# Annex 1 - Philosophers Dinner Petri Net Simulation

This laboratory assignment aims at studying Discrete Event Systems (DESs) in the aspects of modeling, analysis of properties and synthesis. Synthesis will be based on a recent methodology in the framework of supervised control. This assignment further develops the previous assignment in the keyboard reading component by introducing fault handling mechanisms.

The tools to be used in this work are MATLAB, including a supervisor design toolbox, and a Petri Net editor. In the last part of the work the Schneider PLCs will be used once more to validate the proposed methodologies.

**DES modeling:**

Using as a base guideline the work done in the first laboratory assignment, it is now desired to model the process of reading one keyboard key as a DES. In particular is supposed to develop a Petri Net that describes the events and the state evolution of the key reading system. Complementing the formal defxinition of the Petri Net, it is desired also to obtain the corresponding incidence matrix.

Note: In order to obtain a Petri Net as simple as possible (less than 20 places), the rejection of multiple keys (first laboratory, part B) should not be included now. The rejection of multiple keys will be subject of formal analysis in part C of this assignment.

Note2: See in the course SVN tools helping this part of the assignment, namely the graphical freeware editor "PIPE2" which allows creating models to import with the Matlab toolbox "tpn5" (use the most recent version made available in the API course). See also Matlab functions simulating Petri nets, in particular the "5 Philosophers" demo and a template for simulating the keyboard "lab2\_sim\_kb\_v6.zip". All these tools are installed in the laboratory computers and can be used at home with SVN:

<svn://svn.isr.tecnico.ulisboa.pt/dccal/projects/lsdc4_api>

See more details in the course webpage.

Please find in the SVN of the course the "5 philosophers dinner" simulation folder. Locate and run in Matlab the file pdinner\_tst.m. Note that you need also the rdp.m function found in the tpn5 toolbox which is also in the SVN - see the SVN root file \_readme.txt for details.

The pdinner\_tst.m demonstration script loads a Petri net model shown in figure1 and built using PMEDIT. The Petri net model is read using the referred rdp.m function.

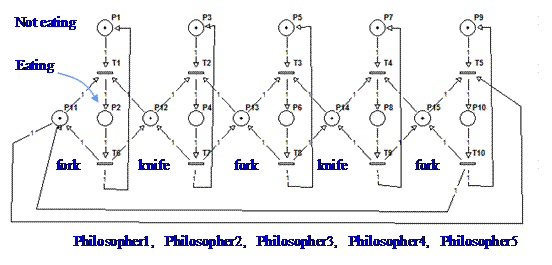


Figure 1: Petri net model representing the 5 philosophers dinner problem.

The default simulation has the transitions scheduled, driven by a time table. In order to run a simulation where the transitions are driven randomly, you can edit the file PN\_tfire.m and replace the line:

qk= pdinner\_IO(act, t);

by the line:

qk= round(rand(1,5)); qk= [qk not(qk)];

Note that only transitions 1 till 5 are computed randomly. Transitions 6 till 10 are defined simply as the negation of the transitions 1 till 5.

To have more information on the Petri net simulator and the specific simulation of the "5 philosophers dinner" see:

<http://users.isr.ist.utl.pt/~jag/course_utils/pn_sim/PN_sim.html>

# Annex 2 - Convert a Petri net to a PLC program

In this annex is detailed the testing of a converter of a Petri net to a PLC program. The converter is written and runs in Matlab. The converter and the demonstration example are included in the SVN of the course.

Please find in the SVN of the course the folder ./tst1\_blink\_turn\_on\_off/ and run tst1\_blink\_on\_off.m . Note that the compiler is in folder ./pn\_to\_plc\_compiler/ which is going to be asked you the first time you run de demonstration.

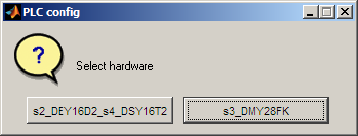


Figure 1: Selection of the hardware configuration.

In the first run of tst1\_blink\_on\_off.m it is asked the hardware configuration. Two configurations are considered, namely the "s2\_DEY16D2\_s4\_DSY16T2" and the "s3\_DMY28FK" (see figure 1). Configuration "s2\_DEY16D2\_s4\_DSY16T2" means the PLC has two modules, at slots two and four, and the modules are named "DEY16D2" and "DSY16T2". Configuration "s3\_DMY28FK" means the PLC has only the module "DMY28FK" mounted at slot three. In case you want to revisit this configuration you can either restart Matlab or run the command clear global.

After running the demonstration, you obtain file tst1\_mk\_program\_res.txt containing structured text code that you can copy into a Unity project. Within the Unity project you need to do declare some additional variables like the timer names, timing values and flags. Please see more details in the text file ./tst1\_blink\_turn\_on\_off/\_readme.txt .

To have more information on the Petri net to PLC code converter see:

<http://users.isr.ist.utl.pt/~jag/course_utils/pn_to_plc/pn_to_plc.html>