



# Performance Messungen von FreeRTOS und $\mu$ C/OS-III auf ARM-Architekturen

Tim Wachter (wht4@bfh.ch)

Master of Science in Engineering  
MRU Production Technology

16. August 2011/ CH-3400 Burgdorf

# Outline



## 1 Ziel dieser Präsentation

## 2 Grundlagen

## 3 Entscheidungskriterien

- Allgemeine Metrics
- Messbare Metrics
- Messsystem

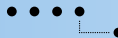
## 4 Messungen

- Allgemeines
- Beispiel Interrupt Response

## 5 Vergleich

- Gesamtergebnis
- Allgemeine Metrics
- Messbare Metrics

# Ziel dieser Präsentation



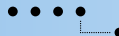
## Ziel?

Gegenüberstellen der Echtzeitbetriebssysteme freeRTOS und  $\mu\text{C}/\text{OS-III}$  auf den ARM-Architekturen XScale PXA-270 und LM3S9B92 Cortex-M3.

## Lösung

- Finden von geeigneten Entscheidungskriterien (Metrics), damit ein objektiver Vergleich möglich ist.
- Bewerten und gewichten der einzelnen Entscheidungskriterien.

# Ziel dieser Präsentation



## Ziel?

Gegenüberstellen der Echtzeitbetriebssysteme freeRTOS und  $\mu$ C/OS-III auf den ARM-Architekturen XScale PXA-270 und LM3S9B92 Cortex-M3.

## Lösung

- Finden von geeigneten Entscheidungskriterien (Metrics), damit ein objektiver Vergleich möglich ist.
- Bewerten und gewichten der einzelnen Entscheidungskriterien.

## Eigenschaften freeRTOS

- RTOS unter einer modifizierten GPL-Lizenz erhältlich
- Optionale kostenpflichtige Versionen (openRTOS und safeRTOS)
- Preemptives Multitasking mit Round Robin Scheduling
- IPC-Mechanismen:
  - Binäre Semaphoren
  - Counting Semaphoren
  - Mutex Semaphoren
  - Message Queues
- 27 unterstützte Architekturen [1]

## Eigenschaften $\mu$ C/OS-III

- Kostenpflichtiges RTOS von Micrium
- Source beim Kauf einer Lizenz verfügbar; Royalty Free
- Preemptives Multitasking mit Round Robin Scheduling
- IPC-Mechanismen:
  - Binäre Semaphoren
  - Counting Semaphoren
  - Mutex Semaphoren
  - Message Queues
  - Event Flags
  - Memory Management
- 45 unterstützte Architekturen [2]

# Allgemeine Metrics

Metric	Gewichtung
<b>Kernel</b>	<b>8.0%</b>
Multicore Unterstützung	0.8%
Zusätzliche Module	2.4%
Skalierbarkeit	2.4%
Unterstützte Standards	2.4%
<b>Scheduler</b>	<b>4.0%</b>
Scheduling Algorithmus	2.8%
Anzahl unterstützte Prioritäten	1.2%
<b>Task Modell</b>	<b>2.0%</b>
Maximale Anzahl Tasks	1.2%
Dynamisches ändern der Prioritäten	0.8%
<b>Kommerzielle Überlegungen</b>	<b>10.0%</b>
Kosten	2%
Dokumentation / Support	
Dokumentation	2.0%
Internetauftritt	2.0%
Forum	2.0%
Kontaktperson	2.0%

Metric	Gewichtung
<b>API-Umfang</b>	<b>15.0%</b>
IPC-Mechanismen	
Binäre Semaphoren	1.0%
Counting Semaphoren	1.0%
Mutex Semaphoren	1.0%
Message Queues	1.0%
Mailboxes	1.0%
Event Flags	1.0%
Pipes	1.0%
Condition Variables	1.0%
Signals	1.0%
Hook Funktionen	1.5%
Stack-Überwachung	3%
Statistik Task	1.5%
<b>Entwicklungsüberlegungen</b>	<b>6.0%</b>
RTOS als Source vorhanden	2.4%
Unterstützte Architekturen	3.6%
<b>Allgemeine Metrics Gewichtung</b>	<b>45.0%</b>

# Messbare Metrics



- Eigenschaften welche der Kernel abhängig von der Architektur und den gewählten Entwicklungstools aufweist.
- Es interessieren die maximalen Zeiten.

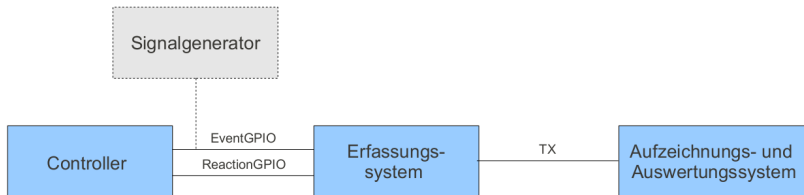
Metric	Gewichtung
<b>Interrupt Response</b>	<b>13.0%</b>
<b>Interrupt Response Task Level</b>	<b>12.0%</b>
<b>System Call Latenzzeit</b>	<b>5.0%</b>
<b>Context Switch Overhead</b>	<b>12.0%</b>
<b>Jitter eines periodischen Tasks</b>	<b>8.0%</b>
<b>Ressourcen Verbrauch</b>	<b>5.0%</b>
Datentypen	2.5%
Programmspeicher	2.5%
<b>Messbare Metrics Gewichtung</b>	<b>55.0%</b>

# Messsystem

## Anforderungen Messsystem

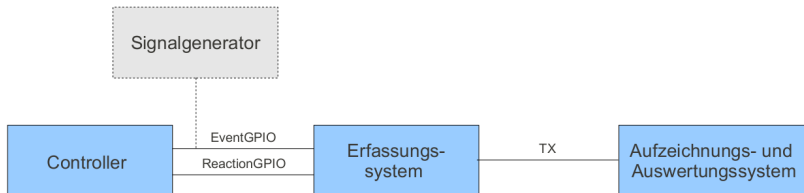
Das Messsystem muss folgende Kriterien erfüllen:

- Genügende Auflösung und Genauigkeit
- Messungen über längere Zeit aufzeichnen
- Alle Messdaten aufzeichnen (Keine Komprimierung oder verpassten Messdaten)



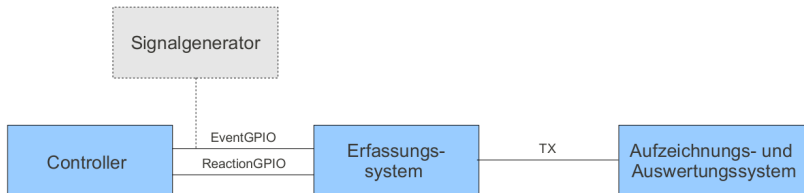
# Beispiel Interrupt Response

- 1 Auslösen eines Interrupts mit dem Signalgenerator → Erfassungssystem startet Counter
- 2 Controller wechselt vom Task- in den Interrupt-Level
- 3 Interrupt Service Routine (ISR) wird ausgeführt
- 4 Controller setzt den ReactionGPIO → Erfassungssystem stoppt Counter
- 5 Counter-Wert mittels serieller Schnittstelle an Aufzeichnungssystem senden



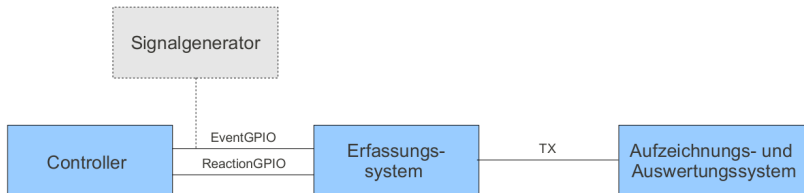
# Beispiel Interrupt Response

- 1 Auslösen eines Interrupts mit dem Signalgenerator → Erfassungssystem startet Counter
- 2 Controller wechselt vom Task- in den Interrupt-Level
- 3 Interrupt Service Routine (ISR) wird ausgeführt
- 4 Controller setzt den ReactionGPIO → Erfassungssystem stoppt Counter
- 5 Counter-Wert mittels serieller Schnittstelle an Aufzeichnungssystem senden



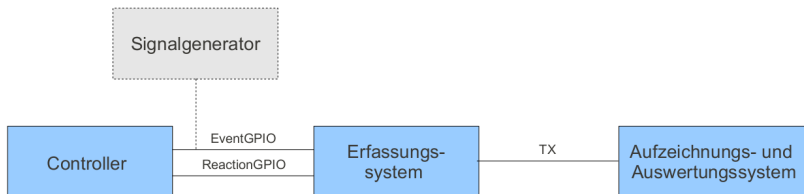
# Beispiel Interrupt Response

- 1 Auslösen eines Interrupts mit dem Signalgenerator → Erfassungssystem startet Counter
- 2 Controller wechselt vom Task- in den Interrupt-Level
- 3 Interrupt Service Routine (ISR) wird ausgeführt
- 4 Controller setzt den ReactionGPIO → Erfassungssystem stoppt Counter
- 5 Counter-Wert mittels serieller Schnittstelle an Aufzeichnungssystem senden



# Beispiel Interrupt Response

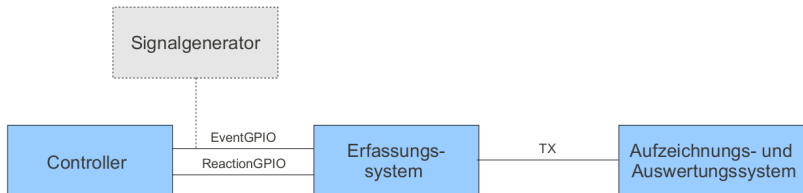
- 1 Auslösen eines Interrupts mit dem Signalgenerator → Erfassungssystem startet Counter
- 2 Controller wechselt vom Task- in den Interrupt-Level
- 3 Interrupt Service Routine (ISR) wird ausgeführt
- 4 Controller setzt den ReactionGPIO → Erfassungssystem stoppt Counter
- 5 Counter-Wert mittels serieller Schnittstelle an Aufzeichnungssystem senden



# Beispiel Interrupt Response



- 1 Auslösen eines Interrupts mit dem Signalgenerator → Erfassungssystem startet Counter
- 2 Controller wechselt vom Task- in den Interrupt-Level
- 3 Interrupt Service Routine (ISR) wird ausgeführt
- 4 Controller setzt den ReactionGPIO → Erfassungssystem stoppt Counter
- 5 Counter-Wert mittels serieller Schnittstelle an Aufzeichnungssystem senden



## Messungen

- Die messbaren Metrics werden stark durch die Einstellung des Systems beeinflusst.
  - Konfiguration und Version des eingesetzten RTOS
  - Belastung des Systems durch Interrupts und oder andere Tasks
  - Zuteilung der Task- und Interrupt-Prioritäten
  - Entwicklungstools und deren Einstellungen
  - ...
- Metrics müssen bei verschiedenen untereinander vergleichbaren Einstellungen ausgemessen werden.

## Messungen

- Die messbaren Metrics werden stark durch die Einstellung des Systems beeinflusst.
  - Konfiguration und Version des eingesetzten RTOS
  - Belastung des Systems durch Interrupts und oder andere Tasks
  - Zuteilung der Task- und Interrupt-Prioritäten
  - Entwicklungstools und deren Einstellungen
  - ...
- Metrics müssen bei verschiedenen untereinander vergleichbaren Einstellungen ausgemessen werden.

# Beispiel Interrupt Response I



## LM3S9B92 Cortex-M3

<b>OS Konfiguration:</b>	Standard (OS-Tick, Debug Variablen, Hook Funktionen, Stack Overflow Test, ...)
<b>Beschreibung:</b>	ExtInterrupt @ PrioLevel5 500Hz OSTick @ PrioLevel7 100Hz Ein LED-Task @ LowPrio 10Hz
<b>Optimization:</b>	-O2

# Beispiel Interrupt Response II



**Diese Folie wurde aus Lizenzgründen weggelassen**

# Gesamtresultat



**Diese Folie wurde aus Lizenzgründen weggelassen**

# Allgemeine Metrics



## freeRTOS

- + Unter modifizierten GPL-Lizenz erhältlich
- + Internetauftritt und Forum

## $\mu$ C/OS-III

- + Grosser API-Umfang
- + Zusätzliche Module (GUI, FileSystem, USB,...)
- + Unterstützung von Standards in Bearbeitung (DOB178B, IEC61508,...)



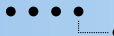
# Interrupt Response



**Diese Folie wurde aus Lizenzgründen weggelassen**



# Context Switch Overhead



**Diese Folie wurde aus Lizenzgründen weggelassen**

# Schlusswort



## Bewertung des Gesamtergebnisses

Die Resultate zeigen, dass es keine relevanten Unterschiede zwischen  $\mu\text{C}/\text{OS-III}$  und freeRTOS gibt.

### Achtung!

Diese Aussage gilt für die gewählte Gewichtung der Kriterien!

# Schlusswort



## Bewertung des Gesamtergebnisses

Die Resultate zeigen, dass es keine relevanten Unterschiede zwischen  $\mu\text{C}/\text{OS-III}$  und freeRTOS gibt.

### **Achtung!**

Diese Aussage gilt für die gewählte Gewichtung der Kriterien!



# Fragen



# Fragen???

# Quellenverzeichnis



- [1] Homepage freeRTOS  
*[www.freertos.org/](http://www.freertos.org/)*  
Stand August 2011
- [2] Micrium Homepage  
*<http://micrium.com>*  
Stand August 2011
- [3] Wachser Tim  
*Performance Measurement of RTOS*  
BFH-TI; MSc Engineering Projektarbeit 2  
Burgdorf; 26. Januar 2011