Ludwig-Maximilians-Universität München
Institut für Informatik
Prof. Dr. Rolf Hennicker
M.Sc. Annabelle Klarl

SS 2016
Übungsblatt 4

Entwurf und Implementierung paralleler Programme

Aufgabe 1

In einer Fabrik mit Tag- und Nachtschicht wird fortwährend ein Produkt durch Zusammenbauen zweier Teile A und B hergestellt. Die beiden Teile werden unabhängig voneinander produziert aber gleichzeitig bereit gestellt um anschließend zusammengebaut werden zu können. Während des Zusammenbauens können bereits die nächsten beiden Teile produziert werden.

Modellieren Sie den Fertigungsprozess in FSP und geben Sie ein Strukturdiagramm für den Prozess an. Geben Sie auch das zugehörige LTS des Prozesses an.

Aufgabe 2

Ein einelementiger Puffer werde durch folgenden Prozess beschrieben:
BUFF = (in $\rightarrow$ out $\rightarrow$ BUFF).

(a) Modellieren Sie einen zweielementigen Puffer TBUFF durch geeignete Hintereinanderschaltung zweier einelementiger Puffer und geben Sie das Strukturdiagramm des Prozesses an. Geben Sie auch das LTS von TBUFF an.

(b) Wie könnte ein zweielementiger Puffer auch direkt (ohne Zusammensetzen zweier einelementiger Puffer) durch einen sequentiellen FSP Prozess beschrieben werden?

(c) Modellieren Sie das nach außen sichtbare Verhalten des zweielementigen Puffers aus Teil a) durch einen Prozess TWOBUFF und geben Sie das Strukturdiagramm von TWOBUFF an. Geben Sie das LTS von TWOBUFF an. Geben Sie ein dazu beobachtbar äquivalentes LTS mit einer minimalen Menge von Zuständen an. Vergleichen sie dieses mit dem LTS Ihrer Lösung aus Teil b).

(d) Berücksichtigen Sie in den Teilen a) - c) nun auch Daten im Bereich 0..1, die von dem Puffer gemäß des FIFO-Prinzips eingelesen und ausgegeben werden sollen.

Aufgabe 3

Gegeben sei der Prozess ARADIO von Übungsblatt 3, Aufgabe 1; siehe Rückseite.

(a) Definieren Sie einen Prozess ARADIO_1, der die Aktionen *godown*, *gomax*, *goup* und *gomin* des RADIO Prozesses verbirgt und geben Sie das zugehörige, bzgl. beobachtbarer Äquivalenz minimalisierte LTS an.

(b) Definieren Sie einen Prozess ARADIO_2, der in einer Schnittstelle die Aktionen *onoff*, *down* und *up* anbietet und geben Sie das zugehörige, bzgl. beobachtbarer Äquivalenz minimalisierte LTS an.

```
const MAX = 108
const MIN = 88
range F = MIN..MAX

ARADIO = (onoff -> LOCKED[MAX]),

LOCKED[f: F] = ( onoff -> ARADIO
               | down -> ( when (f  > MIN) godown -> DOWN[f-1]
                         | when (f == MIN) gomax  -> DOWN[MAX])
               | up   -> ( when (f  < MAX) goup   -> UP[f+1]
                         | when (f == MAX) gomin  -> UP[MIN])),

DOWN[f: F] = ( onoff -> ARADIO
             | when (f  > MIN) godown -> DOWN[f - 1]
             | when (f == MIN) gomax  -> DOWN[MAX]
             | {lock, up, down} -> LOCKED[f]),

UP[f: F] = ( onoff -> ARADIO
           | when (f  < MAX) goup  -> UP[f+1]
           | when (f == MAX) gomin -> UP[MIN]
           | {lock, up, down} -> LOCKED[f]).
```