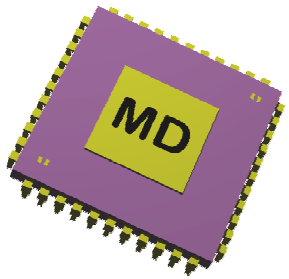
Rapid-Prototyping mit rekonfigurierbarer Hardware für eingebettete Echtzeit-Systeme mit harten Echtzeiteigenschaften

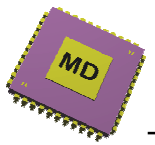
J. Hildebrandt, D. Timmermann



Universität Rostock
Fachbereich Elektrotechnik und Informationstechnik
Institut für Angewandte Mikroelektronik und Datentechnik

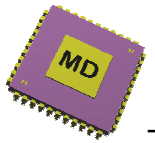
Übersicht

- Anliegen des Projektes
- Aufgabenschwerpunkte
- Gegenwärtiger Stand
- Zusammenfassung
- Ausblick

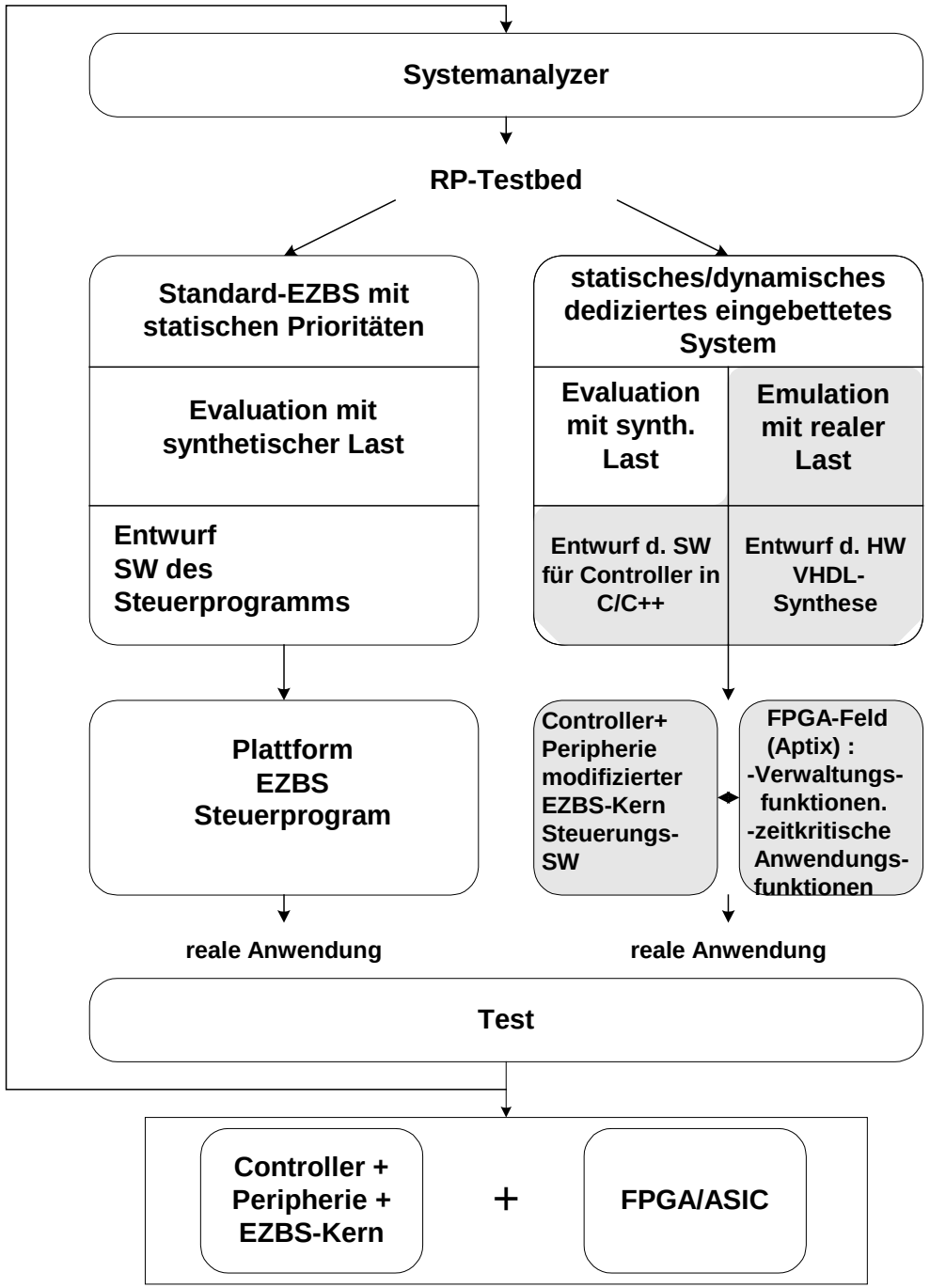


Anliegen des Projektes

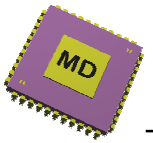
- Entwicklung einer Rapid-Prototyping-Umgebung für eingebettete EZ-Systeme
- Abbildung zeitkritischer Anwendungen auf Standard-Controller mit Peripherie und Echtzeit-Betriebssystem
- Umsetzung zeitintensiver Anwendungs- und Betriebs-system-Funktionen in rekonfigurierbarer Hardware



Rapid-Prototyping-System

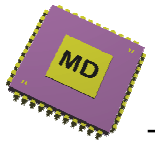


Grau unterlegte Anteile sind im Rahmen des Projektes zu entwickeln



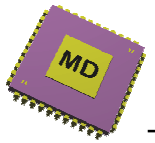
Aufgabenschwerpunkte

- Aufbau der Rapid-Prototyping - Hardwareplattform
- Anpassung eines Echtzeit-Betriebssystems an die Zielhardware
- Entwicklung einer Bibliothek von Hardware-Modulen für typische Anwendungs- und Betriebssystem-Funktionen unter Einbeziehung neuer Algorithmen
- Integration in bestehendes Analysesystem

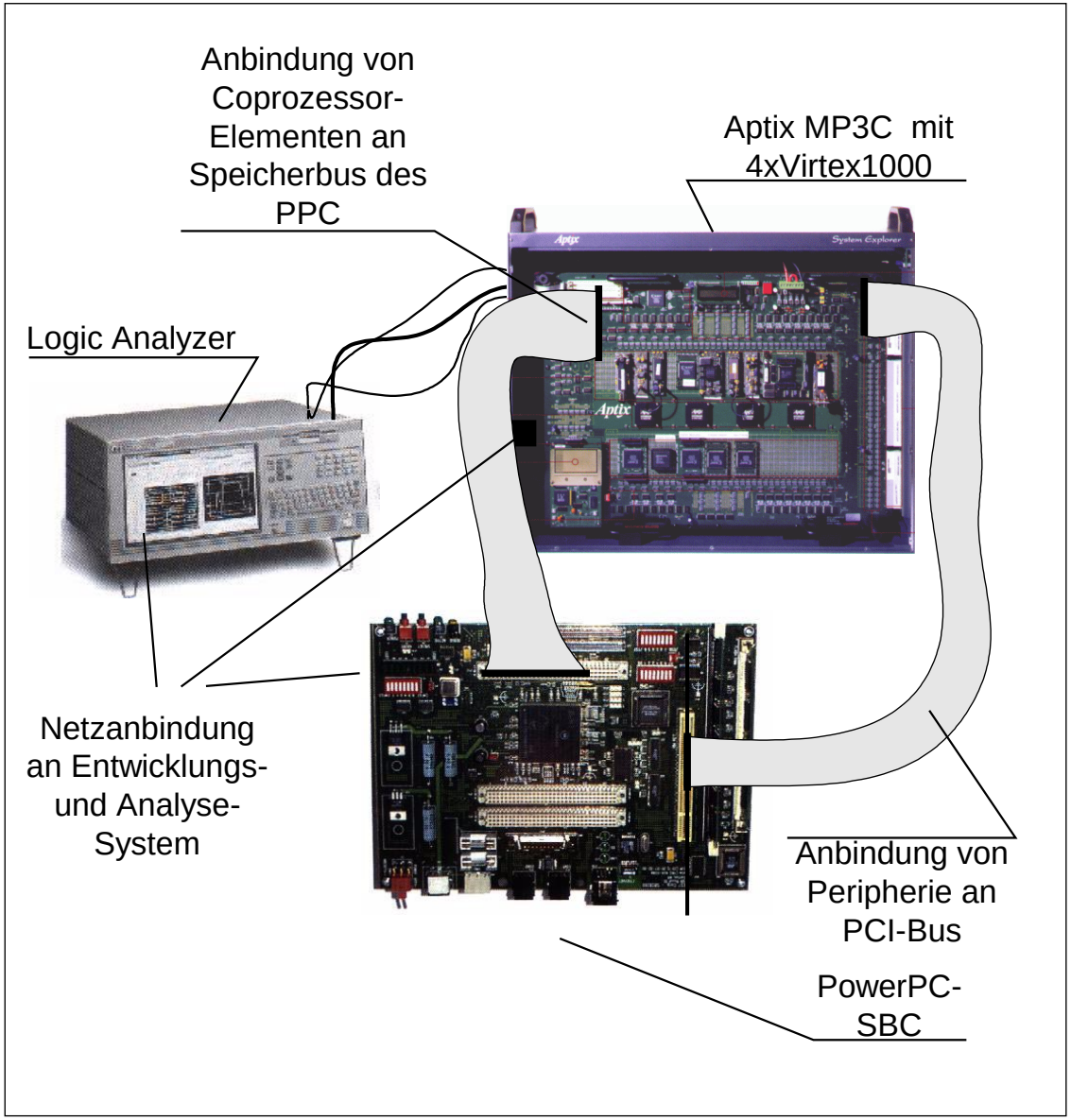


Aktueller Stand

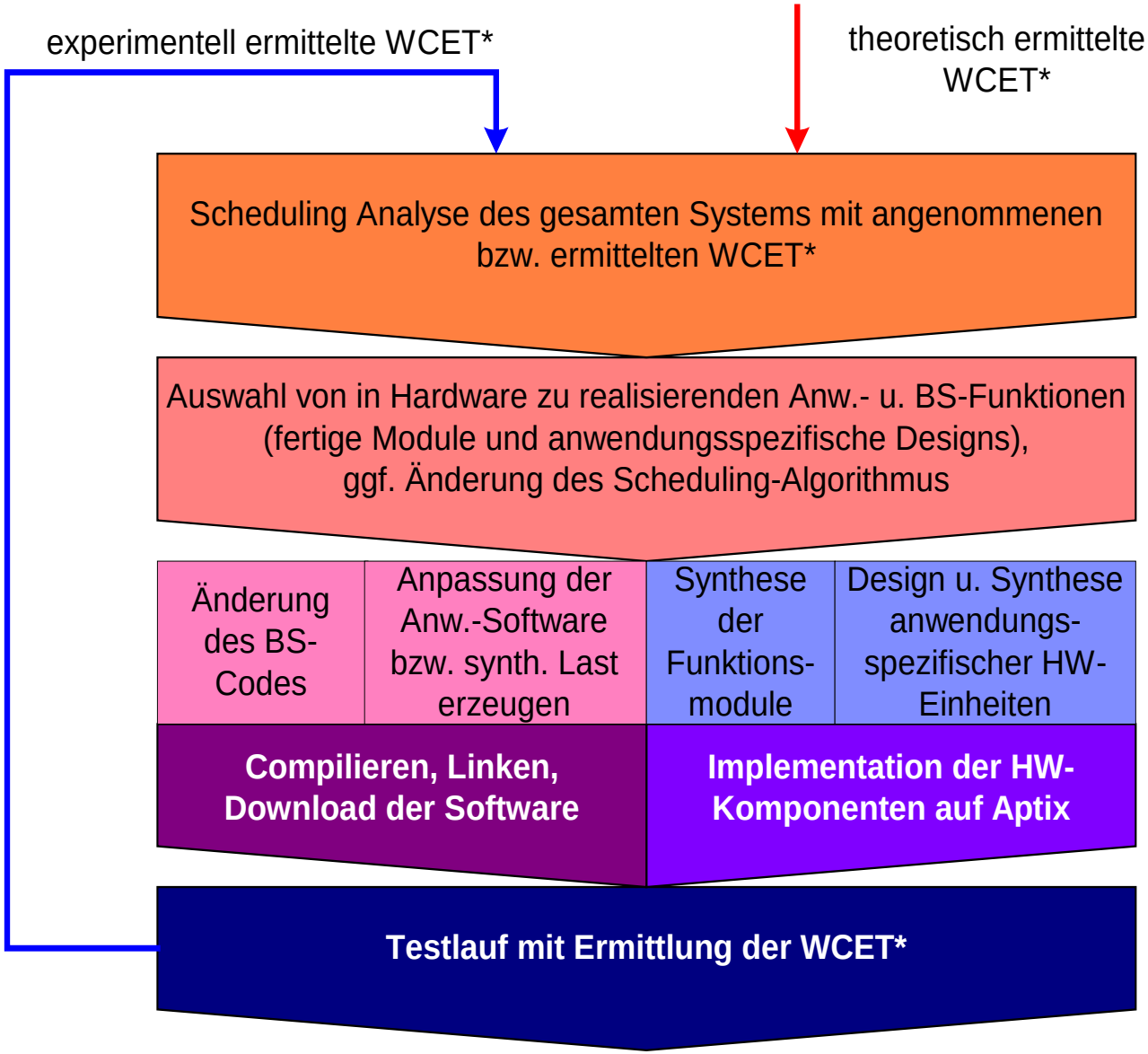
- Projektbeginn 1.6. 1998
- HW-Plattform im Aufbau
- Coprozessor-Module für dynamisches Scheduling entwickelt und implementiert
- Neuer dynamischer Scheduling-Algorithmus entwickelt
- Erweiterbarer Scheduling-Analyzer in Weiterentwicklung



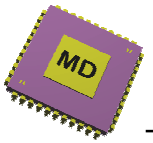
Konzept der Hardware-Plattform



Design-Zyklus

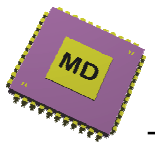


*) Worst-Case-Execution-Time der Anwendungsprozesse ***und*** Betriebssystemfunktionen

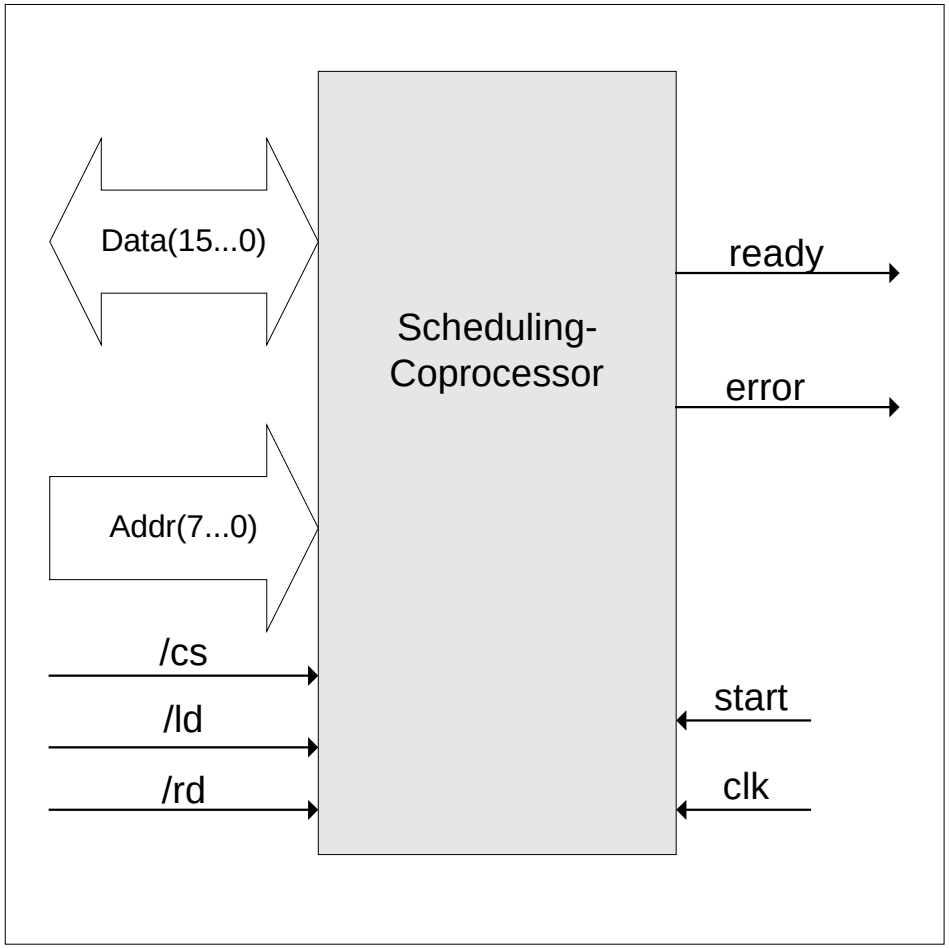


Scheduling - Coprozessoren

- Universeller Einsatz für Ein-Prozessor-Systeme
- Deterministische Arbeitsweise
- Anpassung an reale Erfordernisse über Parameter (Prozeßanzahl, Auflösung)
- Scheduler in HW- und SW-Teil aufgespalten
- Bekannte Algorithmen implementiert (RM, EDF, LLF)
- Neuer, verbesserter Scheduling-Algorithmus (ELLF) entwickelt und implementiert

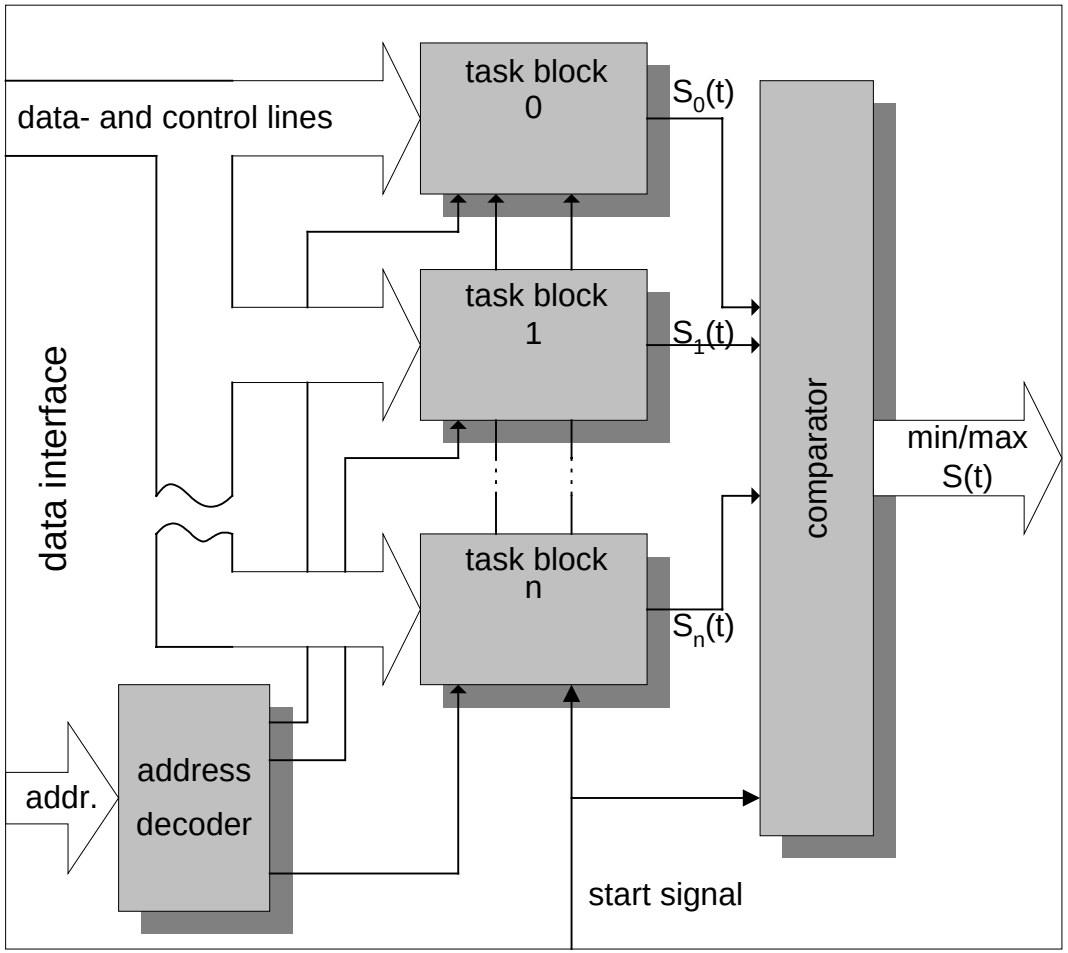


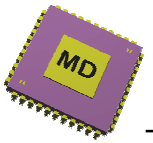
Scheduling - Coprozessor: Interface und Funktionsumfang



Operation	in...
Prozeßzustands-Verwaltung	Software
Berechnung von Prioritäten	Hardware
Prozeß-Auswahl	Hardware
Erkennung von Zeit-schranken-Verletzungen	Hardware
Prozeßverwaltung	Software
Kontextwechsel	Software
Ereignisverwaltung	Software

Grundstruktur des Scheduling-Coprozessors



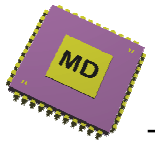


Enhanced- Least-Laxity-First-Algorithmus (ELLF)

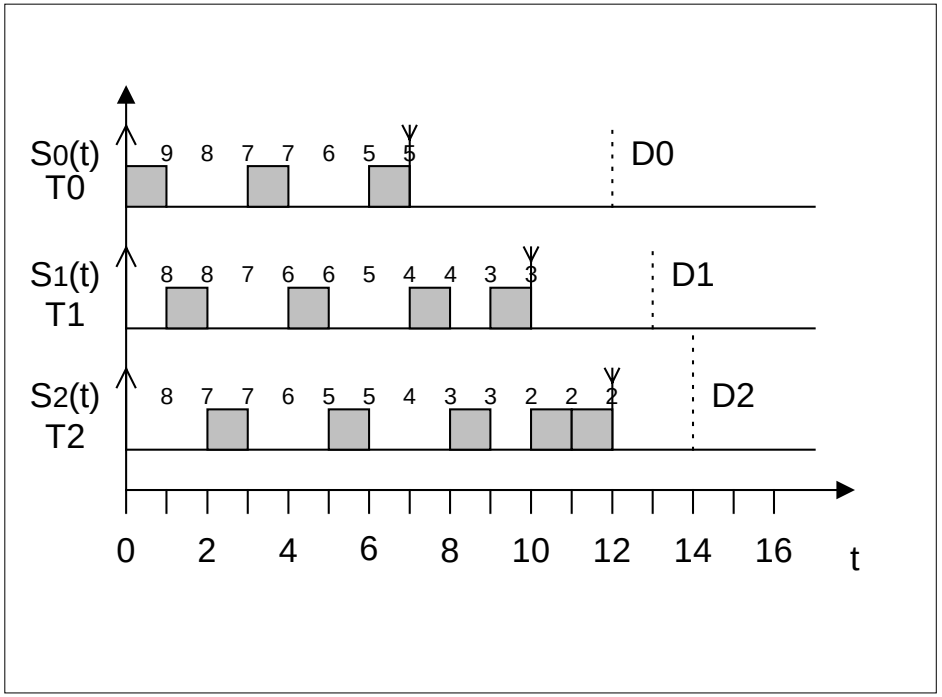
Funktionsprinzipien:

- **Derjenige Prozeß wird ausgeführt, der den kleinsten Schlupf und einen früheren Endtermin als alle anderen Prozesse mit gleich großem Schlupf hat.**

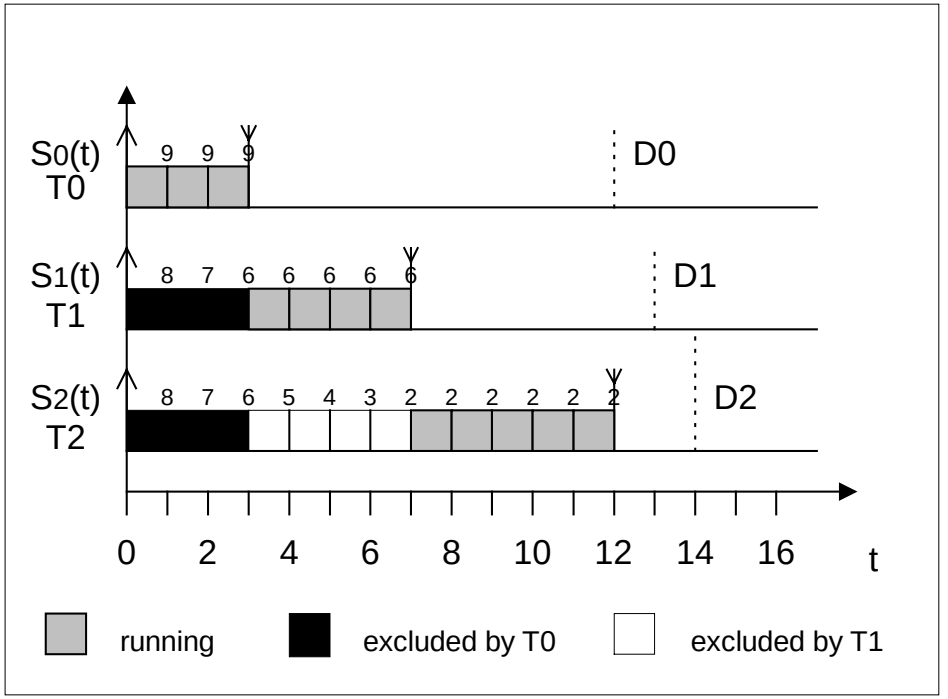
- **Kommt ein Prozeß zur Ausführung, so werden alle Prozesse, die zu diesem Zeitpunkt den gleichen Schlupf wie dieser hatten, von der Ausführung ausgeschlossen, bis der aktuell laufende Prozeß beendet oder verdrängt wird.**



Enhanced- Least-Laxity-First-Algorithmus (ELLF)

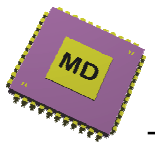


Least-Laxity-First



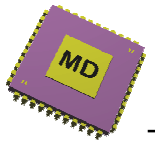
Enhanced Least-Laxity-First

$D_0 = 12$ $D_1 = 13$ $D_2 = 14$ $C_0 = 3$ $C_1 = 4$ $C_2 = 5$ $S_0(0) = S_1(0) = S_2(0) = 9$



Vorteile des ELLF-Algorithmus

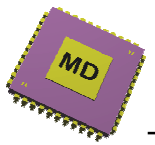
- Frühzeitiges Erkennen von Endtermin - Überschreitungen
- Höhere Auslastung als statische Verfahren
- Kein „thrashing“ wie bei LLF
- Geringere Anzahl an Prozeßwechseln als LLF
- Kürzere oder gleiche Reaktionszeiten verglichen mit LLF



Leistungsvergleich

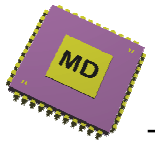
Vergleich von Hardware- und Software-Realisierung des ELLF-Algorithmus:

	Co-Prozessor	CPU mit 1 Instr./Takt
Bedingungen	Hardware-ELLF für 32 Prozesse, 16 Bit Parameter-Auflösung	Software-ELLF für 32 Prozesse, 16 Bit Parameter-Auflösung
Berechnungs-Zeit (in Takt-Zyklen)	34	350...550 abhängig von Prozeßanzahl und Zahl der Prozesse mit kleinstem Schlupf



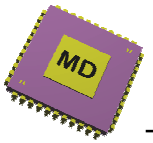
Zusammenfassung

- RP-Hardwareplattform erlaubt Umsetzung von Echtzeit-Anwendungen in kombiniertes HW-SW-System
- Neuer ELLF-Algorithmus ermöglicht dynamisches Scheduling mit weniger Kontextwechseln
- Durch Coprozessor wird Ausführungs-geschwindigkeit des Schedulers mehr als verzehnfacht
- Deterministische Arbeitsweise des Scheduling-Coprozessors erleichtert Scheduling-Analyse



Ausblick*

- Inbetriebnahme der Hardware-Plattform
- Portierung eines geeigneten Echtzeit-Betriebssystems auf die Zielhardware
- Einbindung der bereits vorhandenen HW-Module in das Betriebssystem
- Entwicklung weiterer HW-Module nach Analyse von Betriebssystem und Anwendung
- Erweiterung des Scheduling-Analyzers zur Rapid-Prototyping-Software



* bis zum Ende der 1. Projektphase

Veröffentlichungen

Golatowski, F.; Hildebrandt, J.
"Rapid-Prototyping integrierter Steuerungssysteme", 2. Wismarer Automatisierungssymposium, Wismar, 16.-17. Sept. 1999

Hildebrandt, J.; Golatowski, F.; Timmermann, D.
"Scheduling Coprocessor for Enhanced Least-Laxity-First Scheduling in Hard Real-Time Systems", 11th Euromicro Conference on Real-Time Systems, York, England, Juni 1999

Golatowski, F.; Hildebrandt, J.; Timmermann, D.
"Rapid Prototyping with Reconfigurable Hardware for Embedded Hard Real-Time Systems", 19th IEEE Real-Time Systems Symposium 98, WIP -Session, Madrid, Spanien, Dezember 1998

Golatowski, F.; Timmermann, D.
"Using Hartstone Uniprocessor Benchmark in a Real-Time Systems Course", Third IEEE Real-Time Systems Education Workshop, Poznan, Polen, Nov. 1998