

Antwortzeit

Antwortzeitmessung und Zeitverbrauch

Florian Franzmann Tobias Klaus Peter Wägemann

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

29. Oktober 2015



1 Zeitmessung

- Wiederholung: Zeitgeber
- Antwortzeit
- Ausführungszeit

2 libEVS



Zähler (Counter) zählen hardwarebasiert Ereignisse z. B. von:

- Externem Drehgeber (Radumdrehung)
- Externem Quarz (Real-Time Clock)
- Internem Prozessortakt (hohe Auflösung)

Äquidistante Ereignisse ermöglichen einen *Zeitgeber (Timer)* für

- Periodische Aktivierung
- *Messen von Zeitabständen*
- *Kontrolliertes Verbrennen von Prozessorzeit*



Zähler bzw. Zeitgeber bieten zwei Betriebsmodi:

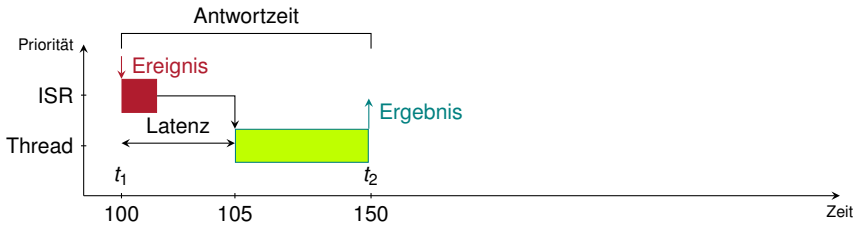
Abfragebetrieb (Polling) Aktives Auslesen des Zählers

~> bis Erreichen eines vorgegebenen Wertes

Unterbrecherbetrieb (Interrupt) Zähler unterbricht System

~> Erreichen eines konfigurierten Zählerstandes.

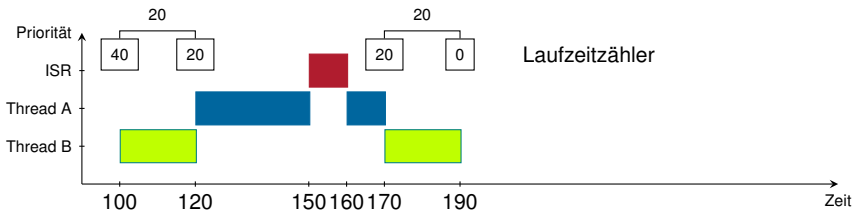




Stoppuhr

- Punkte auf der Zeitachse t_1 und $t_2 \rightsquigarrow$ Ereignis und Ergebnis
- Antwortzeit ist $\Delta t = t_2 - t_1$ (Beispiel: $150 - 100 = 50$ Zählerticks)





Rechenzeitsimulation

- Verbrauchte *Laufzeit* eines Threads
- Vorgegebene Zeit aktiv warten \rightsquigarrow Laufzeit verbrauchen

Umsetzung

- Funktion, die *aktiv* t_{wcet} wartet \rightsquigarrow Schleife auf Zählerwert
- HW-Zähler läuft bei Unterbrechungen weiter! \rightsquigarrow *lokaler* Zähler
- Dekrement bei jeder Änderung? Beispiel: Sprung von 120 \rightarrow 170



- 1 Zeitmessung
 - Wiederholung: Zeitgeber
 - Antwortzeit
 - Ausführungszeit

- 2 libEVS



Plattformunabhängige Hilfsfunktionen

- Timer-Zugriff (Zeitmessung)
- DAC-Zugriff
- GPIO-Zugriff
- ...

```
aufgabe2
|-- CMakeLists.txt
|-- app.c
|-- ecos
'-- libEVS
    |-- include
    | |-- ezs_dac.h
    | |-- ezs_gpio.h
    | '-- ezs_stopwatch.h
    |-- src
    | '-- ezs_stopwatch.c
    '-- drivers
        '- tc1796
            |-- ezs_dac.c
            |-- ezs_counter.c
            '-- ezs_gpio.c
```

Die *libEVS* wird im Laufe der Übungen von euch erweitert.



Zeitmessung in `ezs_stopwatch.c/.h`

Die Zeitmessung wird durch zwei Funktionen implementiert:

```
void ezs_watch_start(cyg_uint32 *state);  
cyg_uint32 ezs_watch_stop(cyg_uint32 *state);
```

- Parameter: Zeiger auf *globale* Variable
→ viele unabhängige Messzeitpunkte
- `ezs_watch_stop(cyg_uint32 *state)` gibt Zeitdifferenz in *Ticks* zurück

Hinweis

`ezs_counter_get()` in `drivers/include/ezs_counter.h`

Hinweis

Auflösung der Zähler in Pikosekunden:

→ `ezs_counter_resolution_ps()`



Zu implementieren:

```
void ezs_lose_time(cyg_uint32 wcet, cyg_uint8 percentage);
```

■ Parameter:

- 1.: Gewünschte WCET in *Ticks*
- 2.: *Maximum* des zufällig zu subtrahierenden *WCET-Anteils*

■ Implementierung muss internen Zähler verwalten

↪ Bei welcher Änderung des Systemzählers anpassen?

↪ Welche Auflösung ist erreichbar

- Jeder Thread besitzt einen eigenen Stack! ↪ *keine* globale Zustandsvariable notwendig/sinnvoll

■ *Abfragebetrieb*

Hinweis

Auflösung des Zählers in Pikosekunden:

→ `ezs_counter_resolution_ps()`



Besprechung der Übungsaufgabe „Antwortzeit“



Fragen?



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

