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.

1 Aufgabenstellung

1.1 Zeitmessung mit der libEZS:

Wir haben für Sie bereits die Funktionen `checksum()` und `bubblesort()` implementiert.

Teilaufgabe 1.
Von welchen Faktoren ist die Ausführungszeit dieser Funktionen abhängig?
Antwort:

Teilaufgabe 2.
Wie unterscheiden sich die von Ihnen erwarteten Laufzeiten der beiden Funktionen in Abhängigkeit von den Eingabewerten und der Eingabe(array)größe?
Antwort:

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 mit einem Stichprobenumfang von mindestens $N = 100$. Versuchen Sie dabei Störungen der Ausführungsumgebung so gut wie möglich zu

☞ `ezs_print_measurement()`

eliminieren bzw. zu berücksichtigen (`http://ecos.sourceware.org/ecos/docs-2.0/ref/kernel-interrupts.html`).

Teilaufgabe 3.

Welche Anforderungen sind an die Stichprobe zu stellen? Wieso fordern wir in dieser Aufgabe einen Stichprobenumfang von mindestens 100?

Antwort:

Teilaufgabe 4.

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?

Antwort:

Teilaufgabe 5.

Wie groß ist der Mittelwert und dessen Standardfehler bei dieser Eingabe? Wie groß ist die *Spanne* zwischen gemessenem Minimal- und Maximalwert?

Antwort:

Teilaufgabe 6.

Welche Bedeutung haben Mittelwert, Standardabweichung, Standardfehler, Minimum und Maximum? Wie hängen diese Werte mit der WCET zusammen?

Antwort:

1.2 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 PD12 als GPIO zu verwenden.

Teilaufgabe 7.

Wie können Sie mit Hilfe des GPIO-Pins Ausführungszeiten messen?
Antwort:

Teilaufgabe 8.

Führen Sie nun die Messungen der vorangegangenen Teilaufgabe noch einmal mit Hilfe von GPIO-Pin und Oszilloskop durch.

Teilaufgabe 9.

Wie genau stimmen diese mit den Messungen Ihrer Stoppuhr überein?
Antwort:

Teilaufgabe 10.

Wie viele Stichproben schaffen Sie pro Minute mittels Oszilloskop?
Antwort:

Teilaufgabe 11.

Welche der beiden Messarten würden Sie bevorzugen? Warum?
Antwort:

Teilaufgabe 12.
Welche Probleme sehen Sie sowohl bei der WCET-Abschätzung durch den Zeitgeber als auch bei der durch das Oszilloskop?
Antwort:

1.3 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*, das in der Tafelübung vorgestellte Werkzeug, der Firma *AbsInt* zum Einsatz. Laden Sie Ihr Programm, wie in den Übungsfolien dargestellt, in den aiT und analysieren Sie die WCET der Funktionen `bubblesort()` und `bubblesort_job()`. Falls aiT Sie nach einer Lizenzdatei fragt, geben Sie `/proj/i4ezs/tools/absint-license/license.dat` an.

☞ make aiT

Teilaufgabe 13.
Worin liegt aus Sicht von aiT der Unterschied zwischen `bubblesort()` und `bubblesort_job()`? Welche ist besser geeignet um eine realistische WCET-Analyse für die vorliegende Anwendung zu erhalten? Wieso?
Antwort:

Teilaufgabe 14.
Während der Analyse treten Warnungen auf. Weshalb dürfen Sie diese nicht ignorieren?
Antwort:

Stellen Sie aiT mithilfe der in der Übung vorgestellten Annotationen so ein, dass die Analyse ohne Warnung gelingt.

Teilaufgabe 15.
Wie unterscheiden sich die Ergebnisse zwischen dem ersten Durchlauf und dem ohne Warnung?

Antwort:

Teilaufgabe 16.

Für welche Arten von Echtzeitsystemen ist die werkzeuggestützte WCET-Bestimmung gut geeignet? Für welche weniger gut?

Antwort:

Teilaufgabe 17.

Für welche Eingaben weist `bubblesort()` die größtmögliche Laufzeit auf?

Antwort:

1.4 WCET-Analyse-freundliche Entwurfsmuster:

Teilaufgabe 18.

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

Antwort:

Teilaufgabe 19.

Inwiefern ist die werkzeuggestützte WCET-Bestimmung eines eCos Faden problematisch? Haben Sie eine Idee, wie ein günstigerer Ansatz aussehen könnte?

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 Analyse von heapsort

Teilaufgabe 20.
Vermessen und analysieren Sie nun die Funktion `heapsort_job()`. Hinweis: Abhängig von den gewälten Parametern kann die Analyse mittels aiT bis zu 20 Minuten dauern.

2.2 Wort-Case Laufzeit von heapsort

Teilaufgabe 21.
Versuchen Sie eine möglichst lange Laufzeit der Funktion `heapsort_job()` für ein Eingabearray der Länge 500 zu erzwingen. Die Laufzeitdauer welche Sie dabei zwingend überschreiten sollen ist $12542\,\mu s$. Die Gruppe welche die längste Laufzeit demonstrieren kann, erhält eine Belohnung.

Hinweise

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