation

# Sequentielle/Nebenläufige (OSEK) Programme

- Sequentielles Programm
  - Zyklisches Zeitscheibenverfahren mit fester Folge von Funktionsaufrufen.
  - Funktion ist das Testobjekt.
  
  - Globale Daten.
  - Mehrfachzugriff auf globale Daten ausgeschlossen.
- Nebenläufiges Programm (OSEK)
  - Interrupt basierter Taskablauf; Signal (Event) kann zu jeder Zeit wirksam werden.
  - Task ist das zentrale Objekt.
  - Unabhängige Tasks konkurrieren um Prozessorzuteilung.
  - Kommunikation der Tasks über Events.
  - Priorität, preemptiv, non-preemptiv, Schedulig-Strategie, Synchronisation.
  - Datenkapselung.
  - Bei globalen Daten ist Mehrfachzugriff nicht ausgeschlossen.

# Feststellung der zum Test relevanten Schnittstelle

## ● Sequentielles Programm

- Parsierung der C-Quelle(n) inklusive aller Header-Dateien.

⇒ ANSI-C Parser.

- ➊ Funktionsnamen, Subfunktionen.
- ➋ System-globale und Modul-globale Variablen, Funktionsparameter, extern deklarierte Variablen.
- ➌ Passierrichtung.

## ● Nebenläufiges Programm (OSEK)

- Parsierung der C-Quelle(n) inklusive aller Header-Dateien.

⇒ Erweiterter ANSI-C Parser zur Erkennung OSEK spezifischer Konstrukte (Task, 4...).

- Auslesen von Informationen aus OIL (5).

- ➊ Tasknamen, Funktionen.
- ➋ System-globale und Task-globale Variablen, extern deklarierte Variable.
- ➌ Passierrichtung.
- ➍ Events, Alarme, Counter, Nachrichten, ISR.
- ➎ Task-Anzahl, Task-Typ, Task-Aktivierung, Task-Priorität, Scheduling-Strategie, Task-Event-Beziehung.

# Testdaten

- Sequentielles Programm
  - Konkrete Werte für globale Variablen und Funktionsparameter.
  - Stubs.
  - Testdatengenerierung zur Ermittlung der längsten/kürzesten Laufzeit.
- Nebenläufiges Programm (OSEK)
  - Konkrete Werte für globale Variable.
  - Setzen von Events über:
    - Task
    - Nachrichten
    - Alarm
    - ISR
  - Einstellung von Ablaufszenarien:
    - zeitabhängig
    - datenabhängig
  - ⇒ Überprüfung der:
    - Synchronisation
    - Prioritätsverhalten
    - Zeitverhalten
  - Stubs

# Testtreiber

- Sequentielles Programm
  - Bestückung der kompletten Schnittstelle mit Testdaten.
  - Funktionsaufruf.
  - Ggf. Auslesen der Istwerte für den Soll-/Istvergleich bei der Testauswertung.
  - ☒ Realisierung des Treibers in eigener C-Quelle/Modul.
- Nebenläufiges Programm (OSEK)
  - Bestückung der kompletten Schnittstelle mit Testdaten.
  - Taskaktivierung (bei Systemaufruf oder im Treiber -> `ActivateTask(T1)`).
  - Zeit- und zustandsabhängiges Auslösen von Events.
  - ☒ Realisierung des Treibers als Task zur Aktivierung von Tasks, Auslösung von Events, Steuerung des Testdurchlaufs.
  - ☒ Andere Alternativen?

# Instrumentierung (Monitoring)

- Sequentielles Programm
  - Einfügen von Zählern bei der zu testenden Funktion und ggf. in den Subfunktionen.
  - ⊗ Zeitablauf verändert, logischer Ablauf mit nicht instrumentierten Ablauf i.d. Regel gleich.
- Nebenläufiges Programm (OSEK)
  - Einfügen von Zählern in der zu testenden Task.
  - Einfügen von Zählern in allen Tasks.
    - ⇒ Nicht deterministischer Ablauf.
  - Instrumentierung für Taskablaufverfolgung?

# Teststand, Teststeuerung und Testdurchführung

- **Sequentielles Programm**
  - Übersetzen und Binden des Treibers, Testobjekt-Quelle, Stub-Quelle, Bibliotheken.
  - Start des Teststands, Aufruf der durchzuführenden Testfälle und Testschritte, Versorgung mit Testdaten über eine entsprechende Kommunikationsschnittstelle.
  - Testdurchführung auf dem Entwicklungsrechner und/oder der Zielplattform.
- **Nebenläufiges Programm (OSEK)**
  - Übersetzen und Binden des Task-Treibers, Task-Quellen, Stub-Quelle, Bibliotheken.
  - Start des Task-Treibers, Aufbau der Kommunikation zum Testwerkzeug.
  - Testdurchführung nur auf OSEK-fähigen Plattformen.

# Testauswertung und Testdokumentation

- Sequentielles Programm
  - Soll-/Istvergleich der globalen Variablen und der Funktionsparameter.
  - Auswertung der Struktur-Überdeckung.
  - Auswertung der konsumierten Zeit.
- Nebenläufiges Programm (OSEK)
  - Soll-/Istvergleich der globalen Variablen und der Funktionsparameter.
  - Auswertung der Struktur-Überdeckung.
  - Auswertung des festgelegten Ablaufszenarios.
  - Auswertung des Zeitablaufs.
  - Auswertung der eingetretenen Events, ...

# OSEK - Testansatz (zunächst technische Unterstützung)

- Parsierung auch der OSEK-Konstrukte.
- Editor zur Modellierung von Testszenarien.
- Automatische Generierung des Testtreibers zur Versorgung der Task(s) mit Events, Messages und Setzen von Zuständen (Treiber und Task laufen parallel unter Ausnutzung der normalen BS-Funktionen).
- Monitoring und Aufzeichnung des Testablaufs.
- Testauswertung und Dokumentation.