

WCET-Analyseverfahren in der automobilen Softwareentwicklung

Martin Däumler¹ Robert Baumgartl² Matthias Werner¹

¹ Technische Universität Chemnitz

² HTW Dresden

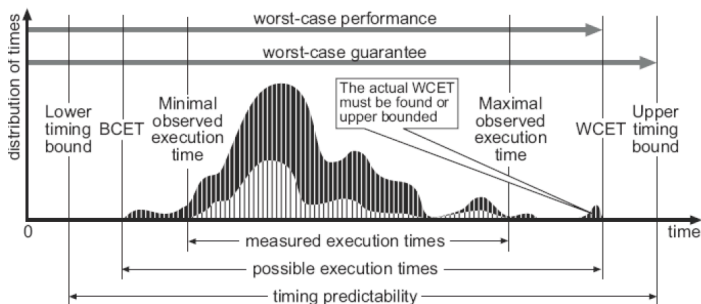
28. November 2008

Inhaltsverzeichnis

1. Einleitung
2. Softwarebasierte dynamische WCET-Analyse
3. Hardwarebasierte dynamische WCET-Analyse
4. Statische WCET-Analyse mit aiT
5. Dynamische WCET-Analyse mit Tessy
6. Fazit

Grundlagen der WCET-Analyse

- ▶ **Definition:** Die Worst-Case Execution Time (WCET) ist die längste Ausführungszeit von Code auf einer bestimmten Hardwareplattform.
- ▶ **Ziel der WCET-Analyse:** Ermittlung einer sicheren (und knappen) oberen Schranke der WCET.



Quelle: A. Ermedahl et al, The Worst-Case Execution Time Problem – Overview of Methods and Survey of Tools

Systembeschreibung

- ▶ Durchführung einer Studie bei ZF Friedrichshafen AG im Rahmen des TIMMO-Projektes
- ▶ Steuergerät mit Infineon TriCore 1766-Prozessor
- ▶ Betriebssystem nach OSEK-OS-Standard: Zyklische Tasks, asynchrone Interrupts
- ▶ Für zugeliessene Software ist typischerweise kein Quellcode verfügbar.

- ▶ Struktur des untersuchten Codes:

```

Ueberwachungsfunktion() {
    IF "fehler" THEN {
        Unterfunktion5();
    }
    ELSE {
        Unterfunktion1();

        IF "fehler" THEN {
            Unterfunktion5();
        }

        Unterfunktion2();

        IF "fehler" THEN {
            Unterfunktion5();
        }
    }

    Unterfunktion3();

    Unterfunktion4();
};

```

```

Unterfunktion1() {
    while(schleife1)
        ...;

    IF "fehler" THEN {
        Unterfunktion5();
    }

    while(schleife2)
        ...;

    IF "fehler" THEN {
        Unterfunktion5();
    }
};

```

```

Unterfunktion2() {
    while(schleife1)
        ...;

    IF "fehler" THEN {
        Unterfunktion5();
    }
};

```

Bewertungskriterien

1. **Stabilität des Verfahrens:** Sind die Ergebnisse reproduzierbar? Sind sie konzeptionell bedingt sicher? Werden die Ausführungszeiten durch das Verfahren selbst verzerrt?
2. **Einsatzzeitpunkt im Softwareentwicklungsprozess:** Ab welchem Schritt im Softwareentwicklungsprozess (V-Modell) kann das Verfahren eingesetzt werden?
3. **Anwendbarkeit des Verfahrens:** Müssen für eine Analyse Software- bzw. Hardware-Interns bekannt sein?
4. **Anwendbarkeit auf zukünftige Software- und Hardwaregenerationen**

Softwarebasierte Aufzeichnung der Ausführungszeiten

- ▶ Die Ausführungszeiten einzelner Tasks (oder Prozesse) werden während des Betriebs des Systems mit dem Tool INCA softwarebasiert aufgezeichnet.
 - ▶ Prozesse sind nicht-unterbrechbare Abschnitte einer Task.
- ▶ Die untersuchte Funktion wurde zur Analyse in einen leeren Prozess verschoben.
- ▶ Während der Messung wurde das Steuergerät skriptbasiert stimuliert, wobei verschiedene Fahrsituationen simuliert wurden.
- ▶ Dieses Verfahren ist der gegenwärtige Stand der Technik im Unternehmen.

Softwarebasierte Aufzeichnung der Ausführungszeiten

1. **Stabilität:**

- ▶ Die Ergebnisse sind unsicher, da nicht verifiziert werden kann, ob die Worst-Case-Situation gemessen wurde.
- ▶ Das Verfahren selbst beeinflusst das Timing-Verhalten des Systems.
- ▶ Die Ergebnisse schwanken stark, da ISR-Ausführungszeiten in denen der Tasks/Prozesse enthalten sein können.

2. **Einsatzzeitpunkt:** Nach der Integrationsphase.

3. **Anwendbarkeit des Verfahrens:** Software-Internen müssen nicht bekannt sein, um eine WCET-Analyse durchzuführen. Es können somit alle Softwarekomponenten erfasst werden.

4. **Zukünftige Anwendbarkeit:** Es wird eine Anpassung der Messsoftware erforderlich.

Hardwarebasierte Aufzeichnung der Ausführungszeiten

- ▶ Während des Betriebs des Systems wird über eine OCDS2-Schnittstelle ein Hardware-Trace aufgezeichnet (maximale Dauer: 850 ms).
- ▶ Eine skriptbasierte Auswertung des Traces ermöglicht die Ermittlung der Ausführungszeiten einzelner Tasks, Prozesse, ISR und Funktionen.
- ▶ Dieses Verfahren stellt eine messtechnische Präzisierung der softwarebasierten Aufzeichnung dar, da geringer(er) (Hardware-) Overhead entsteht.
- ▶ Die Auswertung des Hardware-Traces nimmt an einem Arbeitsplatzrechner teilweise mehrere Stunden in Anspruch und ist damit unpraktikabel.

Hardwarebasierte Aufzeichnung der Ausführungszeiten

1. **Stabilität:**

- ▶ Die Ergebnisse sind unsicher, da nicht verifiziert werden kann, ob die Worst-Case-Situation gemessen wurde.
- ▶ Die kurze Messdauer erschwert es, gezielt eine kritische Situation zu messen.
- ▶ Manche Ausführungszeiten konnten zum Zeitpunkt der Untersuchung nur auf Task- und nicht auf Prozessebene erfasst werden.

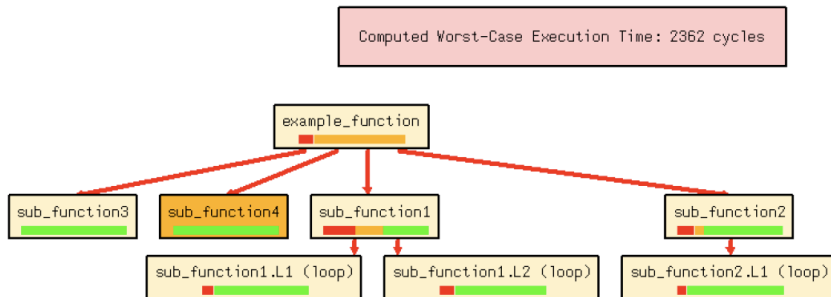
2. **Einsatzzeitpunkt:** Nach der Integrationsphase.

3. **Anwendbarkeit des Verfahrens:** Software-Internas müssen nicht bekannt sein, um eine WCET-Analyse durchzuführen. Es können somit alle Softwarekomponenten erfasst werden.

4. **Zukünftige Anwendbarkeit:** Das System muss über eine OCDS2-Schnittstelle verfügen.

Statische WCET-Analyse mit aiT

- ▶ Das Tool aiT berechnet anhand eines Binärs des zu untersuchenden Codes und ggf. notwendigen Nutzerangaben eine obere Schranke der WCET.
- ▶ Detaillierte Angaben zur Hardware und zum Programmfluss können die Analyse präzisieren.
- ▶ aiTs Ausgabe ist ein Aufrufgraph mit detaillierten Informationen zur ermittelten WCET:



Statische WCET-Analyse mit aiT

1. **Stabilität:**

- ▶ Die Ergebnisse werden als sicher angenommen und sind reproduzierbar.
- ▶ Die resultierenden WCETs eines manuell erzeugten und finalen Binärys unterscheiden sich um bis zu 12%.
- ▶ Die Ermittlung aller Nutzerangaben zur Analyse größerer Ausführungseinheiten ist aufwändig, andernfalls könnten pessimistische WCETs berechnet werden.

2. **Einsatzzeitpunkt:** Während der Implementationsphase.

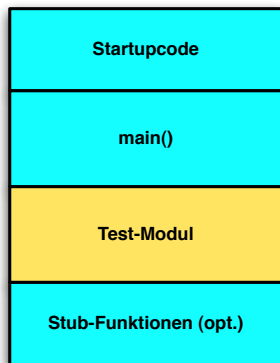
3. **Anwendbarkeit des Verfahrens:** Software- und Hardware-Internas müssen bekannt sein, um eine (präzise) WCET-Analyse durchzuführen. Zugelieferte Softwarekomponenten können somit nicht immer erfasst werden.



4. **Zukünftige Anwendbarkeit:** Die Analyse ist bisher auf Code ohne dynamische Features objektorientierter Programmierung und bestimmte Programmiersprachen beschränkt. Der verwendete Prozessor muss unterstützt werden.

Kontrolliert dynamische WCET-Analyse mit Tessa

- ▶ Tessa erlaubt die Durchführung von Unit-Tests auf dem Target.
- ▶ Ansatz:
 1. Aufteilung des Codes in Module
 2. Spezifikation zeitkritischer Pfade als Testfälle
 3. (automatische) Erzeugung einer Standalone-Applikation
 4. Messen der Ausführungszeit der Unit-Tests
- ▶ Es wurden keine Unit-Tests auf dem Target durchgeführt, da zugeliessene Debugger-Skripte nicht angepasst werden konnten.

- ▶ Struktur einer Tessa-Applikation:



-  von Tessa erzeugter Code
-  unveränderter Quell-Code

Erkenntnisse

- ▶ Gute Integration des derzeitigen Verfahrens, aber unzureichende Genauigkeit.
- ▶ Häufig gestellte Frage: Wahrscheinlichkeit einer WCET und die möglichen Folgen (Scheduling-Analyse).
- ▶ Abwägung zwischen Kosten zur Beseitigung eines Problems und dessen Folgekosten.
- ▶ Schutz Intellectual Property.
- ▶ Die Aufnahme der Timing-Eigenschaften in Standards ist notwendig.
 - ▶ Laufendes Projekt: TIMMO

Zusammenfassung

Tabelle: Bewertung der einzelnen Analyseverfahren (+/o/- $\hat{=}$ gut/mittel/schlecht)

Verfahren	Stabilität	Einsatz- Zeitpunkt	Anwendbarkeit	Zukünftige Anwendbarkeit
INCA	–	–	+	+
HW-Traces	o	–	o	+
aiT	+	+	–	o
Tessy	k. A.	+	–	+

Ende des Vortrags

Vielen Dank für Ihre Aufmerksamkeit.
Fragen?