Problem 3 - Analysis of a controller for an automation system with N cascaded cells (with N = 3).

First evaluation work:

Execution plan (suggested):

14 October: Parts I to IV; 21 October: Parts V to VII; 28 October: Parts VIII to X.

Brief description:

Consider a controller for an automation system with N cascaded cells. Each cell is composed of a machining center or robot and a conveyor belt. The conveyor belts have sensors for detecting pieces/components or pallets at their inputs and outputs, which provide the IN [1..N] and OUT [1..N] signals, respectively; each conveyor belt behaves as a buffer with capacity to hold one piece/pallet, and when the conveyor belt is stopped with one piece at the exit, a second piece can be received at the entrance. Its movement is controlled by the MOVE [1..N] signals.

Removing pieces/pallets from the system is carried out using an exit conveyor belt, considered as an infinite capacity warehouse and supplying the input signal IN [N + 1].

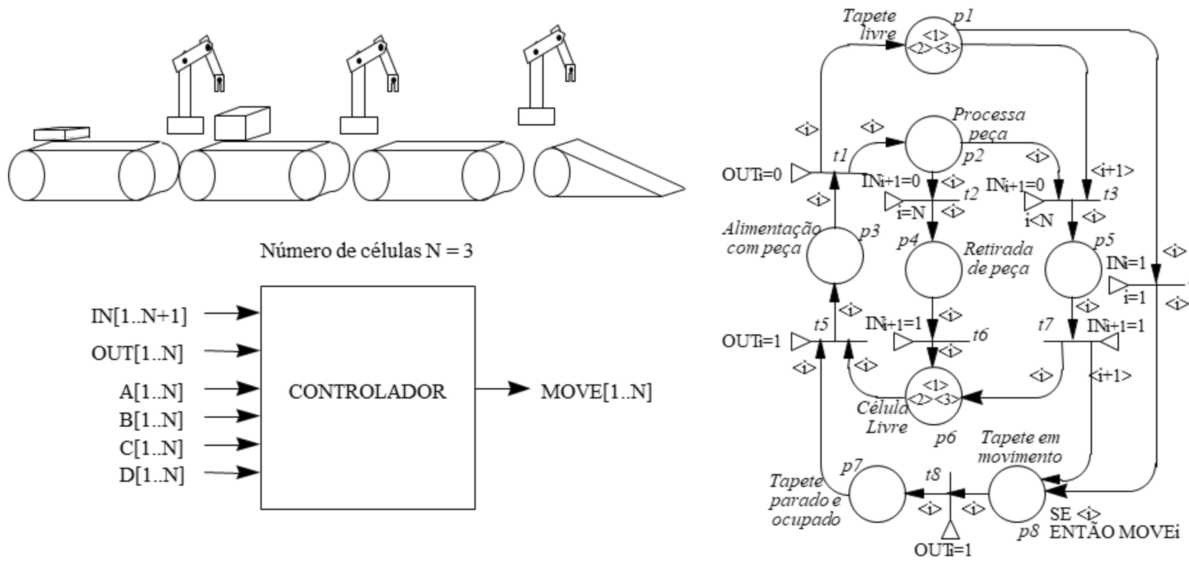


Figure 1 – Model for a production system with several conveyors belts.

Figure 1 shows the simplified characterization of the system as well as a high-level Coloured Petri net model for the control of the conveyor belts movement.

Due to the use of compactness characteristics of the Coloured Petri net, the model shown in Figure 1 is easily changed to represent the control of systems with N cells, by simply changing the initial marking of places p1 and p6, using in each place as many tokens as there are the number of cells of the line. The "regularity" of the model is broken using the transition t4 responsible for modeling the admission of a new piece or pallet into the line (vertical flow shown at the right of the figure).

In order to illustrate the compactness gains associated with the use of Coloured Petri nets models, a low-level RdP that models the same control system is shown in Figure 2. In order not to make the model too "heavy", the identifiers of places and transitions are omitted.

In Figure 2, several regions are identified:

- three regions, represented in layers and associated with the use of the colored tokens <1>, <2> and <3> respectively;
- several vertical regions where relationships can be established between the places and transitions of that vertical region and the equivalent places and transitions used in the model of Figure 1 and referenced at the top of Figure 2.

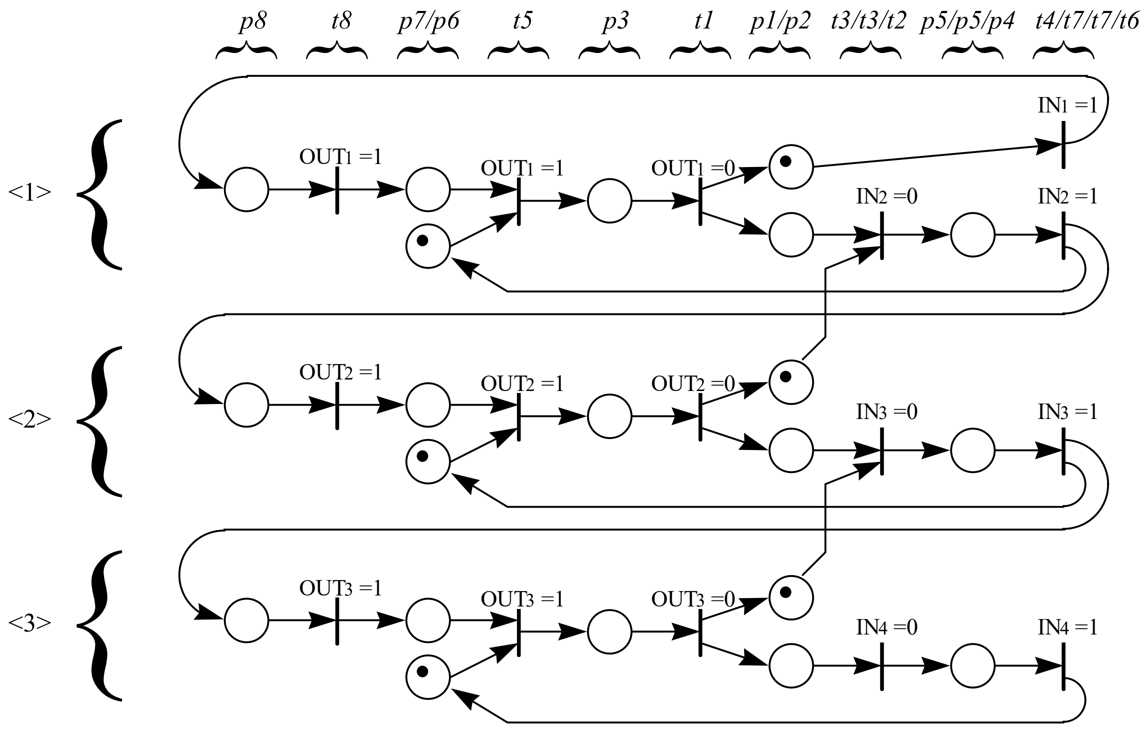


Figure 2 – Low-level model for a production system with several conveyors belts.

From this model, it is possible to identify a sub-model encapsulating the behavior of a cell, as presented in Figure 3.

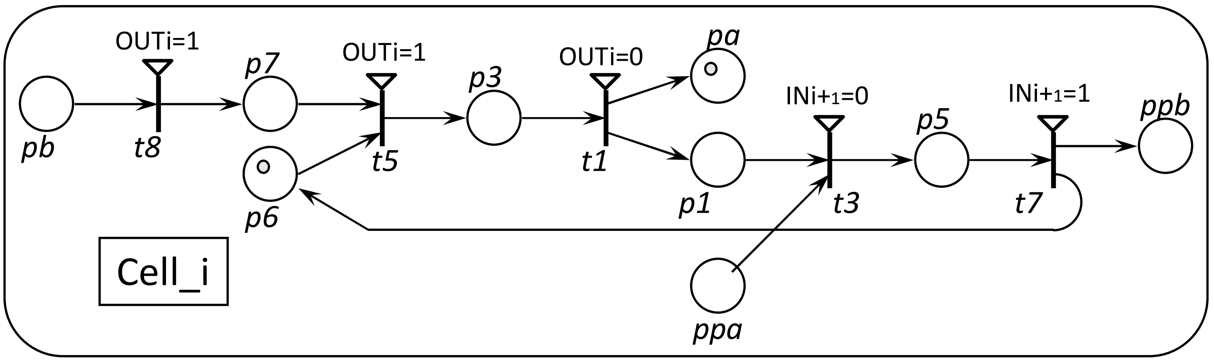


Figure 3 – Low-level model of a cell of the production system with several conveyors belts.

PART I

Goal:

- Use of the IOPT-Tools development environment for modeling the controller of a production system with three conveyor belts, using the net addition operation.

Workplan:

- edit the model presented in Figure 3, including activation of output signal MOVEi associated with the p8/pb so that it is possible to identify situations of a piece on the conveyor belt moving;
- edit two other models as in Figure 4 and Figure 5;

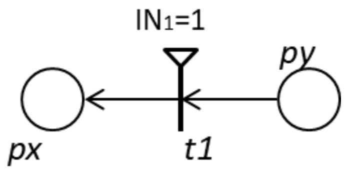


Figure 4 – Low-level model for the initial feeding of the system.

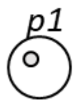


Figure 5 – Low-level model allowing system's output.

- when editing the model using the IOPT Editor use the following options to build the whole system model using the net addition considering three steps:

a) use the "merge nets" button to assure a disjoint addition of the model of Figure 4, three instances of the model of Figure 3, and the model of Figure 5 (as in Figure 6, resulting Figure 7);

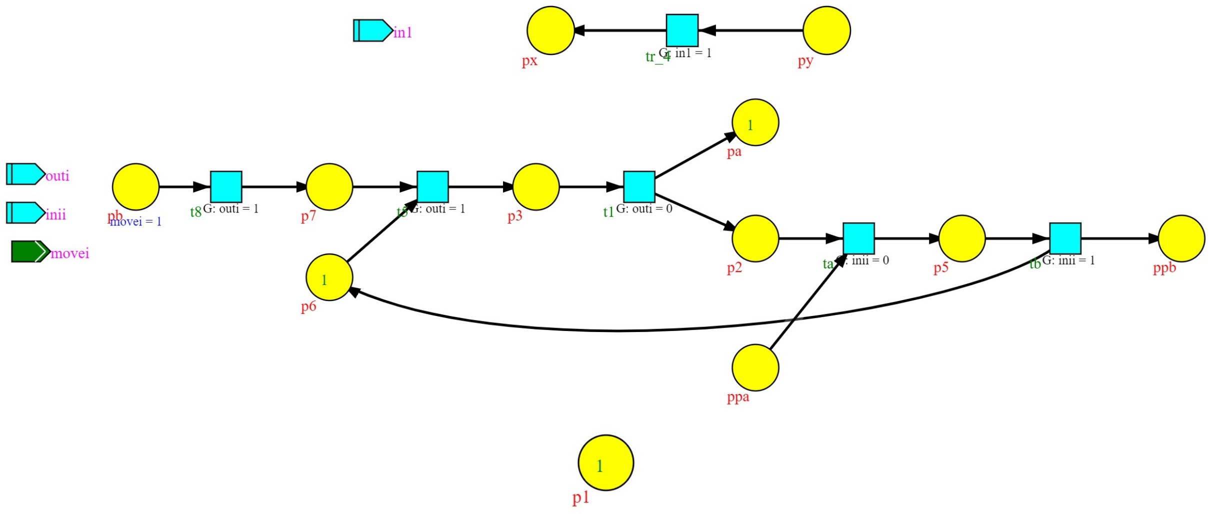


Figure 6 – Sub-models to compose using the net addition operation.

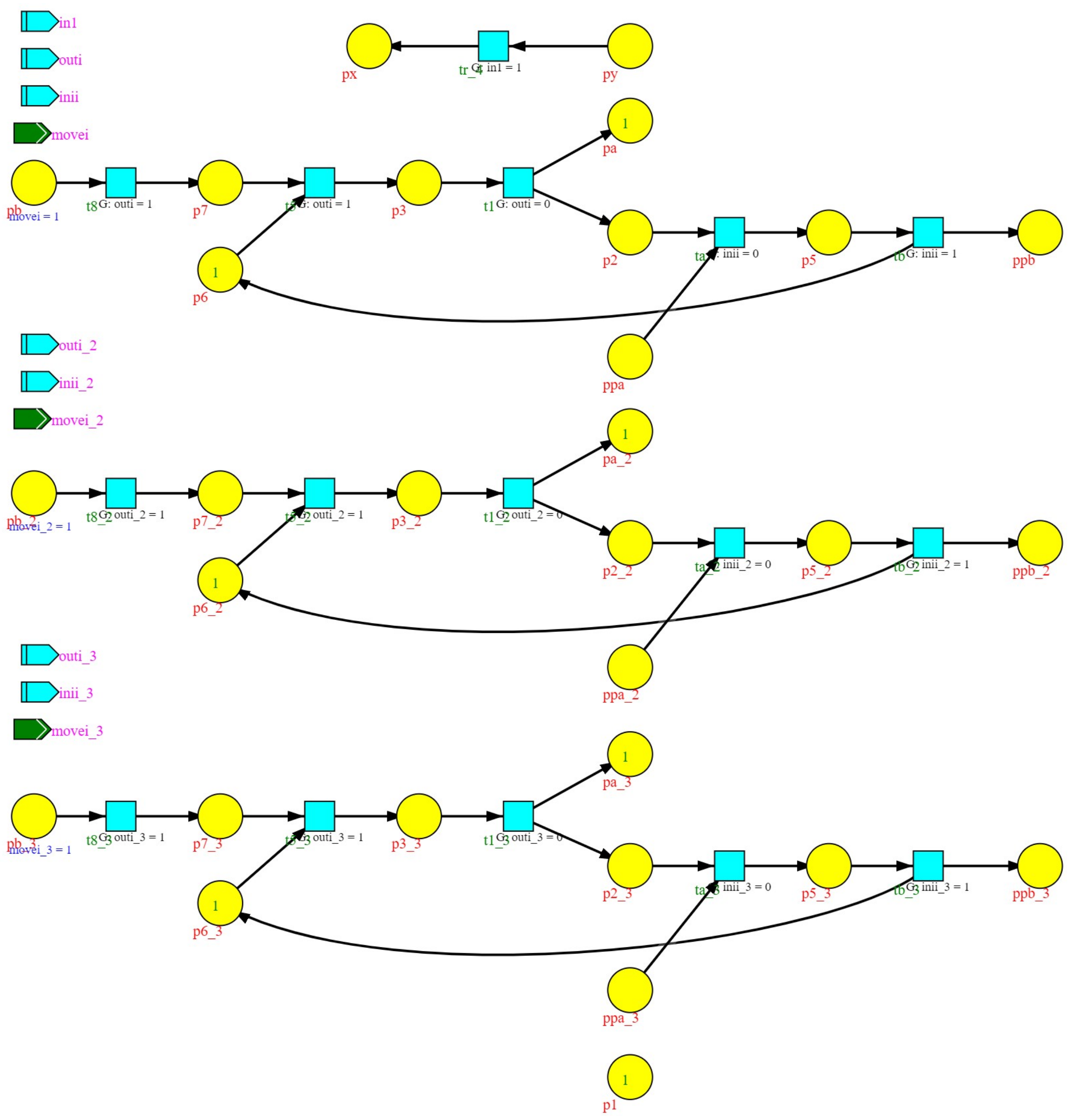
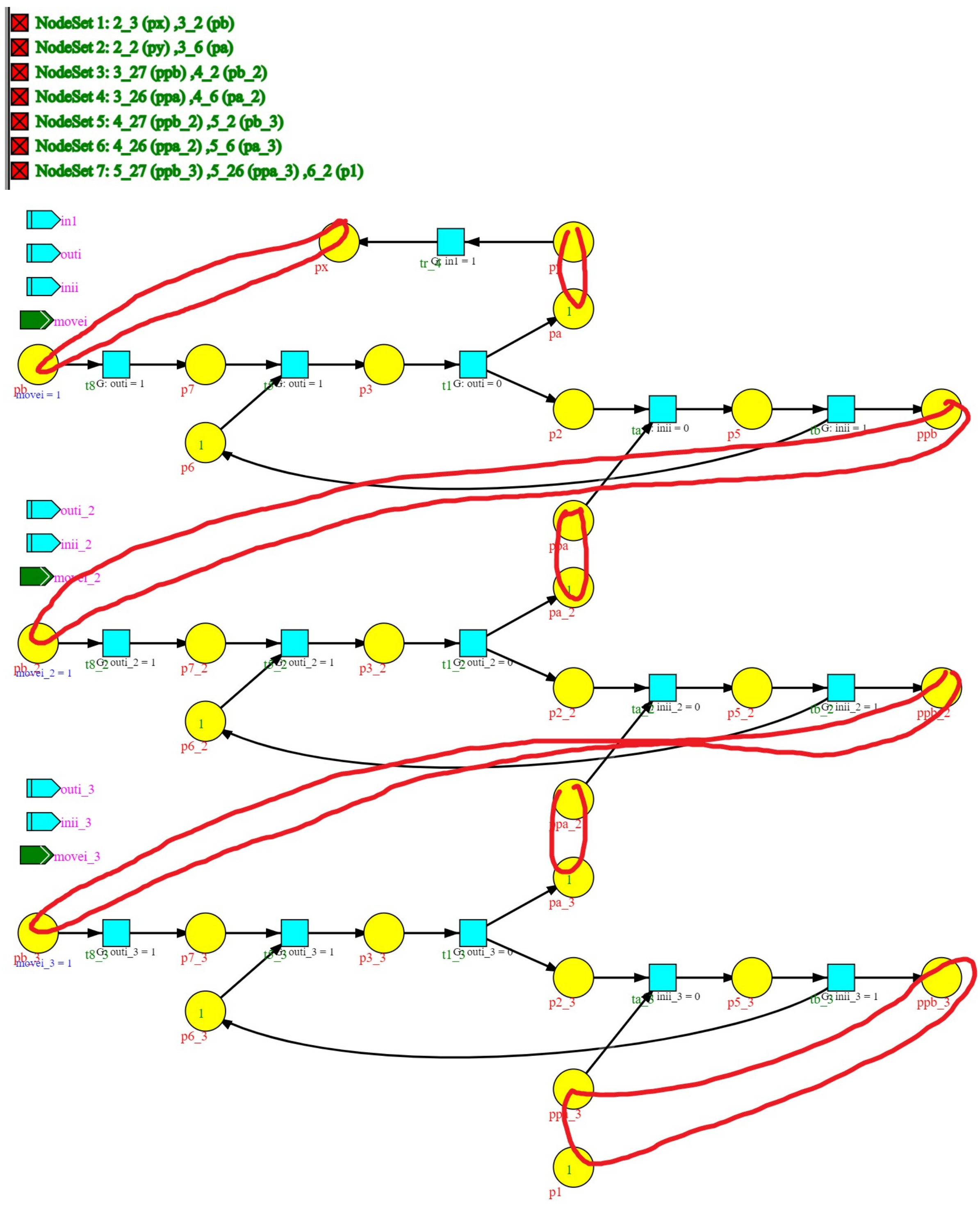


Figure 7 – After disjoint addition of the sub-models.

b) using the "define node set" button, define the following seven node sets to be used as fusion sets: {px,pb}, {py,pa}, {ppb,pb_2}, {ppa,pa_2}, {ppb_2,pb_3}, {ppa_2,pa_3}, {ppb_3,ppa_3,p1} (as in Figure 8);



☒ NodeSet 1: 2_3 (px) ,3_2 (pb)
☒ NodeSet 2: 2_2 (py) ,3_6 (pa)
☒ NodeSet 3: 3_27 (ppb) ,4_2 (pb_2)
☒ NodeSet 4: 3_26 (ppa) ,4_6 (pa_2)
☒ NodeSet 5: 4_27 (ppb_2) ,5_2 (pb_3)
☒ NodeSet 6: 4_26 (ppa_2) ,5_6 (pa_3)
☒ NodeSet 7: 5_27 (ppb_3) ,5_26 (ppa_3) ,6_2 (p1)

Figure 8 – Fusion sets to be used by the net addition operation.

c) using the "net addition" button, obtain the complete behavioral model of the controller (as in Figure 9).

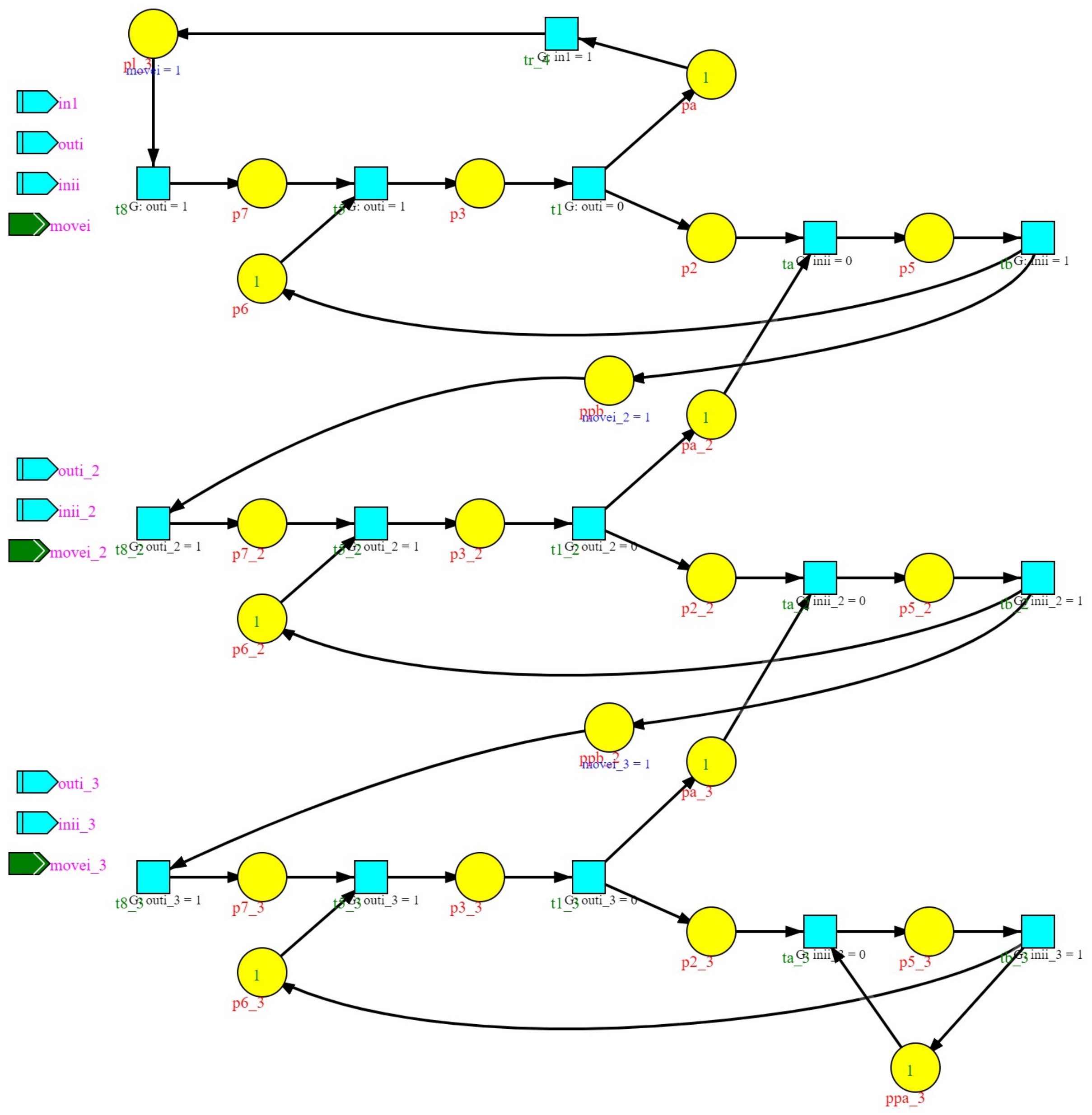


Figure 9 – IOPT model obtained using IOPT-Tools framework.

PART II

Goal:

- Use of the IOPT-Tools development environment for analyzing the model of the controller of the production system.

Workplan:

- starting with the model obtained in Part I, edit the model in order to assure that output signals are included and to change some node labels (if needed);

- use the simulation capabilities of IOPT-Tools framework to validate the model; consider to use both the token-player simulator, as well as the timing simulator visualization;

- use the option to connect to the framework HIPPO and get some additional information about your model, namely how many place invariants are present; (this connection is currently not available, as it is under maintenance)
- to use the facilities for generation of state space associated with a model (Button "Generate State Space" in IOPT-Tools framework);
- analyzing the state space generated through the use of the Query Editor and its analysis of the results produced through Query Results; Examples of questions: How many tokens (representing parts or pallets) will it be possible to have on each belt? Is it possible to have all three conveyor belts moving simultaneously?

PART III

Goal: Deploy in a FPGA a centralized controller of the automation system with 3 cascaded cells**.**

Description:

Taking the result of modeling activities performed at Part I as a starting point, it is intended to deploy implementation code into an FPGA and make the experimentation around its execution.

Workplan:

- using the model obtained at Part I, get the VHDL code from IOPT-Tools and deploy it into the selected FPGA.
- compare experimentation results from execution with the results of the simulations performed in Part II.

PART IV

Goal: Deploy into an Arduino a centralized controller of the automation system with 3 cascaded cells**.**

Description:

Taking the result of modeling activities performed at Part I as a starting point, it is intended to deploy implementation code into an Arduino and make the experimentation around its execution.

Workplan:

- using the model obtained at Part I, get the C code from IOPT-Tools and deploy it into the selected Arduino.
- configure the FPGA board in order to use its input switches and leds to be used in conjunction with the Arduino board.
- compare experimentation results from execution with the results of the simulations performed in Part II.

PART V

Goal:

- Analyze a controller based on the partition of an IOPT model obtained in Part I, using the IOPT-Tools development environment for IOPT Petri nets.

Description:

Taking as a starting point the controller for automation system obtained in Part I, it is intended to analyze and implement a distributed controller resulting from the initial model partition into several concurrent components (considering the global synchronism paradigm at beginning, and globally asynchronous locally synchronous paradigm at the end).

Workplan:

- having the goal to generate a set of components to be used for concurrent/parallel execution, as shown in Figure 10, identify an appropriate cutting set;

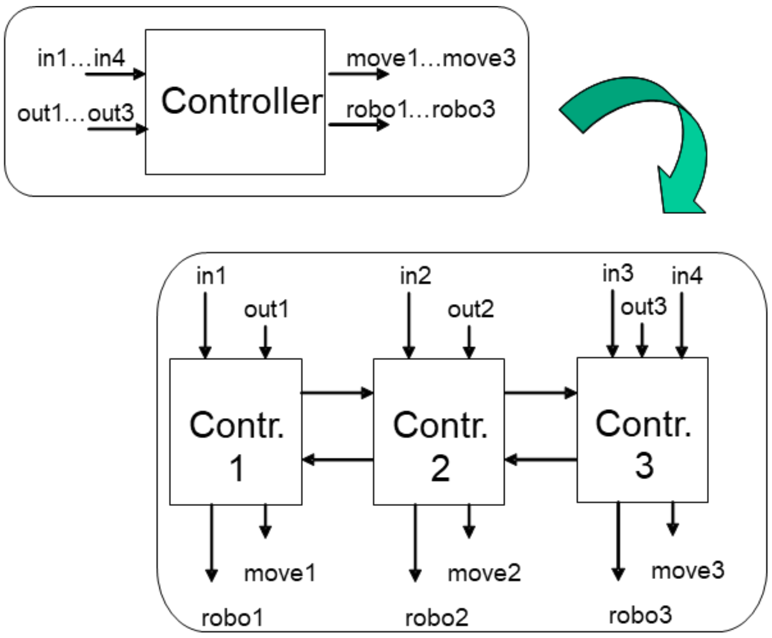


Figure 10 – Decomposition of the global controller into several interconnected local controllers.

- Using the IOPT-Tools editor plug-in SPLIT, considering one of the referred cutting sets and assuring that the “Comments” attribute is filled with the “id” of nodes to be in the same final sub-net (note: instruction for SPLIT operation should be read), decompose the initial model considering the usage of synchronous channels (based on the synchronous execution paradigm); after some editing of the resulting model, a model similar to Figure 11 should be obtained.

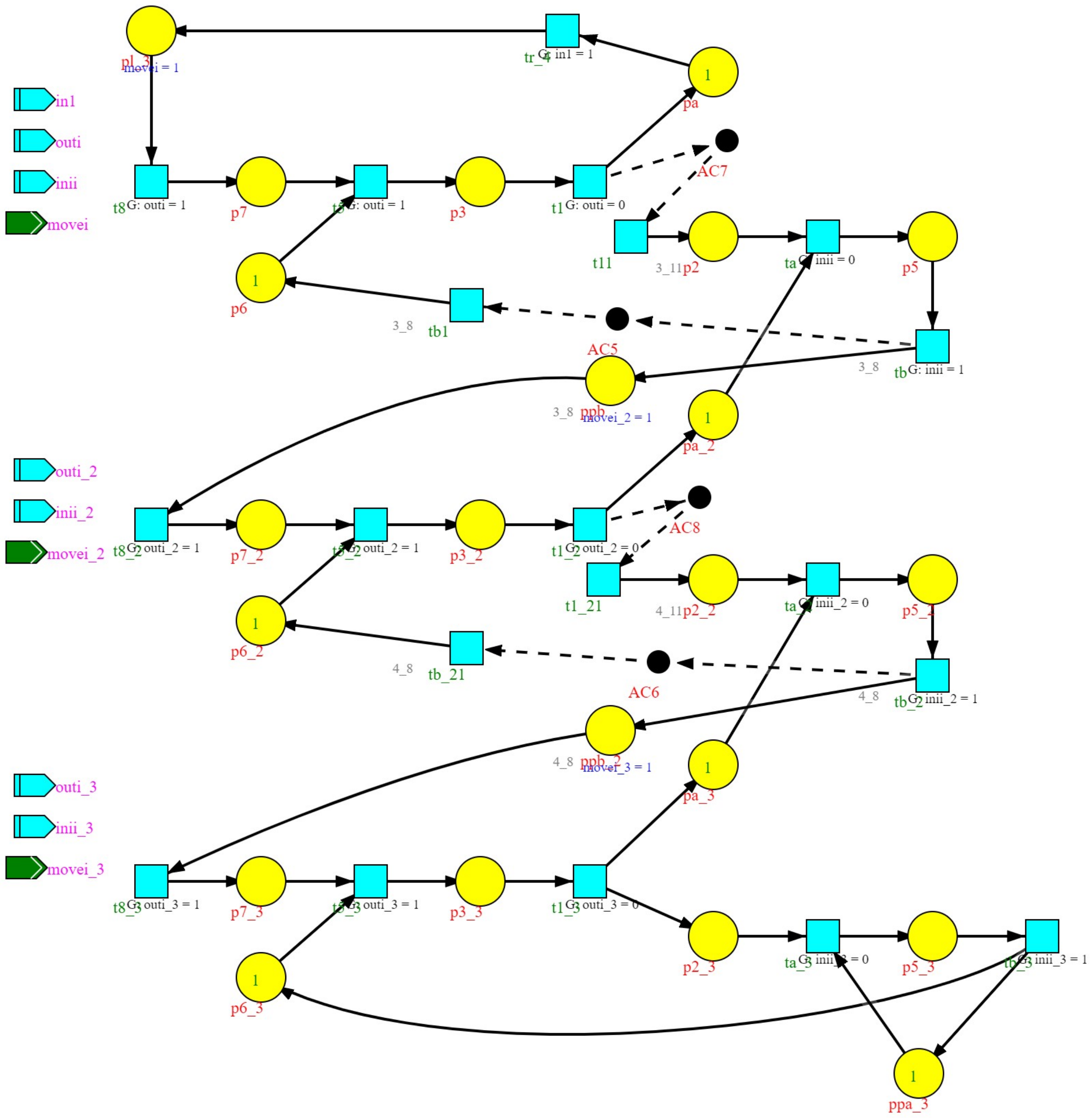


Figure 11 – Decomposition of the model using synchronous channels.

- use the simulation capabilities of IOPT-Tools framework to validate the model; consider to use both the token-player simulator, as well as the timing simulator visualization.
- in order to prepare a distributed execution (and move away from the synchronous paradigm in order to adopt the globally asynchronous locally synchronous paradigm), change the attribute of the communication nodes (AC5, AC6, AC7, and AC8 in Figure 11) to Asynchronous set and introduce the attribute of Time Domain to all nodes of the model, where each Time Domain will be associated with one different physical controller (a model “similar” to the one presented at Figure 12 should be obtained).

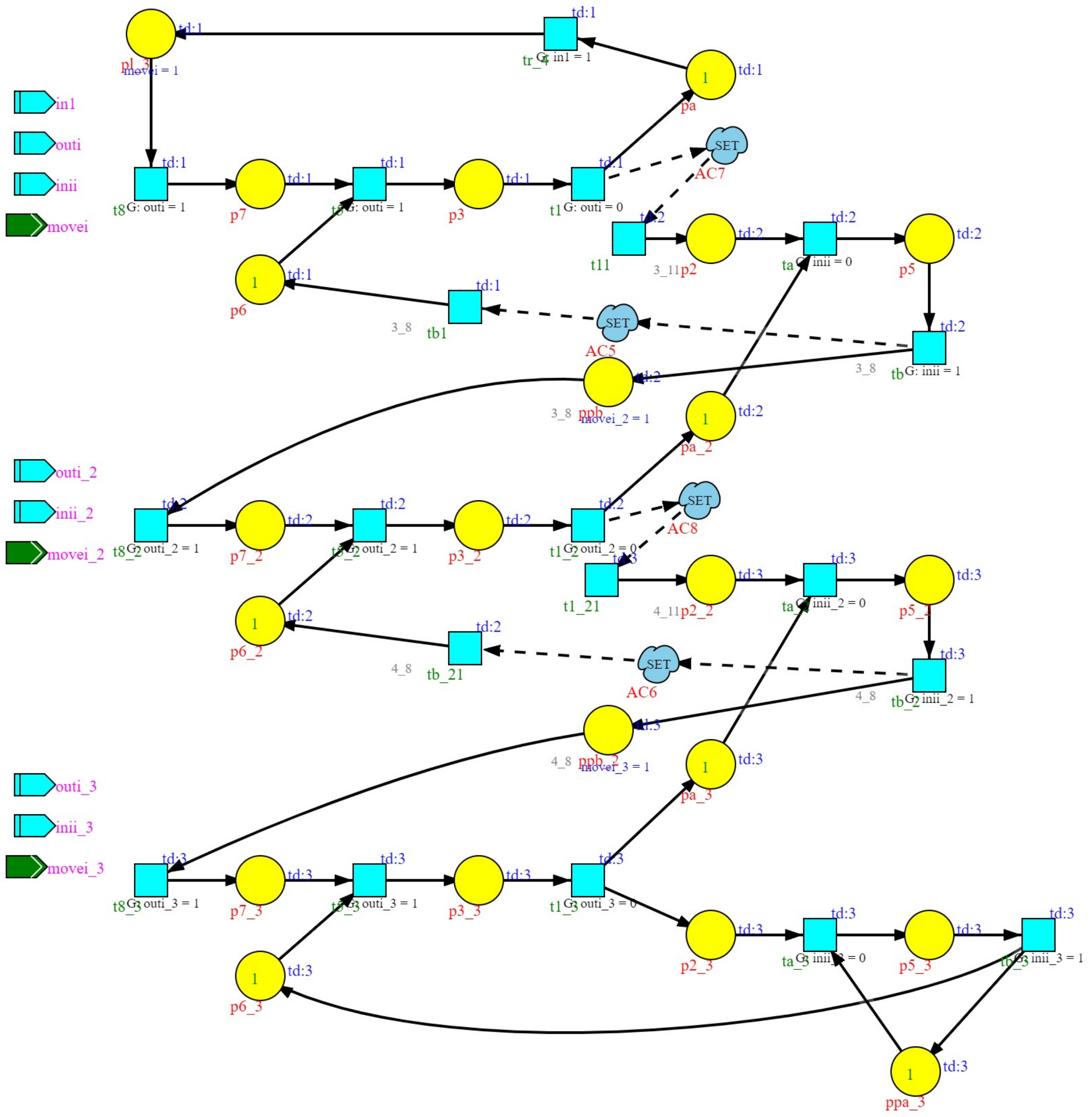


Figure 12 - Decomposition of the model using asynchronous channels.

- use the simulation capabilities of IOPT-Tools framework to validate the model; consider to use both the token-player simulator, as well as the timing simulator visualization.
- using the "Decompose GALS" button of the IOPT-Tools framework, generate three separated models for the three controllers (obtaining models similar to those presented in Figures 13, 14, and 15, after deleting signals not used in that controller).

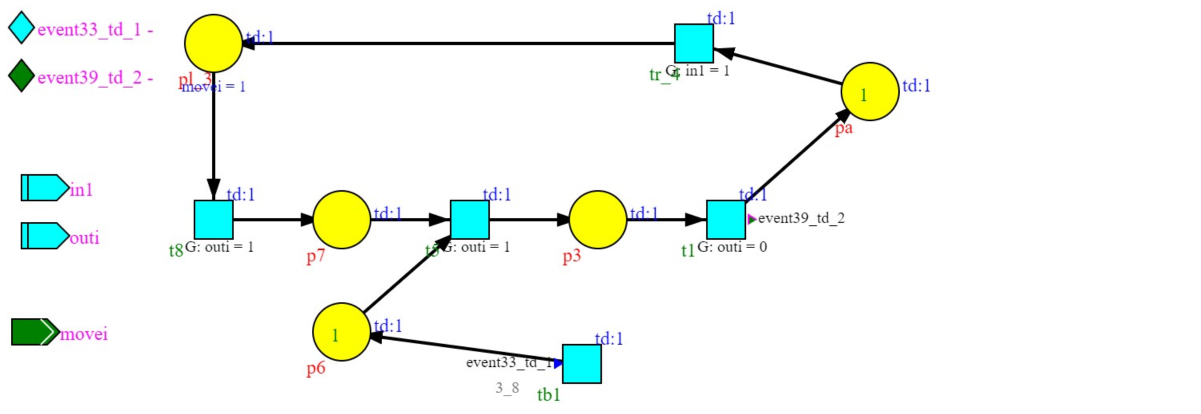


Figure 13 – Model for controller 1.

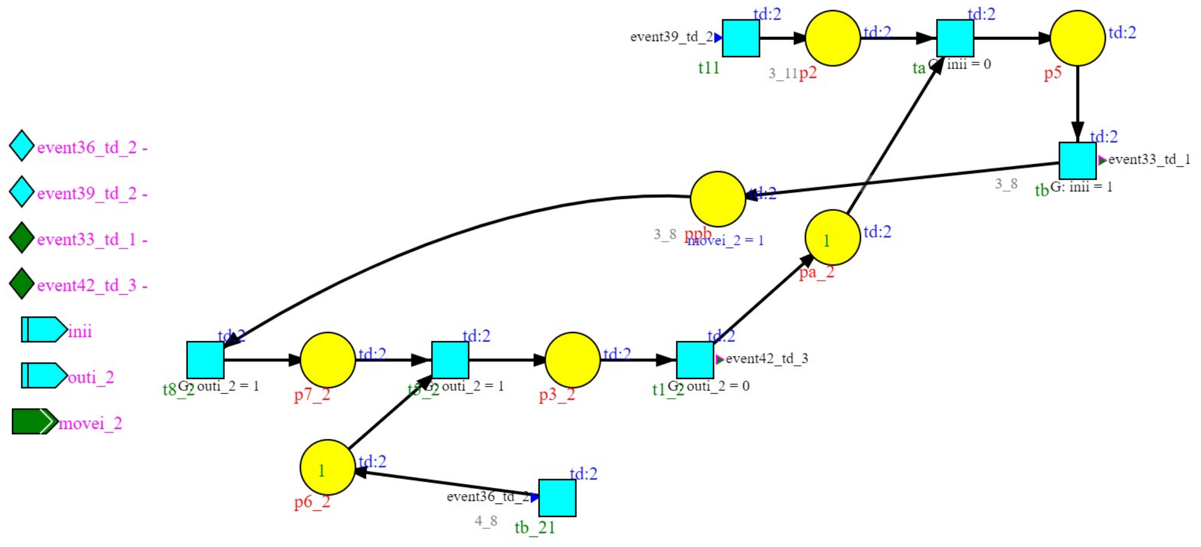


Figure 14 – Model for controller 2.

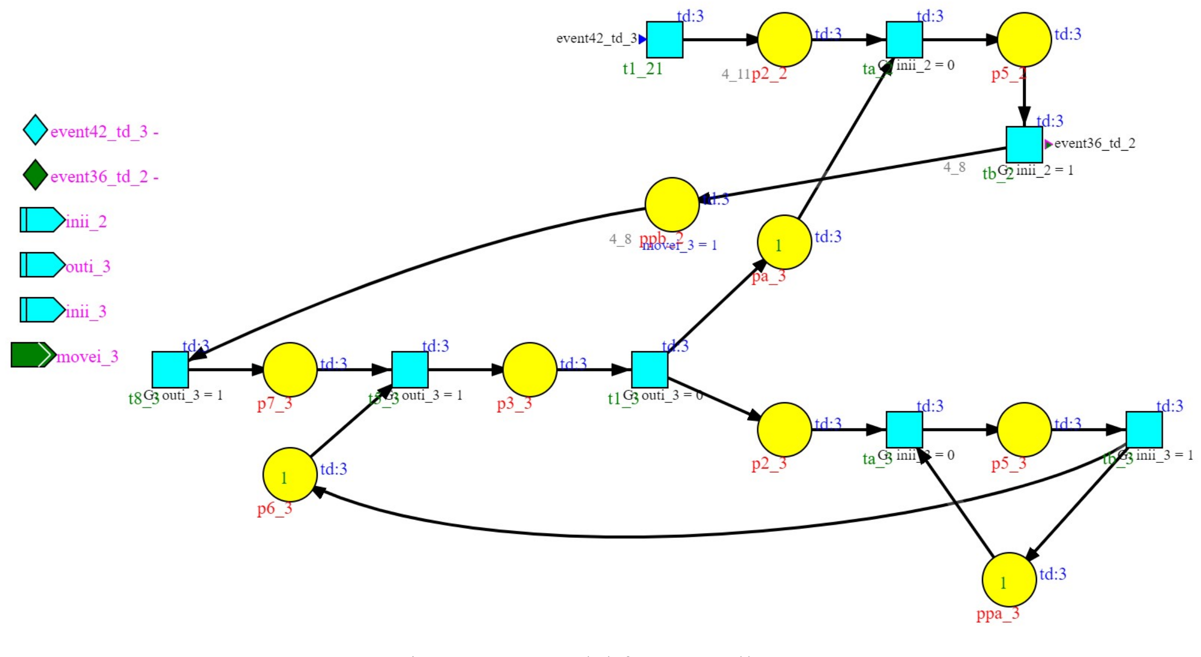


Figure 15 – Model for controller 3.

PART VI

Goal: Deploy in a FPGA a controller of the automation system with 3 cascaded cells composed by three components**.**

Description:

Taking the result of modeling activities performed at the end of Part V as a starting point, where three models were obtained, one per conveyor belt, it is intended to deploy implementation code into an FPGA and make the experimentation around its execution considering a synchronous execution paradigm.

Workplan:

- using the models obtained at the end of Part V (Figures 13, 14, and 15), get the VHDL code from IOPT-Tools for the controllers of each conveyor belt;
- edit a VHDL description interconnecting the three components, considering a global clock for all of them and a direct connection of internal events (between the event generated by one component and associated event received by other component, as a result of the usage of the split operation using asynchronous communication channels);
- deploy the resulting code into the selected FPGA.
- analyze experimentation results.

PART VII

Goal: Deploy in a FPGA a controller of the automation system with 3 cascaded cells composed by three components with non-instantaneous internal communications**.**

Description:

Taking the result of activities performed at the end of Part V as a starting point, where three models were obtained, one per conveyor belt, it is intended to deploy implementation code into an FPGA and make the experimentation around its execution considering that pseudo-random delays are introduced in the communication between components. Note: the pseudo-random delays are multiple of the clock signal period.

Work plan:

- construct a module that produces a pseudo-random delay in its output after receiving a trigger in its input. It is suggested to use a LFSR - Linear feedback shift register with 27 or 28 bits and a decreasing counter that is loaded with the value of the LFSR upon receipt of a new event and which remains at 0 at the end of the countdown; the output event will be generated when the count state is 1 (during a clock cycle). Information of interest about LFSR can be found in the Xilinx XAPP 052 Application Note - "Efficient Shift Registers, LFSR Counters, and Long Pseudo-Random Sequence Generators" available at the moodle area of the course;

- using the VHDL description obtained at the end of Part V, replace the direct connection used for internal events by the pseudo-random communication component previously built introducing a pseudo-random delay;
- deploy the resulting code into the selected FPGA.
- analyze experimentation results.

PART VIII

Goal:

- Change the model produced in Part I in order to allow the presence of several simultaneous pieces in the various conveyor belts (i.e. that the belts behave as buffers of finite capacity, but different from one).

Description:

Starting from the controller for the automation system obtained in Part I, it is intended to introduce the ability of the various belts to carry several pieces simultaneously (maximum 3 pieces), ensuring that whenever a part is present at the end of one belt, it is stopped waiting for its removal by the processing robot.

Workplan:

- as for Part II for each of the following steps:

a) introducing more than one piece at the first belt;

b) adding introducing more than one piece at the second belt;

c) adding introducing more than one piece at the third belt.

- as for Part IV, after including a module to allow visualization of the number of objects present in each conveyor, deploy the resulting code into the selected FPGA considering a centralized implementation.
- analyze experimentation results.

PART IX

Goal:

- After conclusion of Part VIII, follow the same procedures as in Part V and Part VI to implement a distributed deployment for the controller system, still considering a global synchronous execution.

PART X

Goal:

- After conclusion of Part VIII, follow the same procedures as in Part VII to implement a distributed deployment for the controller system, considering non-instantaneous internal communications.