UNIVERSITÄT ZUM BEISPIEL
INSTITUT FÜR BEISPIELE

Monitoring der AUTOSAR Timing Extensions mittels TeSSLa

Monitoring of the AUTOSAR Timing Extensions with TeSSLa

Bachelorarbeit

im Rahmen des Studiengangs
Informatik
der Universität zu Lübeck

vorgelegt von
Hendrik Streichhahn

ausgegeben und betreut von
Prof. Dr. Martin Leucker

mit Unterstützung von
Martin Sachenbacher

Lübeck, den 1.1. 1970

Erklärung

Ich erkläre hiermit an Eides statt, dass ich diese Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel benutzt habe.

(Hendrik Streichhahn)
Lübeck, den 1.1. 1970

Kurzfassung Abstract Deutsch

Abstract Kurzfassung Englisch.

Inhaltsverzeichnis

1. Einleitung **1**
1.1. Verwandte Arbeiten . . . 1
1.2. Aufbau der Arbeit . . . 2

2. Grundlagen **3**
2.1. TeSSLa . . . 3

3. Monitorbarkeit **5**
3.1. Quellen . . . 5
3.2. Tabellen . . . 6
3.3. Abbildungen und Diagramme . . . 6
3.4. Quelltext . . . 6
3.4.1. Quelltext mit automatischer Nummerierung . . . 8
3.5. Pseudocode . . . 8
3.6. Formeln mit ε . . . 8
3.7. Theoreme . . . 9
3.8. Anführungszeichen . . . 9

4. AUTOSAR TIMEX Constraints **11**

5. Implementierungen **13**
5.1. Implementierungen . . . 13

6. Zusammenfassung und Ausblick **15**

A. Anhang **17**
A.1. Abschnitt des Anhangs . . . 17

Liste der Todos

Die Abschnitte dieses Kapitels sollten natürlich nicht so in die Arbeit übernommen werden. 5

Notizen an einen selbst oder den Betreuer der Arbeit sind während der Arbeit sehr nützlich. Für die finale Version können diese Todo-Notes dann komplett aufgeblendet werden. 5

1. Einleitung

Das Zeitverhalten ist eine der wichtigsten Eigenschaften von vielen Hard- und Software. Insbesondere in sicherheitskritischen Anwendungen kann eine zeitlich falsche Reaktion verheerende Folgen haben, so kann zum Beispiel eine verfrühte oder verspätete Reaktion eines Herzschrittmachers das Leben eines Menschen gefährden. In Cyber-Physical-Systems, wie zum Beispiel der Fahrdynamikregelung in Kraftfahrzeugen (ESP), kann ein fehlerhaftes Zeitverhalten hohe Personen- und Sachschäden hervorbringen, durch die Vernetzung verschiedener Komponenten wird das Erstellen und die Analyse des Zeitverhalten aber erheblich erschwert, da nicht nur die einzelnen Komponenten, sondern auch das Gesamtsystem untersucht werden muss. Auch Umweltschutzaspekte können eine Rolle spielen, da z.B. eine zeitlich fehlerhafte Steuerung eines Verbrennungsmotors zu erhöhten Emissionen führen kann. Der Wichtigkeit des Zeitverhaltens steht der große Aufwand einer Zeitanalyse und somit auch wirtschaftliche Aspekte, so dass abgewägt werden muss, ob und in welcher Tiefe eine Analyse des Zeitverhaltens nötig ist.
Nicht nur in der Entwicklung von Systemen ist die Analyse des Zeitverhaltens ein wichtiger Bestandteil, auch im Betrieb von diesem sollte das Zeitverhalten des Systems und der einzelnen Komponenten geprüft werden, da Schäden an einzelnen Bauteilen nicht ausgeschlossen werden können. Ein frühzeitiges Erkennen dieser Schäden kann Folgeschäden verhindern, außerdem wird die Instandsetzung des Systems erleichtert, da bei der Verwendung geeigneter Monitoringtools die Eingrenzung des Fehlers erleichtert wird.
In dieser Arbeit geht es um die Entwicklung eines Monitoringtools, mit dessen Hilfe die online Überwachung von Zeitverhalten, also der Prüfung von Eigenschaften in *nahezu* Echtzeit, ermöglicht wird. Im Fokus der Entwicklung liegt der Automobilbereich, die Ergebnisse sind aber auf andere Bereiche übertragbar.

1.1. Verwandte Arbeiten

AUTOSAR (AUTomotive Open System ARchitecture) ist eine Partnerschaft aus Automobilherstellern und dazugehörigen Software, Hardware Unternehmen, deren Zulieferern und weiteren. Ziel dieser Partnerschaft ist die Erstellung offener Standards für Soft- und Hardwarekomponenten im Automobilbereich, sowie deren Entwicklungsprozesse [AUT20].

Die AUTOSAR Timing Extensions (kurz **AUTOSAR TIMEX**) spezifieren Constraints, mit denen das Zeitverhalten von Komponenten, die mit Hilfe anderer AUTOSAR Standards definiert wurden, beschrieben werden kann [AUT18].
TODO [BFL+12]
TeSSLa (Temporal Stream-based Specification Language) ist eine turingfähige Programmiersprache, die zur Analyse und zur Überwachung von Zeitverhalten, insbesondere das von Cyber-Physical Systems. Es nimmt dabei Ströme von Datenpunkten, die mit Zeitstempeln verknüpft sind, entgegen und führt auf diesen Berechnungen durch [CHL+18].
Damit eng verknüpft ist das **COEMS**-Project, in dem Möglichkeiten von hardwarebasierte, non-intrusive, online Stream Runtime Verification erarbeitet wurden. Hierbei werden vorhandene Debug-Informationen aus einem System mittels einer TeSSLa-Spezifikation, die auf eine FPGA-basierter Hardware übertragen wurde, analysiert.

1.2. Aufbau der Arbeit

Neben dieser Einleitung und der Zusammenfassung am Ende gliedert sich diese Arbeit in die folgenden drei Kapitel.

Kapitel 2 beschreibt die für diese Arbeit benötigten Grundlagen. In diesem Kapitel werden . . . , . . . und . . . eingeführt, da diese für die folgenden Kapitel dringend benötigt werden.

Kapitel 3 stellt das eigentliche Konzept vor. Dabei handelt es sich um ein Konzept zur Verbesserung der Welt. Das Kapitel gliedert sich daher in einen globalen und einen lokalen Ansatz, wie die Welt zum Besseren beeinflusst werden kann.

Kapitel 5 beinhaltet eine Evaluation des Konzeptes aus dem vorherigen Kapitel. Anhand von Simulationen wird in diesem Kapitel untersucht, wie die Welt durch konkrete Maßnahmen deutlich verbessert werden kann.

2. Grundlagen

2.1. TeSSLa

TeSSLa (**Te**mporal **S**tream-based **S**pecification **La**nguage) ist eine funktionale Programmiersprache, die für die Laufzeitverifikation von Datenströmen konzipiert wurde. In TeSSLa sind Ströme als Folgen von Events definiert, wobei ein Event aus einem Element der jeweiligen Datentypmenge $\mathbb{D}$ sowie aus einem Zeitwert aus der diskreten Zeitdomäne $\mathbb{T}$ besteht. In dieser muss es eine totale Ordnung geben und ein Event b, welches zeitlich nach einem Event a auftritt, muss einen höheren Zeitwert aufweisen. Innerhalb einer Spezifikation können mehrere Ströme aus unterschiedlichen Datentypmengen $\mathbb{D}_1, \ldots, \mathbb{D}_n$ verwendet werden, wobei die Zeitdomäne $\mathbb{T}$ innerhalb einer Spezifikation auf allen Strömen dieselbe sein muss.
In TeSSLa wird zwischen synchronen Strömen, in denen alle Ströme einer Spezifikation Events in gemeinsamen Zeitpunkten haben, und asynchronen Strömen unterschieden, bei denen die Zeitpunkte der Events zwar einer globalen Ordnung folgen, die Zeitpunkte aber sonst unanhängig von einander sind.Die Spezifikationen mit synchronen Strömen sind eine echte Teilmenge der Spezifikationen mit asynchronen Strömen, da Ströme mit geordneten, aber ansonsten unabhängigen Zeitpunkten Events mit gleichzeitigen Zeitpunkten auf allen Strömen zulassen. Andersherum gilt diese Relation offensichtlich nicht. Aufgrund dieser Teilmengenrelation werden im Folgenden nur asynchrone Ströme behandelt, wenn von Strömen die Rede ist, sind immer asynchrone gemeint.
Die Berechnungen erfolgen, nachdem sie von eintreffenden Events gestartet wurden, wodurch ein Ausgabestrom mit den gleichen Zeitwerten wie Eingabeströme, allerdings kann der *delay*-Operator verwendet werden, um neue Zeitpunkte zu erzeugen, dazu später mehr. Ohne neue Zeitpunkte heißt die Spezifikation *timestamp conservative*. Innerhalb dieser Berechnungen sind Direktzugriffe nur auf die aktuellen Datenwerte der Ströme möglich. Diese Werte bleiben solange bestehen, bis ein neues Event auf diesem Strom eintrifft, der Zeitwert des Events ändert sich hierbei nicht. Mit dem *last*-Operator, welcher auch rekursiv angewendet werden kann, sind Zugriffe auf das jeweils letzte Element möglich. Der *lift*-Operator wendet eine Funktion über Datenwerten auf die Datenwerte jedes eintreffenden Events an. Der *slift*-Operator agiert ähnlich dem *lift*-Operator, allerdings wird die Funktion erst dann angewendet, wenn auf jedem Strom, der dem *slift*-Operator übergeben wurde, bereits ein Event übertragen wurde. (TODO→ weiter ausführen)

In [CHL+18] werden verschiedene Fragmente von TeSSLa beschrieben, die unterschiedliche Mächtigkeiten haben und äquivalent zu verschiedenen Transduktormodellen sind. Im Fragment $TeSSLa_{bool}$ sind die Datentypmengen der Ströme auf boolesche Werte beschränkt, als Operatoren sind nur der oben genannte *last*-Operator, der *lift*-Operator

3. Monitorbarkeit

In diesem Kapitel wird die eigentliche Erkenntnis dieser Arbeit beschrieben. Der Aufbau dieses Kapitels hängt stark vom Thema der Arbeit ab. Die in dieser Vorlage vorgeschlagenen Kapitel sind auch nur als Vorschlag und auf keinen Fall als verbindlich zu verstehen.

Die folgenden Abschnitte dieses Kapitels enthalten Beispiele für die diversen Inhaltselemente einer Arbeit.

Notizen an einen selbst oder den Betreuer der Arbeit sind während der Arbeit sehr nützlich. Für die finale Version können diese Todo-Notes dann komplett aufgeblendet werden.

Die Abschnitte dieses Kapitels sollten natürlich nicht so in die Arbeit übernommen werden.

3.1. Quellen

Quellen sind wichtig für gutes wissenschaftliches Arbeiten. Eine Quelle kann dabei zum Beispiel

- ein Beitrag in einer Zeitschrift [**?**],
- ein Beitrag in einem Sammlungsband [**?**],
- ein Buch [**?**],
- ein Beitrag im Berichtsband einer Konferenz [**?**],
- ein technischer Bericht [**?**],
- eine Dissertation [**?**],
- eine Abschlussarbeit [**?**],
- ein (noch) nicht veröffentlichter Artikel [**?**] oder
- ein Artikel auf einer Website [**?**] sein.

Dabei ist zu beachten, dass nicht veröffentlichte Artikel und insbesondere Webseiten nur in Ausnahmefällen gute Quellen sind, da diese nicht durch Fachleute begutachtet wurden.

Im Bereich der Informatik können Quellenangaben im BibTEX-Format direkt der dblp[1] entnommen werden.

[1] zum Beispiel `http://dblp.uni-trier.de`

Jahr	Prozessor	MHz
1975	6502 (C64)	1
1985	80386	16
2005	Pentium 4	2 800
2030	Phoenix 3	7 320 000
2050	...	
2070	...	

Tabelle 3.1.: Rechengeschwindigkeit von Computern. Inhaltlich vollkommen egal, ist dies doch ein sehr schönes Beispiel für eine Tabelle.

3.2. Tabellen

In Tabelle 3.1 sehen wir ein Beispiel für eine Tabelle.

3.3. Abbildungen und Diagramme

In Abbildung 3.1 auf der nächsten Seite sehen wir ein Beispiel für eine Abbildung, die aus einer externen Grafik geladen wurde. In Abbildung 3.2 auf der nächsten Seite sehen wir ein Beispiel für eine Abbildung, die in LaTeX generiert wurde.

3.4. Quelltext

Quelltext sollte in Abschlussarbeiten nur äußerst sparsam eingesetzt werden. Wichtig ist insbesondere, dass Quelltextauszüge sorgsam ausgewählt und gut erklärt werden.

```
public class Main {
  // Hello Word in Java
  public static void main(String[] args) {
    System.out.println("Hello World");
  }
}
```

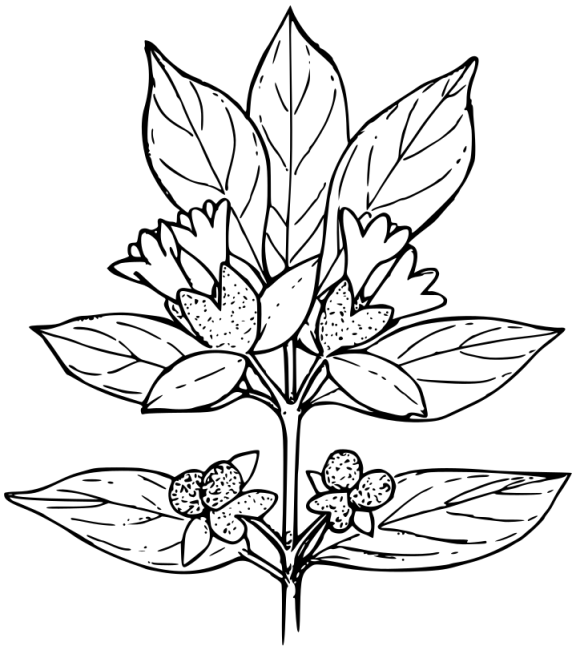

Abbildung 3.1.: Lange Version der Beschreibung, die direkt unter der Abbildung gesetzt wird. Es ist wichtig, für jede Abbildung eine umfangreiche Beschreibung anzugeben, da der Leser beim ersten Durchblättern der Arbeit vor allem an den Abbildungen hängen bleibt.

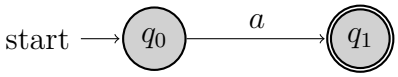

Abbildung 3.2.: Graph des Büchi-Automaten $\hat{A}$. Der Zustand q_1 hat dabei keine ausgehende Kante. Der Zustand ist trotzdem akzeptierend, da beide enthaltenen Zustände von $\acute{A}$ akzeptierend sind. Die naive Anwendung des Leerheitstests auf alternierenden Büchi-Automaten liefert in diesem Fall also zu viele akzeptierende Zustände.

3.4.1. Quelltext mit automatischer Nummerierung

Manchmal möchte man Quelltext eher als Abbildung und nicht als Fließtext behandeln. In diesem Fall soll der Quelltext eine Bildunterschrift und eine automatische Nummerierung erhalten. Die automatische Nummerierung kann dann natürlich auch in Referenzen verwendet werden: In Quelltext 3.1 haben wir Java-Quelltext.

```
public class AnotherClass {
  private int number = 0;
  public void add() {
    this.number++;
  }
}
```

Quelltext 3.1: Ich bin die Bildunterschrift des Quelltextes

Wenn man Quelltext mit Bildunterschrift setzt, muss man darauf achten, dass der Quelltext nach wie vor nicht als Fließumgebung behandelt wird. Entsprechend kann es passieren, dass der Quelltext über zwei Seiten hinweg gesetzt wird. Während das normalerweise nicht stört, kann dieser Umstand in Zusammenhang mit einer Bildunterschrift den Leser irritieren.

3.5. Pseudocode

Um Algorithmen zu erklären ist Pseudocode viel besser geeignet als Quelltext. Im Pseudocode kann man alles unwichtige weglassen und sich auf die mathematische Modellierung des Algorithmus konzentrieren. So kann die Struktur des Verfahrens unabhängig von Implementierungsdetails des jeweiligen Frameworks erklärt werden.

```
// Schleife von 1 bis 5
  for i ← 1 to 5 do
    while S[i] ≠ S[S[i]] do
      S[i] ← S[S[i]]
```

3.6. Formeln mit ε

Das ε in der Überschrift dieses Abschnitts ist ein Beispiel für ein mathematisches Symbol, dass in den PDF-Lesezeichen als reiner Text gesetzt wird. Siehe `macros.tex`. In dieser Datei wird auch $n \in \mathbb{N}$ definiert.

Wir wissen aus der Analysis, dass

$$f(x) = x^2 + px + q$$

Nullstellen bei

$$x_1 = -\frac{p}{2} + \sqrt{\frac{p^2}{4} - q} \text{ und}$$
$$x_2 = -\frac{p}{2} - \sqrt{\frac{p^2}{4} - q}$$

hat.

3.7. Theoreme

Definition 3.1 (Definition)**.** Eine *Definition* ist die Bestimmung eines Begriffs oder eines mathematischen Zusammenhangs.

Lemma 3.2 (Unwichtiger Hilfssatz)**.** *Der Satz ...*

Beweis. ... und sein Beweis. □

Theorem 3.3 (Wichtiger Satz)**.** *Ein wichtiger Satz.*

Beweis. Der Beweis folgt unter Verwendung der Erkenntnisse aus Lemma 3.2. □

Korollar 3.4 (Ein Geschenk)**.** *Eine unmittelbare Folgerung.*

Beweis. Die Folgerung folgt unmittelbar aus Theorem 3.3. □

Beispiel 3.5. Ein Beispiel.

Bemerkung 3.6. Beispiele und Bemerkungen werden nicht in das Verzeichnis der Theoreme und Definitionen aufgenommen.

3.8. Anführungszeichen

Anführungszeichen werden nur für wörtliche Rede oder direkt übernommene kurze Zitate verwendet. Für die Hervorhebung von neu eingeführten Fachbegriffen eignet sich *kursiver Satz* besser.

4. AUTOSAR TIMEX Constraints

5. Implementierungen

In der Evaluierung wird das Ergebnis dieser Arbeit bewertet. Eine praktische Evaluation eines neuen Algorithmus kann zum Beispiel durch eine Implementierung geschehen. Je nach Thema der Arbeit kann sich natürlich auch die gesamte Arbeit eher im praktischen Bereich mit einer Implementierung beschäftigen. In diesem Fall gilt es am Ende der Arbeit insbesondere die Implementierung selber zu evaluieren. Wesentliche Fragen dabei können sein:

– Was funktioniert jetzt besser als vor meiner Arbeit?
– Wie kann das praktisch eingesetzt werden?
– Was sagen potenzielle Anwender zu meiner Lösung?

5.1. Implementierungen

Wenn Implementierungen umfangreich beschrieben werden, ist darauf zu achten, den richtigen Mittelweg zwischen einer zu detaillierten und zu oberflächlichen Beschreibung zu finden. Eine Beschreibung aller Details der Implementierung ist in der Regel zu detailliert, da die primäre Zielgruppe einer Abschlussarbeit sich nicht im Detail in den geschriebenen Quelltext einarbeiten will. Die Beschreibung sollte aber durchaus alle wesentlichen Konzepte der Implementierung enthalten. Gerade bei einer Abschlussarbeit am Institut für Softwaretechnik und Programmiersprachen lohnt es sich, auf die eingesetzten Techniken und Programmiersprachen einzugehen. Ich würde in einer solchen Beschreibung auch einige unterstützende Diagramme erwarten.

6. Zusammenfassung und Ausblick

Die Zusammenfassung greift die in der Einleitung angerissenen Bereiche wieder auf und erläutert, zu welchen Ergebnissen diese Arbeit kommt. Dabei wird insbesondere auf die neuen Erkenntnisse und den Nutzen der Arbeit eingegangen.

Im anschließenden Ausblick werden mögliche nächste Schritte aufgezählt, um die Forschung an diesem Thema weiter voranzubringen. Hier darf man sich nicht scheuen, klar zu benennen, was im Rahmen dieser Arbeit nicht bearbeitet werden konnte und wo noch weitere Arbeit notwendig ist.

A. Anhang

Dieser Anhang enthält tiefergehende Informationen, die nicht zur eigentlichen Arbeit gehören.

A.1. Abschnitt des Anhangs

In den meisten Fällen wird kein Anhang benötigt, da sich selten Informationen ansammeln, die nicht zum eigentlichen Inhalt der Arbeit gehören. Vollständige Quelltextlisting haben in ausgedruckter Form keinen Wort und gehören daher weder in die Arbeit noch in den Anhang. Darüber hinaus gehören Abbildungen bzw. Diagramme, auf die im Text der Arbeit verwiesen wird, auf keinen Fall in den Anhang.

Abbildungsverzeichnis

3.1. Kurzfassung der Beschreibung für das Abbildungsverzeichnis 7
3.2. Graph des Büchi-Automaten $\hat{A}$. 7

Tabellenverzeichnis

3.1. Rechengeschwindigkeit von Computern 6

Definitions- und Theoremverzeichnis

3.1. Definition (Definition) 9
3.2. Lemma (Unwichtiger Hilfssatz) 9
3.3. Theorem (Wichtiger Satz) 9
3.4. Korollar (Ein Geschenk) 9

Quelltextverzeichnis

3.1. Ich bin die Bildunterschrift des Quelltextes 8

Abkürzungsverzeichnis

TDO zu erledigen *T*o *D*o

Literaturverzeichnis

[AUT18] AUTOSAR: Specification of Timing Extensions / AUTOSAR. 2018. – Forschungsbericht. – Version 4.0

[AUT20] AUTOSAR: *AUTOSAR History*. `https://www.autosar.org/about/history/`, 2020. – Online; Zugriff am 24.8.2020

[BFL+12] Blom, Hans ; Feng, Dr. L. ; Lönn, Dr. H. ; Nordlander, Dr. J. ; Kuntz, Stefan ; Lisper, Dr. B. ; Quinton, Dr. S. ; Hanke, Dr. M. ; Peraldi-Frati, Dr. Marie-Agnès ; Goknil, Dr. A. ; Deantoni, Dr. J. ; Defo, Gilles B. ; Klobedanz, Kay ; Özhan, Mesut ; Honcharova, Olha: Language syntax, semantics, metamodel V2 / ITEA2. 2012. – Forschungsbericht

[CHL+18] Convent, Lukas ; Hungerecker, Sebastian ; Leucker, Martin ; Scheffel, Torben ; Schmitz, Malte ; Thoma, Daniel: *TeSSLa: Temporal Stream-based Specification Language*. 2018