AUFGABE 5: CYCLIC SCOPE

Hinweis: Die *Basisaufgabe* ist eine reine Textaufgabe, eine konkrete Implementierung ist nicht erforderlich.

1 Aufgabenstellung

Nach dem erfolgreichen Aufsetzen des `build`*-Verzeichnisses können Sie sich mittels* `make doc` *eine Übersicht über alle in* `libEZS` *bereitgestellten Funktionen inklusive deren Dokumentation erzeugen.*

Denken Sie daran, Ihren Quellcode nach vollständiger Bearbeitung noch vor dem Beginn der Rechnerübung abzugeben. Rufen Sie hierzu in Ihrem `build`*-Verzeichnis* `make submit` *auf.* In dieser Übungsaufgabe sollen die unter der Bezeichnung „Zyklische Ablaufpläne“ vorgestellten Konzepte auf unser einfaches Oszilloskop angewendet werden. Grundlage bildet das aus Aufgabe 4 bekannte System von periodischen Aufgaben, nun allerdings mit größerer Last für die Darstellung:

Aufgabe	Bezeichnung	Periode ms	WCET ms	Prioritt ms
T_1	Abtastung Signal 1	10	2	_
T_2	Abtastung Signal 2	20	2	_
T_3	Analyse	20	4	_
T_4	Darstellung	40	12	_

Die Aufgaben verfügen über implizite Termine zu Beginn ihrer nächsten Periode. Behalten Sie in den folgenden Teilaufgaben die Ziele der zyklischen Ablaufplanung im Hinterkopf. Bearbeiten Sie die Problemstellungen konstruktiv, halten Sie Ihr Vorgehen für die Abgabe geeignet fest (grafisch, textuell, ...).

1.1 Planbarkeitsanalyse:

Zunächst ist die grundsätzliche Frage der Planbarkeit des periodischen Aufgabensystems zu klären. Vergeben Sie hierfür statische Prioritäten für die Aufgaben gemäß des aus der Vorlesung bekannten *Ratenmonotonen Algorithmus* (RMA).

Aufgabe 1

Aus der Vorlesung kennen Sie die Planbarkeitsanalyse nach CPU-Auslastung. Ist das oben vorgegebene Aufgabensystem mittels RMA planbar? Führen Sie dazu die Rechenzeitbedarfsanalyse durch.

Antwort:

1.2 Strukturierter Ablaufplan:

Erstellen Sie unter Verwendung der in der Vorlesung vorgestellten Strukturelemente einen *zyklischen Ablaufplan*.

Aufgabe 2

Berechnen Sie unter Berücksichtigung der vier Bedingungen die Hyperperiode und eine gültige Rahmenlänge. Passen Sie das Aufgabensystem systematisch an, sofern Sie im ersten Anlauf keine gültige Lösung finden. Halten Sie hierbei alle Lösungswege schriftlich fest.

Antwort:

Aufgabe 3

Welche fundamentale Eigenschaft herrscht (bzw. fehlt) innerhalb eines Rahmens? Welche Schwierigkeit ergibt sich hieraus für den Anwendungsentwickler?

Antwort:

Aufgabe 4

Stellen Sie den vollständigen zyklischen Ablaufplan (z. B. mit Papier und Stift) auf.

Aufgabe 5

Was sind die wesentlichen Vor- und Nachteile des zyklischen Ablaufmodells?

Antwort:

1.3 Nicht-periodische Aufgaben:

Im zweiten Teil der Aufgabe soll nun eine *aperiodische* Aufgabe in den Ablaufplan integriert werden:

exTask	Bezeichnung	Min. Zwischenankunftszeit / ms	WCET / ms
T_5	Signal-Trigger	5	1

In der Vorlesung haben Sie verschiedene Möglichkeiten kennengelernt, wie man nicht-periodische Ereignisse in Echtzeitsystemen umsetzten kann: Unterbrecherbetrieb, Hintergrundbetrieb und periodische Zusteller.

Aufgabe 6

Identifizieren Sie die kritischen Punkte in ihrem Ablaufplan (z. B. volle Rahmen, nahe Deadline, ...) und fügen Sie die Aufgabe T_5 an diesen Stellen exemplarisch in den Ablaufplan ein. Variieren Sie die Art der Behandlung und schätzen Sie die Auswirkungen auf die Antwortzeit von T_5 **sowie** das Verhalten der periodischen Aufgaben $T_1 - T_4$ ab.

Antwort:

Aufgabe 7

Wie lassen sich aperiodische Aufgaben also sinnvoll in Ihren zyklischen Ablaufplan integrieren? Begründen Sie ihre Entscheidung! Wie steht es nun um die Vor- und Nachteile des zyklischen Ablaufmodells?

Antwort:

1.4 Slack-Stealing:

Sie haben in der Vorlesung eine weitere Möglichkeit zur Behandlung von aperiodische Aufgaben kennengelernt, das Slack-Stealing.

Aufgabe 8

Planen Sie die Aufgabe T_5 unter Ausnutzung des zur Verfügung stehenden Schlupfs ein (z.B. Papier und Stift)! Wie können Sie den Ablaufplan in Bezug auf die periodischen Aufgaben modifizieren um möglichst viel Schlupf zu erzeugen?

Aufgabe 9

Wieso eignen sich Ablaufpläne besonders gut für Slack-Stealing? Wie steht es um die Vorteile des zyklische Ausführungsmodell bei Nutzung von Slack-Stealing?

Antwort:

Aufgabe 10

Wie würden Sie das zyklische Ablaufmodell in eCos (oder einem anderen RTOS) implementieren? Wieso ist dies gerade bei kleinen, ressourcenbeschränkten Mikrocontrollern eine häufige Option? *Hinweis: Eine Implementierung ist in dieser Aufgabe nicht erforderlich.*

Antwort:

2 Erweiterte Aufgabe

Die Erweiterten Übungsaufgaben sind nur für Teilnehmer verpflichtend, die das 7,5-ECTS-Modul belegen. Wir werden Sie natürlich auch dann bei der Bearbeitung unterstützen, wenn Sie diese Teilaufgaben freiwillig bearbeiten.

2.1 Implementierung der Cyclic Executive:

Aufgabe 11

In der erweiterten Aufgabe sollen Sie eine *Cyclic Executive* für das Tasksystem aus Teilaufgabe 1 der Basisübung mit Hilfe von eCos erstellen. Verwenden Sie **einen** Faden um die Hauptschleife umzusetzen und **einen** eCos-Alarm zum Setzen der Flags. Die WCET der Arbeitsaufträge können Sie wieder mit Hilfe der Funktion `ezs_simulate_wcet()` umsetzen. *Hinweis: Bitte beachten Sie, dass eventuelle* `printf`*-Aufrufe im Funktionsrumpf der Aufgaben das Zeitverhalten negativ beeinflussen. Deaktivieren sie diese unbedingt für Funktionstests mit Rechenzeiten nahe der eingeplanten WCET, insbesondere für die kommende Teilaufgabe „Deadlineüberprüfung“.*

Aufgabe 12

Wie würden Sie die nichtperiodische Aufgabe T_5 in Ihrer Cyclic Executive bei Unterbrecherbetrieb, Hintergrundbetrieb bzw. Slack Stealing jeweils umsetzen?

Antwort:

2.2 Deadlineüberprüfung:

Aufgabe 13

Implementieren Sie nun eine Deadlineüberprüfung für Ihre Cyclic Executive. Als *Ausnahmebehandlung* ist es für diese Aufgabe hinreichend, wenn Sie mit Hilfe von `ezs_printf()` eine Fehlermeldung ausgeben, die mitteilt, welche Aufgabe ihre Deadline verletzt hat.

Aufgabe 14

Testen Sie die korrekte Funktion der Deadlineüberprüfung! ☞ `ezs_simulate_wcet()`

Hinweise

- Bearbeitung: Gruppe mit je drei Teilnehmern.
- Abgabefrist: 24.06.2024 (23:59) ☞ `make submit`
- Fragen bitte an `i4ezs@lists.cs.fau.de`