Prof. Bernd Finkbeiner, Ph.D.
Dipl.-Inf. Rüdiger Ehlers
Dipl.-Inf. Hans-Jörg Peter
Michael Gerke, B.Sc.

Reactive Systems Group
Saarland University
Due: Thursday, 20th November 2008
before the lecture

Embedded Systems 08/09 – Problem Set 4

Problem 1 (Stateflow: Second Tutorial)

To prepare for problem 2 on this assignment and for the forthcoming project, work through the second Stateflow tutorial:

`http://react.cs.uni-sb.de/~peter/sf_tutorial2.pdf`

Problem 2 (Stateflow: Measuring Frequencies) (30 pts.)

Measuring frequencies is one of the most fundamental operations in embedded control, as a variety of physical quantities is in practice measured via measuring a frequency related to the quantity. For example, speeds are often measured by measuring the frequency of a signal originating from a rotation sensor. Basically, there are two methods for measuring frequencies (mixtures and more advanced schemes are obviously possible) of digital signals:

1. to count the number of edges over a time window of fixed length;
2. to measure the time lack between two (or more generally n, with $n \geq 2$) successive edges.

These two methods have their specific pros and cons. In implementations polling their inputs at regular time intervals those two techniques do, in particular, yield measurements which reach their maximum imprecision at different ends of the frequency scale.

Exercise: Download the following model:

`http://react.cs.uni-sb.de/~peter/frequencycount.mdl`

This model contains

- a frequency generator that can be set to any frequency between 1Hz and 500Hz by moving a slider called "`Frequency`" (double-left-click on the block named "`Frequency`" to gain access to the slider);
- an empty Stateflow state chart called "`Frequency counter`", which has
  - an output to Simulink called "`Measurement`", via which the chart shall deliver the measured value for the frequency,
  - an input from Simulink called "`Signal`" that feeds the signal from the frequency generator into the chart,

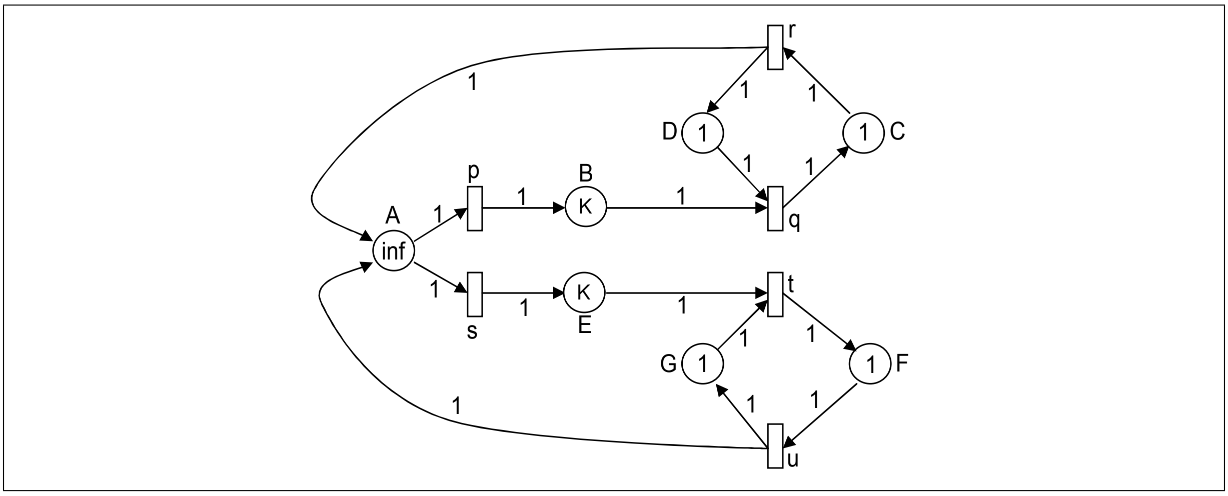

Figure 1: An example Petri net.

  - a trigger input "clk" driving the chart via a positively edge-triggered signal of 1kHz frequency (i.e., the chart performs a step every ms);

- an oscilloscope "Freq" which displays both the actual and the measured (i.e., output by the Stateflow chart) frequency.

Your task is now to

1. implement both[1] forms of frequency measurement by filling in the Stateflow chart. Whenever it is appropriate use "temporal logic events" instead of e.g. self loops. (10+10 pts.)
2. briefly discuss the pros and cons of the two frequency measurement techniques. (10 pts.)

Please submit your MDL files from subproblem 1 to `es08@react.cs.uni-sb.de`. Write your solution to subproblem 2 in your paper submission.

Problem 3 (Petri Net Computer) (35 pts.)

Build a weak petri net computer that takes an integer number n as input and computes 2^n as its output. Please represent it graphically.

Problem 4 (Petri Net Analysis) (35+15 pts.)

Consider the Petri net that is given in Figure 1. It consists of places A, B, C, D, E, F, and G, and transitions p, q, r, s, t, and u.

1. Compute the corresponding incidence matrix. (15 pts.)
2. Compute all invariants. (20 pts.)
3. (*Bonus question*) Prove that there are no further invariants which are not linear combinations of the invariants that you computed in subproblem 2. (15 pts.)

[1] i.e., submit two models, one for each technique.