

# Cyclic Scope

## Strukturelemente in Echtzeitsystemen

Florian Franzmann   Tobias Klaus   Florian Korschin  
Florian Schmaus   Peter Wägemann

Friedrich-Alexander-Universität Erlangen-Nürnberg  
Lehrstuhl Informatik 4 (Verteilte Systeme und Betriebssysteme)  
<https://www4.cs.fau.de>

16. Dezember 2015



# busy loop

```
1 void main(void) {  
2     while (true) {  
3         Task0();  
4         Task1();  
5         Task2();  
6         Task3();  
7     }  
8 }
```

## Vorteil:

- Kein Echtzeitbetriebssystem nötig
- Simpel, übersichtlich, ...

## Nachteile:

- Nur *eine Periode*, keine *Deadline-Überprüfung* möglich
- Mathematische *Analyse unmöglich*



# busy loop

```
1 void main(void) {  
2     while (true) {  
3         Task0();  
4         Task1();  
5         Task2();  
6         Task3();  
7     }  
8 }
```

## Vorteil:

- Kein Echtzeitbetriebssystem nötig
- Simpel, übersichtlich, ...

## Nachteile:

- Nur *eine Periode*, *keine Deadline-Überprüfung* möglich
- Mathematische *Analyse unmöglich*



# Multi-Perioden-Hauptschleife

Anforderung: wir wollen unterschiedliche Perioden haben

Lösung:

- Jede Aufgabe hat ein *Aktivierungs-Flag*
- In jedem Schleifendurchlauf Ausführung *höchstens* eines Tasks  
    ~> dadurch *Priorisierung*

## Multiraten-Hauptschleife

```
1 void main(void) {  
2     while (true) {  
3         if (activated0) { activated0 = false; Task0(); }  
4         else if (activated1) { activated1 = false; Task1(); }  
5         else if (activated2) { activated2 = false; Task2(); }  
6         else if (activated3) { activated3 = false; Task3(); }  
7     }  
8 }
```



## Setzen der Flags in der Hauptschleife problematisch

↪ Lang laufender Task kann Flag-Setzen/*Deadlineüberprüfung* verzögern

## Lösung: Setzen der Flags in Zeitgeber-Interruptbehandlung

```
1 // Interrupt alle 1ms
2 static unsigned int timer = 0;
3 ++timer;
4
5 if ((timer % 5) == 0) { activated0 = true; } // Task0 alle 5ms
6 if ((timer % 10) == 0) { activated1 = true; } // Task1 alle 10ms
7 if ((timer % 20) == 0) { activated2 = true; } // Task2 alle 20ms
8 if ((timer % 100) == 0) { activated3 = true; } // Task3 alle 100ms
9
10 if (timer >= 100) { timer = 0; } // Ueberlaufbehandlung
```



## Vorteile

- Kein Echtzeitbetriebssystem notwendig, simpel, übersichtlich, ...
- Mehrere Perioden, Deadlineüberprüfung möglich
- Mathematische Analyse anwendbar

## Probleme der Implementierung: *Wetlaufsituationen*

1. zwischen Zeitgeberunterbrechung und main-if/else  
    ↪ **Prioritätsverletzung** (fällt praktisch jedoch nicht ins Gewicht)
2. beim Setzen der Flags ↪ ggf. Unterbrechungen abschalten

## Andere Namen in der Literatur:

Main Loop Scheduling, Main Loop Tasker, Prioritized Cooperative Multitasker, Non-preemptive Scheduler, ...



- 1 Cyclic Executive
- 2 Rekapitulation: Cyclic Schedule
- 3 Hinweis zur Aufgabe 5



# Randbedingungen für die Rahmenlänge Lang genug und so kurz

wie möglich halten. . .

## Jobverdrängung vermeiden $\leadsto f$ hinreichend lang

1. ist erfüllt, wenn gilt:  $f \geq \max(e_i)$ , für  $1 \leq i \leq n$   
 $\leadsto$  jeder Job läuft in der durch  $f$  gegebenen Zeitspanne durch
2.  $f$  teilt die Hyperperiode  $H$  so, dass gilt:  $\lfloor p_i/f \rfloor - p_i/f = 0$   
 $\leadsto$  ermöglicht die zyklische Ausführung des Ablaufplans

- das Intervall  $H$  heißt **großer Durchlauf**  $\equiv$  Hyperperiode
- das Intervall der Länge  $f$  heißt **kleinster Durchlauf**

## Terminüberwachung unterstützen $\leadsto f$ hinreichend kurz

3. erfordert eine rechtzeitige Auslösung:  $f \leq p_i$ , für  $1 \leq i \leq n$
4. ist möglich unter der Bedingung:  $2f - \text{ggT}(p_i, f) \leq D_i$

- anstehenden Aufgaben „passend“ auf die Rahmen verteilen  
 $\leadsto$  Jobs zwischen Auslösezeit und Termin erledigen



# Randbedingungen für die Rahmenlänge Lang genug und so kurz

wie möglich halten. . .

## Jobverdrängung vermeiden $\leadsto f$ hinreichend lang

1. ist erfüllt, wenn gilt:  $f \geq \max(e_i)$ , für  $1 \leq i \leq n$   
 $\leadsto$  jeder Job läuft in der durch  $f$  gegebenen Zeitspanne durch
2.  $f$  teilt die Hyperperiode  $H$  so, dass gilt:  $\lfloor p_i/f \rfloor - p_i/f = 0$   
 $\leadsto$  ermöglicht die zyklische Ausführung des Ablaufplans

- das Intervall  $H$  heißt **großer Durchlauf**  $\equiv$  Hyperperiode
- das Intervall der Länge  $f$  heißt **kleinster Durchlauf**

## Terminüberwachung unterstützen $\leadsto f$ hinreichend kurz

3. erfordert eine rechtzeitige Auslösung:  $f \leq p_i$ , für  $1 \leq i \leq n$
4. ist möglich unter der Bedingung:  $2f - \text{ggT}(p_i, f) \leq D_i$

- anstehenden Aufgaben „passend“ auf die Rahmen verteilen  
 $\leadsto$  Jobs zwischen Auslösezeit und Termin erledigen



wie möglich halten. . .

## Jobverdrängung vermeiden $\leadsto f$ hinreichend lang

1. ist erfüllt, wenn gilt:  $f \geq \max(e_i)$ , für  $1 \leq i \leq n$   
 $\leadsto$  jeder Job läuft in der durch  $f$  gegebenen Zeitspanne durch
2.  $f$  teilt die Hyperperiode  $H$  so, dass gilt:  $\lfloor p_i/f \rfloor - p_i/f = 0$   
 $\leadsto$  ermöglicht die zyklische Ausführung des Ablaufplans

- das Intervall  $H$  heißt **großer Durchlauf**  $\equiv$  Hyperperiode
- das Intervall der Länge  $f$  heißt **kleinster Durchlauf**

## Terminüberwachung unterstützen $\leadsto f$ hinreichend kurz

3. erfordert eine rechtzeitige Auslösung:  $f \leq p_i$ , für  $1 \leq i \leq n$
4. ist möglich unter der Bedingung:  $2f - \text{ggT}(p_i, f) \leq D_i$

- anstehenden Aufgaben „passend“ auf die Rahmen verteilen  
 $\leadsto$  Jobs zwischen Auslösezeit und Termin erledigen



Aufgabe $T_i$	<u>Periode <math>p_i</math></u> ms	<u>WCET <math>e_i</math></u> ms	<u>Termin <math>D_i</math></u> ms
$T_1$	9	2	5
$T_2$	18	3	8
$T_3$	45	3	45



1 Cyclic Executive

2 Rekapitulation: Cyclic Schedule

**3 Hinweis zur Aufgabe 5**



# Aufgabe 5 - Hinweise

## Wichtige Hinweise

Basisübung: Reine Textaufgabe, *Denksportaufgabe*

~> keine Implementierung notwendig

- Kern der Aufgabe: Auswirkung der Rahmenlänge

Erweiterte Übung: Implementierung einer *Cyclic Executive*

Bearbeitung als kleine **Hausaufgabe über die Weihnachtsferien**

**Fester Termin für die Abgabe:** Übungen in der Woche vom 11.01.2016



# Fragen?



<sup>0</sup><https://commons.wikimedia.org/wiki/User:Pensiero~commonswiki>

