AUFGABE 3: AUSFÜHRUNGSZEIT

In dieser Übungsaufgaben werden Sie sich mit verschiedenen Möglichkeiten befassen, maximale Ausführungszeiten abzuschätzen. Ziel dieser Aufgabe ist es, ein Gefühl für die Grenzen dieser Herangehensweisen zu bekommen.

Aufgabenstellung

1. *Zeitmessung mit der libEZS:*
Wir haben für Sie bereits die Funktionen `heapsort()` und `quicksort()` implementiert. *Von welchen Faktoren ist die Ausführungszeit dieser Funktionen abhängig?* Verwenden Sie die von Ihnen implementierte Stoppuhr um die *Worst Case Execution Time (WCET)* beider Funktionen abzuschätzen. Visualisieren Sie sich hierbei auch das Histogramm der gemessenen Zeitwerte. Die Zeitwerte können Sie hierzu auf die serielle Konsole ausgeben. Wenn Sie sich diese beispielsweise mit Hilfe von `cutecom` anschauen, können Sie die Messwerte direkt in eine Datei speichern. Diese Daten können dann z. B. in `qtiplot` plotten lassen (rechte Maustaste auf den Kopf der Spalte mit den Werten, Menüpunkt „Plot“, „Statistical Graphs“, „Histogram“). Implementieren Sie eine automatische Messung von mindestens 100 Stichproben. *Weshalb ist dies sinnvoll? Wie hoch ist die Auflösung Ihrer Messung? In welcher Größenordnung liegt Ihre Messgröße? Ist Ihre Messung damit gültig? Falls nicht, wie können Sie die Messung zumindest statistisch durchführen?*

☞ `print_measurement()`

Versuchen Sie eine möglichst lange Laufzeit der Funktion `heapsort()` zu erzwingen: Die Gruppe, die die längste Laufzeit für ein Eingabearray der Länge 1024 demonstrieren kann, erhält eine Belohnung.
Wie groß ist der Mittelwert und dessen Standardfehler bei dieser Eingabe? Wie groß ist die Spanne zwischen gemessenem Minimal- und Maximalwert? Welche Bedeutung haben diese Wert im Bezug auf die WCET? Wie genau ist Ihre Messung?

2. *Zeitmessung mit dem Oszilloskop:* In dieser Teilaufgabe lernen Sie Ausführungszeiten mit Hilfe des Oszilloskops zu ermitteln. Die Funktion `ezs_gpio_set()` erlaubt es Ihnen den auf der Experimentierplatine herausgeführten Pin des Tricore, der bisher zur Erzeugung des PWM-Signals diente, als GPIO zu verwenden. *Wie können Sie mit Hilfe des GPIO-Pins Ausführungszeiten messen?* Führen Sie nun die Messungen der vorangegangenen Teilaufgabe noch einmal mit Hilfe von GPIO-Pin und Oszilloskop durch. *Wie genau stimmen diese mit den Messungen Ihrer Stoppuhr überein? Wieviel Stichproben schaffen Sie pro Minute?*
Welche der beiden Messarten ist zu bevorzugen?
Welche Probleme sehen Sie sowohl bei der WCET-Abschätzung durch den Zeitgeber als auch bei der durch das Oszilloskop?

3. *Werkzeuggestützte WCET-Ermittlung:* In dieser Teilaufgabe sollen Sie die WCET mit Hilfe eines Werkzeugs zur Codeanalyse bestimmen. Im Rahmen dieser Veranstaltung kommt hierbei der *aiT* der Firma *AbsInt* zum Einsatz. Laden Sie Ihr Programm, wie in den Übungsfolien dargestellt, in den aiT und analysieren Sie die WCET der Funktionen `heapsort()` und `heapsort_job()`. *Worin liegt aus Sicht von aiT der Unterschied zwischen diesen Funktionen? Welche ist besser geeignet um eine realistische WCET-Analyse für die vorliegende Anwendung zu erhalten?* Während der Analyse treten Warnungen auf. *Weshalb dürfen Sie diese nicht ignorieren?*
Stellen Sie aiT mithilfe der in der Übung vorgestellten Annotationen so ein, dass die Analyse ohne Warnung gelingt. *Wie unterscheiden sich die Ergebnisse zwischen dem ersten Durchlauf und dem ohne Warnung?*
Für welche Arten von Echtzeitsystemen ist die werkzeuggestützte WCET-Bestimmung gut geeignet? Für welche weniger gut? Inwiefern ist das Ausführungsmodell von eCos für die werkzeuggestützte WCET-Bestimmung problematisch? Haben Sie eine Idee, wie ein günstigerer Ansatz aussehen könnte?

☞ make aiT

4. *Analyse von quicksort:* Versuchen Sie nun die Funktion `quicksort_job()` ebenfalls zu analysieren. *Wieso gibt die Analyse diesmal Warnungen aus? Welche Programmzeilen erschweren die automatische Analyse am meisten? Wieso wird im Echtzeitbereich für gewöhnlich Heapsort, nicht aber Quicksort verwendet?*

5. *WCET-Analyse-freundliche Entwurfsmuster:* Betrachten Sie nun die Funktion `sample_job()`. *Inwiefern ist das in dieser Funktion verwendete Entwurfsmuster für eine WCET-Analyse schlecht geeignet?*

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.**

Erweiterte Aufgabe

1. *Verbesserung der fehlerhaften* `quicksort()` *Analyse:* Annotieren Sie nun die Funktion `quicksort()` so, dass aiT eine möglichst genaue Abschätzung der WCET liefert.

Hinweise

- Bearbeitung: Gruppe mit je drei Teilnehmern.
- Abgabezeit: 25.11.2015
- Fragen bitte an `i4ezs@lists.cs.fau.de`