Übungen zur Vorlesung
Spezifikation eingebetteter Systeme
Studiengang Informatik
WS 04/05

AG Betriebssysteme FB3
Kirsten Berkenkötter, Jan Peleska

Ausgabe: 9.12.2004
Abgabe: 3.1.2005

Übungszettel 3

Im folgenden findet ihr die Timed CSP–Spezifikation für einen (grob vereinfachten) Bahnübergang, die euch irgendwie bekannt vorkommen sollte. Für jeden Teilprozess sowie für das Gesamtsystem sollen die angegebenen Zusicherungen formal spezifiziert, sowie Folgerungen aus den Zusicherungen begründet werden.

Die formale Spezifikation soll auf Basis der auf den ausgeteilten Kopien angegebenen Syntax erfolgen, außerdem dürfen die Makros `at`, `only` und `not` wie in der Übung besprochen verwendet werden.

```
-------------------------------------------------------------------------------
-- Spezifikation mit Timed CSP
-------------------------------------------------------------------------------

-- Events für den Zug
channel sig_nah, sig_weg, nah, da, weg

-- Events für die Schranke
channel auf, zu

-- Variable position, kann gelesen und beschrieben werden
channel lese_position, schreibe_position : {0..90}

-------------------------------------------------------------------------------
```

Aufgabe 1: Spezifikation des Prozesses der globalen Variablen

```
-------------------------------------------------------------------------------
-- Prozess für die Variable
-------------------------------------------------------------------------------

-- entweder lesen oder schreiben
VAR(value) = lese_position!value -> VAR(value)
            []
             schreibe_position?newValue -> VAR(newValue)

-- am Anfang ist die Position 90 Grad
VAR_PROCESS = VAR(90)

-------------------------------------------------------------------------------
```

1. Der gelesene Wert der Variablen für den Winkel der Schranke ist immer der zuletzt geschriebene.
2. Die gelesenen Werte bewegen sich im Zahlenraum *[0..90]*.

Aufgabe 2: Spezifikation des Zuges

```
-------------------------------------------------------------------------
-- Prozess für den Zug
-------------------------------------------------------------------------

ZUG = Wait 500;                       -- 500 Zeiteinheiten warten
      (|~| d : {0..100} @ Wait d);    -- internal choice: nichtdeterministisch
                                      -- zwischen 0 und 100 Zeiteinheiten warten
      sig_nah ->                      -- hallo, Zug kommt
      Wait 100;
      nah ->                          -- kurz vom Bahnübergang
      Wait 10;
      da ->                           -- am Bahnübergang
      Wait 10;
      weg ->                          -- kurz nach dem Bahnübergang
      Wait 10;
      sig_weg ->                      -- hallo, wir sind weg
      ZUG                             -- und wieder von vorn

-------------------------------------------------------------------------
```

1. Zwischen dem Signal *sig_nah* und dem Signal *sig_weg* vergehen 130 Zeiteinheiten.
2. Zwischem dem Signal *sig_weg* und dem Signal *sig_nah* liegen mindestens 500 Zeiteinheiten, höchstens aber 600 Zeiteinheiten.
3. Die Signale *sig_nah* und *sig_weg* treten immer abwechselnd auf.
4. Die Signale *nah*, *da* und *weg* treten immer nach dem Signal *sig_nah* in genau dieser Reihenfolge auf, darauf folgt das Signal *sig_weg*.

Aufgabe 3: Spezifikation des Controllers

```
-------------------------------------------------------------------------
-- Prozess für den Controller
-------------------------------------------------------------------------

CONTROLLER = sig_nah ->               -- Signal nah kommt
             zu ->                    -- dann Schranken schließen
             sig_weg ->               -- Signal weg kommt
             auf ->                   -- dann Schranken öffnen
             CONTROLLER

-------------------------------------------------------------------------
```

1. Auf jedes Signal *sig_nah* folgt ein Signal *zu*.
2. Auf jedes Signal *sig_weg* folgt ein Signal *auf*.

Aufgabe 4: Spezifikation der Schranke

```
-------------------------------------------------------------------------------
-- Prozess für den Bahnübergang
-------------------------------------------------------------------------------

BAHNUEBERGANG = HEBEN                                       -- der Anfang

SENKEN = lese_position?pos ->                               -- aktuelle Position
         (pos == 0) & auf -> HEBEN                          -- wenn 0, auf auf-
                                                            -- Signal warten und
                                                            -- in den Hebeprozess
                                                            -- übergehen
        []
         (pos > 0)  & Wait 10;                              -- wenn noch nicht
                                                            -- unten, weiter
                      schreibe_position.(pos - 10) ->       -- senken, Position
                      SENKEN                                -- neu setzen

HEBEN = lese_position?pos ->                                -- aktuelle Position
         (pos == 90) & zu -> SENKEN                         -- wenn 90 Grad, auf
                                                            -- zu-Signal warten,
                                                            -- in den Senkprozess
                                                            -- übergehen
        []
         (pos < 90)  & Wait 10;                             -- wenn weniger als 90
                       schreibe_position.(pos + 10) ->      -- weiter heben, neue
                       HEBEN                                -- Position setzen
        []
         zu ->                                              -- zu-Signal kommt
         schreibe_position.(pos + 10) ->                    -- Position schreiben
         SENKEN                                             -- wegen Sicherheits-
                                                            -- bedingung, in den
                                                            -- Senkprozess über-
                                                            -- gehen

-------------------------------------------------------------------------------
```

1. Die Schranke wird nur gesenkt, wenn der aktuelle Winkel größer als 0 Grad ist.
2. Die Schranke wird nur gehoben, wenn der aktuelle Winkel kleiner als 90 Grad ist.
3. Das Senken der Schranke wird nach dem Signal *zu* begonnen und endet mit der Position 0 Grad.
4. Das Heben der Schranke wird nach dem Signal *auf* begonnen, beginnt immer bei der Position 0 Grad und endet immer bei der Position 90 Grad.
5. Das Senken der Schranke benötigt minimal 10 Zeiteinheiten und maximal 90 Zeiteinheiten.
6. Das Heben der Schranke benötigt 90 Zeiteinheiten, sofern die Schranke vollständig geöffnet wird.
7. Ist das Heben und Senken der Schranke strikt alternierend? Begründet eure Aussage.
8. Könnt ihr Aussagen darüber machen, wann die Schranke nach dem Erreichen der Position 0 Grad das nächste Mal die 0 Grad erreicht (hier gilt der in der globalen Variablen gespeicherte Wert)? Wenn ja, spezifiziert sie formal. Wenn nein, begründet eure Aussage.
9. Könnt ihr Aussagen darüber machen, wann die Schranke nach dem Erreichen der Position 90 Grad das nächste Mal die 90 Grad erreicht (hier gilt der in der globalen Variablen gespeicherte Wert)? Wenn ja, spezifiziert sie formal. Wenn nein, begründet eure Aussage.

Aufgabe 5: Spezifikation des Gesamtsystems Bahnübergang

```
------------------------------------------------------------------------------
-- Gesamtsystem aus Variablenprozess, Zug, Bahnübergang und Controller
------------------------------------------------------------------------------

-- Variablenprozess synchronisiert sich über alle schreibe_position- und
-- lese_position-Events mit dem Bahnübergang

-- Zug synchronisiert sich über sig_nah- und sig_weg-Events mit dem
-- Controller (bzw. dem Prozess, in dem der Controller läuft)

-- Bahnübergang synchronisiert sich über alle auf- und zu-Events mit dem
-- Controller (bzw. dem Prozess, in dem der Controller läuft)

SYSTEM = ((VAR_PROCESS
          [| {| schreibe_position, lese_position |} |]
          BAHNUEBERGANG)
          [| {auf, zu} |]
          CONTROLLER)
          [| {sig_nah, sig_weg} |]
          ZUG

------------------------------------------------------------------------------
```

1. Der Zug überquert den Bahnübergang immer bei geschlossener Schranke. Formalisiert diese Aussage und begründet sie auf Basis der Zusicherungen der Teilaufgaben 1 bis 4.