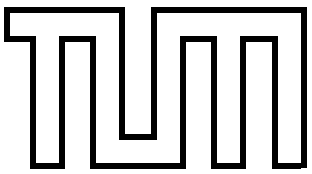


Technische Universität München

Fakultät für Informatik
Forschungs- und Lehreinheit Informatik VI

Übung zur Vorlesung Echtzeitsysteme

Aufgabe 5 – Verifikation mit Esterel

Simon Barner
barner@in.tum.de

Irina Gaponova
gaponova@in.tum.de

Stephan Sommer
sommerst@in.tum.de

Wintersemester 2009/10

Aufgabe 2: Verifikation mit Esterel

In der Vorlesung haben Sie bereits die Techniken der formalen Verifikation kennengelernt. Dieses Übungsblatt bietet dazu einige Aufgaben, um den Umgang mit einem Verifizierungswerkzeug kennen zu lernen und die Inhalte aus der Vorlesung anhand praktischer Beispiele weiter zu vertiefen.
Für das Bearbeiten der Aufgaben wird Esterel Studio benutzt, dass die Möglichkeit der Verifikation bereits integriert hat. Die Eigenschaften der Aufzugssteuerung von der letzten Übung werden verifiziert. Damit die gewünschten Eigenschaften verifizierbar sind, muss das Program zuerst entsprechend erweitert werden. Wie in der letzten Übung werden die Syntax von Esterel und der Esterel Studio Safe State Machine (SSM) Editor verwendet. Sowohl eine Kurzreferenz als auch eine vollständige Sprachbeschreibung zu Esterel finden Sie im Ordner `Uebung03` auf dem Laufwerk Q:. Als Ausgangspunkt für diese Übung wird eine Lösung der Aufgaben aus dem Übungsblatt 3 verwendet. Sie finden die entsprechenden Dateien im Ordner `Uebung04` auf dem Laufwerk Q:. Kopieren Sie diese bitte zuerst vom Netzlaufwerk auf Ihren lokalen Rechner.

Allgemeines

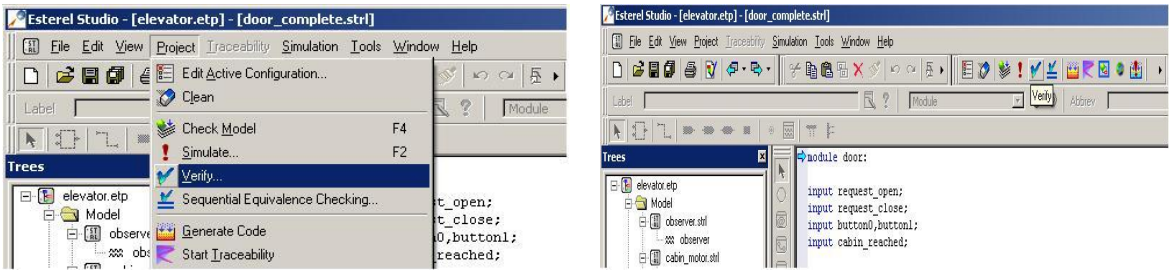
Für die Übungsaufgaben wird die Entwicklungsumgebung *Esterel Studio* von Esterel Technologies verwendet (*Start* → *Programme* → *Esterel Studio* → *Esterel Studio*). Wer sich außerhalb der Übung weiter mit Esterel beschäftigen will, findet unter `http://www-sop.inria.fr/meije/esterel/esterel-eng.html` einen kostenlosen Esterel-Compiler.

Hinweis: Verifikation mit Esterel

Esterel hat einen integrierten Model Checker, der die Verifikation ermöglicht. Um eine Eigenschaft zu verifizieren, sind folgende Schritte nötig:

- Starten der Verifikationsumgebung über den Menüpunk *Project* → *Verify* (siehe Abb. 1(a)) oder direkt über den Menüpunkt *Verify* (siehe Abb. 1(b)).
- Durch einen Rechtsklick auf einen Signalnamen können Sie die folgenden Optionen auswählen: (siehe Abb. 2)
  - Always Present
  - Always Absent

Nach dem Auswählen der entsprechenden Eigenschaft erscheint ein Eintrag unter *Verification* → *Goals*. Der Verifikationsvorgang wird mit *Prove* gestartet. Am Ende kann man dem *Status* entnehmen, ob die Eigenschaft nachgewiesen ist oder nicht. Das Ablaufprotokoll



(a) Starten de Verifikationsumgebung 1 (b) Starten de Verifikationsumgebung 2

Abbildung 1: Starten de Verifikationsumgebung

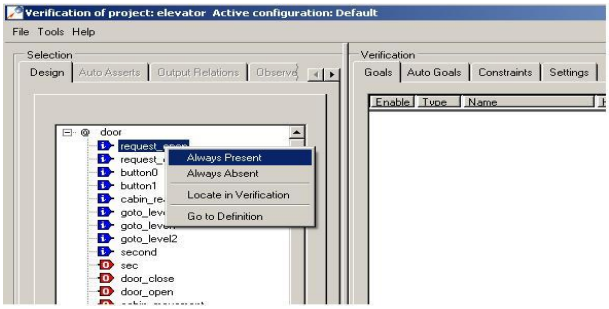


Abbildung 2: Optionen zum Verifizieren

der Verifikation wird in der Datei *$ProjektVerzeichnis$/Verifikation/*<Signal Name>.es gepeichert.

Sie können mittels *Verify* in Esterel Studio überprüfen, on ein Signal nie anwesend ist oder on ein Signal immer anwesend ist. Das heisst, um eine Eigenschaft zu verifizieren, brauchen Sie ein entsprechendes Signal.

Hinweis: Einstellungen für Verifikation

Für die Übung reicht die symbolische Verifikation aus. Gehen Sie auf *Verify* → *Settings* → *Prove* → *Symbolic mode cheking (BDD)*

Hinweis: Verifikation

In der formalen Verifikation unterscheidet man zwischen folgenden Eigenschaften:

- *Liveness Properties*: Zustand x triff immer nach Eintreten des Zustandes y auf
- *Reachability Properties*:ein System erreicht irgendwann einen bestimmten Zustand (und evtl. dort verbleibt)

- *Safety Properties*: Fehlerzustände werden nie erreicht werden

Aufgabe 5.1: Liveness Properties

Weisen Sie nach, dass nachdem der Knopf 'Level1' gedrückt wurde, der Aufzug immer innerhalb von max. 30 Sekunden im ersten Stock ankommt.

Hinweis: Aufgabe 5.1

Achten Sie bitte darauf, dass der Esterel Model Checker bei der Verifikation keine Graphen unterstützt. D.h. ein Signal, z.b. 'goto_level1' oder 'button1', das als ein Ein- oder Ausgabesignal nur in einem Graphen generiert wird, kann in dem Model Checker nicht benutzt werden. Nutzen Sie daher Signale, die in einem Esterel Programm Module generiert werden, für die Verifikation. Für die erste Aufgabe sind das z.B. 'cabin_go1' und 'level1_reached'. Sie kommen beide im Modul *door* vor:

```
case cabin_go1 do
        abort
                sustain cabin_go;
        when level1_reached;
        emit cabin_reached;
```

Aufgabe 5.2: Reachability Properties

Weisen Sie nach, dass es für den Aufzug möglich ist, alle 3 Stockwerke zu erreichen.

Hinweis: realisieren Sie diese Aufgabe durch Negation. Wenn die Eigenschaft 'x kommt nie vor' fehlschlägt, dann haben Sie zumindest einen Fall nachgewisen, wenn x vorkommt.

Aufgabe 5.3: Safety Properties 'Lichtschranke'

Erweitern Sie die Aufzugsteuerung um eine Lichtschranke. Führen Sie dazu ein neues Signal *LightBarrier* ein. Dieses Signal soll immer dann ausgelöst werden, wenn eine Person in den Aufzug ein/aus- steigt. Erweitern Sie das Programm so, dass die Tür immer aufgeht, wenn jemand ein- oder aussteigt. Weisen Sie nach, dass es nie vorkommt, dass eine Person ein/aussteigt und die Tür gleichzeitig zu geht. Falls der Nachweis fehlschlägt, korrigieren Sie Ihr Programm.
Überprüfen Sie jetzt erneut die in Aufgabe 5.1 geforderten Eigenschaften. Warum schläg die Verifikation jetzt fehl?