D-extended Petri nets for simulating of digital devices

A. A. Veselov

Tver State Technical University
Tver, 170006, Russia
aveselov@tvcom.ru

Abstruct: The paper presents D-extended Petri nets, a specific variant of Place/transition nets with inhibitor arcs that are structurally restricted in a way that implies safety. This class of nets is shown to be able to represent basic boolean functions as NOT, NAND, NOR and to build elementary flip flops like TT. A notion of time is added in a way, that handles conflicts among enabled transitions by a non-deterministic selection of one transition to fire and disabling others. Enabling times are not memorized, so the next time the discriminated transition becomes enabled again, it must wait the whole duration before it can fire. The paper proposes the use of the presented class of Petri nets in the design of digital circuits.

Key words. Digital circuits, typical structures of models, combinatorial and consequent logic, the models of functional elements of digital devices.

Introduction

Petri Nets [PJ81] were initially proposed for studying the behavior of parallel systems, they are well suited for modeling event driven systems. Petri Nets (PNs) have gained wide popularity, since they were first introduced (by CA Petri 1960), mainly due to their simplicity as well as their expressive and graphical capabilities and a well developed mathematical apparatus for formal descriptions, modeling, verification and analysis of discrete systems [RR98a, W98, RR98b]. Today PN are effectively used for modeling systems from a great variety of human activity domains including, for instance, modeling hardware too [Va90, NP86, SP01]. Classical PNs and some of their modifications, e.g. so-named evaluative nets or E-nets, are used for that because they were especially worked out for solving such problems [M89, G00]. The most significant disadvantages, which limit application and wide use of this PNs in the practice of designing digital devices, are first of all an improper consideration of functional specificsof the modeled systems and ones of the technology used for their construction.

In other words, despite efforts made by many researchers, a certain discrepancy between the concept of a model as a formal object and the concept of a real discrete device still remains. For the first time, this problem considered by D.E. Muller within the framework of the theory of “circuits” that is still relevant. The author assumes, that his leads to the process of studying the modeled object to be divided into two different processes: modeling the object itself and its subsequently interpreting in order to transfer the results obtained onto the modeled object. In such a situation we need to hire a specialist on simulation modeling or to engage a better qualified designer of digital circuit devices who has a high level of expertise both in his own specialty and in modeling of such systems. Organizational and technical difficulties that are related to unifying the processes of modeling and designing continue to hinder the effective use of modeling in implementation of the digital discrete devices.

Attempts to compensate this shortcoming by using more complex specialized PN extensions have not yielded the desired results. The main reason of this likely to be that a more general and consequently a more universal mathematical model much weaker take into account specifics and behavior of objects for a particular subject domain. On the other hand the more specialized models such as E-nets are complicated and have a large amount of additional attributes. This greatly weakens their general analytical capacities.

Within the framework of solving this problem the author purposed a new PN extension [Ve90] that is especially intended for modeling digital circuit devices and more completely takes into account their behavioral specifics. As a result of subsequent research some of the basic notions of this extension were significantly improved, more deeply understood and corrected in the form of a D-net extension [Ve99].

The general concept of building a D-net includes the following basic propositions:

– At most one event may occur in a discrete system at a time. If more than one event at a time does however occur, this is show that an insufficiently small time scale is likely to be chosen.
– State of the entire system is evaluated as a set of states of its subcomponents. For example, the state of a digital device is evaluated as the corresponding set of voltage levels (states) on all input-output pins of its chips.
– As a result of the occurrence of an event it is possible for one and only one component of the system to change its state.

Besides that a new PN extension must to be more adaptive for executing adequate and formal transformations of notions, characteristics and properties of its theoretical apparatus into similar notions and properties of the digital hardware and vice versa. As the binary logic is the most widely used for digital techniques then it is needed some other simple and reliable ways of guarantee of these nets safety to be stipulated.

A new extension would facilitate the development of tools, which would allow the designer to carry out experiments with the object as well as with its model. Moreover, the developer can perform his experiments directly during designing and obtain the results in terms and notions he is accustomed to.

D-nets facilitate the development of tools, which will allow the developer to carry out experiments with circuital image of the object. Moreover, the developer can perform his experiments directly at design time and obtain the results in terms and notions he is accustomed to. In other words, this tool would allow the designer to construct the required circuit, to examine its behavior and to map the results directly onto the digital circuit. Thus, we can move the simulation model to the back stage and model the behavior from there without any explicit need too manipulate the model. This would allow much greater ease in embedding such an instrument for modeling and analysis, for the general development process and also for its separate parts.

A timed non-pure PN (with inhibitor arcs) was chosen as a basis for the construction of D-net. The following are the main reasons for choosing such a base:

- Availability of quite simple facilities for describing the net, that allows to provide the high modeling ability for digital electronic devices (DED) and at the same time to preserve the basic analytical capabilities of theoretical apparatus for general PN.
- Need to consider the effect of non-primitive events (timed delays in transitions) on the behavior of the model and to include the notion of model time into simulating tool.
- The use of inhibitor arcs enables expanding the operational logic of transitions and makes the models more compacted.
- Using self-loops (bi-directional) arcs allows us to extend the capability of describing conditions for enabling and firing transitions. It enables us to increase the accuracy of describing the functioning of the model and improves adequacy of its interpretations in term of modeled objects. Like for forbidden (inhibitor) arcs, allowing arcs enable us to expand the set of available links to account such, which define active and firing conditions and at the same time they don't directly influence on the redistribution of markers.

Formal definition of the D-nets extension

Thereby, a D-net may be formally introduced as following:

DN=(P, T, F, D, M), (1)

where:

P – a finite set of places;

T – a finite set of transitions ($\mathbf{P \cap T = \varnothing}$);

$\mathbf{F=T \times P \rightarrow l=}$\{ in, out, allow, forbid \} – a finite set of arcs (including the inhibitor arc) connecting items from the sets **T** and **P,** where **l** –is type of link;

D=**T** $\rightarrow$ **{** delay **}** denoted by delay – timed delay on the firing of a transition;

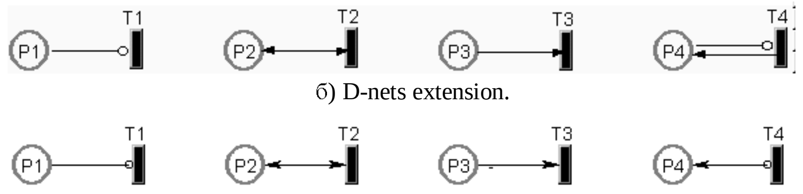
M=P $\rightarrow$**{** 0, 1 **} -** marking.

As it is evident from (1), there are only the following four types of connecting arcs between places and transitions of a D-net:

- input (l=in) – graphically presented as arrows ($\rightarrow$), directed from a place to transition.
- output (l=out) – graphically displayed as arrows with an inhibitor (**o**$\rightarrow$), directed from the transition towards a place.
- allow (l=allow) –is in the form of a double headed arrow ($\leftarrow\rightarrow$) connecting a place and transition.
- forbid (l=ing) – is in the form of a connecting line starting at a place and ending at a transition with an in inhibitor at the end.

As it is evident, from all the possible types of connections between places and transitions the standard output arc (which is not self-looped) is absent. Instead of this a self-loop output arc is used now. This consists of a pair of arcs: an inhibitor arc that ends in a transition, and a single output arc (standard output arc) that leads to a place. As a result a D-net token may be put in place only if it is absent. By nature this restriction is a structural way guaranteeing safety in a D-net. The correspondence between the graphical images of structural components in classical PN and their D-net analogs is presented in fig.1.

a) General Petri nets.



б) D-nets extension.

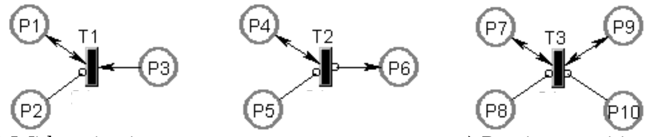
Conditional position (forbid) | Conditional position (allow) | Active position (input) | Active position (output)

Fig.1. Structural components in classic PN and D-nets.

All links in a D-net may be divided into two groups: active (input and output) and conditional links (allowing and forbidding). An active link defines a way of the place's marking change resulting from (putting or extracting) the associated transition firing. Conditional links only define the conditions of the enabling of transitions but they do not influence the marking for the associated places. For further discussion, we will use the following notation: *in(t)* – input place for transition "t", *allow(t)* – set of all its allowing places and *forbid(t)* - forbidding places. As "*t" we will denote all the conditional places linked with the transition, including the input place ($t = in(t) \cup allow(t) \cup forbid(t)$). Using "t*" we will denote the output place for transition "t". The set of all places associated with the transition will be denoted as $nbh(t) = {}^*t \cup t^*$.
In connection with the accepted concept for D-nets, the following restrictions are imposed on links between its nodes:

$$\forall t \in T: |in(t)| + |t^*| <= 1. \qquad (2)$$

This means that any transition of a D-net may have any number of conditional places linked with it and no more than one input or output place. Thus the amount of conditional places are not limited. It should be noted, that this restriction, among other things, simplifies the functioning of the transitions. The basic link structures that are possible in D-nets are presented in fig.2.
In comparison with classical PN, execution rules in D-nets differ. When



a) With active input b) With active output. c) Passive transition.

Fig.2. The basic kind of link structures with transitions.

implementing a mechanism for the net's execution, to account for the influence of time on the net's behavior the Merlin's [BD98] approach is used, under the condition, that dmax=dmin.
For a D-net the condition for an active transition is defined in the following way:

$$\forall p \in allow(t_i) \cup in(t_i): m(p)=1 \wedge \forall p \in ing(t_i) \cup out(t_i): m(p)=0. \qquad (3)$$

In other words, a transition is enabled if there is a marker in each of its allowing and input places and there are no marker from each of its forbidding and output places.
If the transition is enabled, then it may fire. Firing is possible, only if the transition was in a state of activity continuously, for the length of time, defined by the value of delay at "t_i", delay(t_i).
As a result of the transition's firing and depending on the structure of its links, the marker is either extracted from input place or is put into the output place. However, the marking of the places associated with the transition may not change at all. This case is for all linked places each of that is no input or output place for fired transition. In any case, as a result of firing the active transitions the marking of only one of the active places associated with it may change. In general PN, when an active transition fires, markers are extracted from the input places and simultaneously put to output places (in other words, as a result of the transition firing the markings of more than one place change at the same time).
The structural particularities and the execution rules for D-nets make it possible to remove a part of the problem arising in general PN (for example, the problem of introducing a priority scheme for firing transitions where needed). Also importantly D-nets allow more naturally to present, to define, display and more adequately to interpret the behavior of the objects. In this sense, the presented D-net extension ensures a much closer relationship between attributes and properties of a model and its corresponding features, states and behavior of the modeled objects. These modeled objects represented a wide spectrum of electronic and digital devices.

Execution of D-nets

D-nets function in two stages: initialization and execution.
The stage of initialization begins when the net is initially marked. Then, we need to pass through (traverse) each transition with the task of finding all the enabled transitions, and add them to the enabled transition list. After this, the net is ready for further execution.
The execution stage begin by searching among the enabled transitions that have the most short time left to their firing. Such transitions may be a few. The interval of time left to their firing is stored as value of current time step. After this, all this transitions are removed from the enabled list and are moved to firing list. Herewith, for each transition that left into enabled list, the time that left to his firing is reduced on calculated value of current time step. The current value of the modeling time is increased on the same value. As was it already spoken in a general case not one but many transitions may appear in the firing list. In accordance with the accepted concept, if model time coincides for several different events, then it is considered that the real time of occurring for each of them is unique and, consequently, has to different. So the transitions from the firing list are allowed to occur in any order. But time of their operating will be fixed alike.
When a transition is firing, then we change the marking of the active (input or output) place that is associated with it. Then all transitions linked with the active place are checked on fulfillment of their enabling conditions. Herewith the following situations may be arise: either new enabled transition appear or breach the enabling conditions for one or several of the transitions from the enabled or firing lists.

When the first situation arises, then the new enabled transition is added into the enabled list, and we go on processing the next firing transition.
In the second case, the firing transition whose enabling conditions were breached, is forbidden, regardless of membership to list(enabled or firing). Then this transition is removed from enabled or fired list and moved to the list of usual (no enabled and no firing) transitions.
When all of firing transitions are processed, the execution stages described above are repeated. This will continue until fired and the enabled lists not will be rendered empties or the modeling time will reach its given value.

Structural and behavioral particularities of D-nets

Study of the presented the D-net extension and its particularities is presented useful to fulfill with constant taking into account of carrying of the corresponding notions used in D-nets to similar notions for area of the digital hardware. Such accent allows consider the D-net not as just one more new expansion. Instead of this D-net necessary to consider as an extension, which affords a certain new additional kit of attributes and characteristics that could be very useful for user. For example, the using of this kit would allow more simples the implementation of formal mutual transformations of notions and categories pertained to domains of electronic circuits and their models. Consider some from them.
In the same way, as in classical PN, in D-nets the formation of so called isolated places and transitions are possible. That nodes of net which have no links. The similar elemental structural formations do not offer anything interesting for researchers. So let us consider a more complex structures which can be formed in D-nets. In accordance with (2), for D-nets, except of isolated transitions, it is also possible the forming of transitions that have no link with active place, but all links are only with conditional places (fig.2c). The firing of such a transition is only formal in nature and no change takes place in the marking of the associated places. We will call such transitions as *conditionally isolated*. The existing of conditionally isolated transitions does not influence on behavior of D-net and so such transitions may exclude from structure of net. Structures having several transitions with a common place deserve a more careful attention (fig.3).

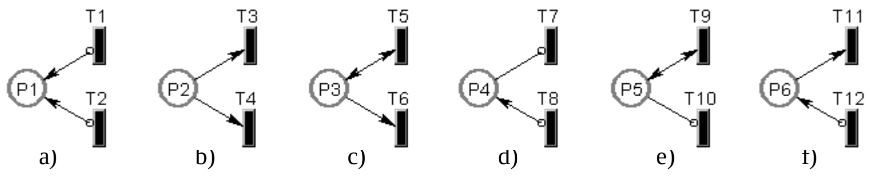


Fig.3. Structures, formed by transitions with a mutual position

The first two structures are formed by transitions with mutually active places (fig.3a,b). If both transitions, for example T1 and T2 (fig.3a), are simultaneously enabled and fire then, in correspondence with the definition for a D-net, they may fire independently from one another and in any order. But the occurrence of any of them immediately breaks the enabling condition of other transition. As a result this transition already cannot fire because it is no longer active. In such situations nothing disagrees the real behavior of the discrete systems. Because, disregard from that

which event is occurred first, the marker in any case will be put into place P1. A similar reasoning can be applied and to the other structure (fig.3b) too. Both this cases are a natural way of solving of the problem related with priorities.
Structures presented in fig.3c and fig.3d also contain two transitions with a common place, which is active for one transition and conditional for the other (or for others). If all transitions in this structure are enabled and simultaneously fired then in connection with execution rules of D-nets the firing of transition with an active link will always break enabling conditions for other transitions (with the conditional link). For structures presented on fig.3e and fig.3f it is just not possible to create conditions under which all of their transitions could be enabled simultaneously. Because, if a marker in the place enables one of the transitions then it automatically mean disabling (forbidding) of the other transition. Analyze of structures presented on fig.3 show that among all possible structures included place can select three typical structures. One of them contains a place with only conditional links (fig.4a). It has got name of conditional place or *control place*. The other contains a place with only active links (fig.4b). It is named as controlled place or *place-indicator*.

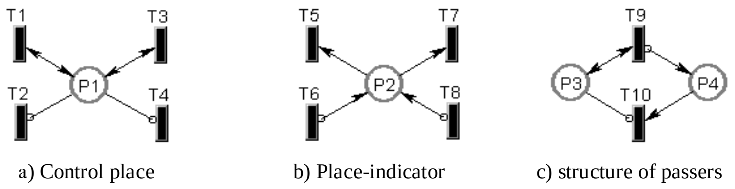


Fig.4. Structure controlling position, position-indicator and passers.

If a D-net has a control place, then it is not possible to change the marking of such place with help the D-net model itself. In other words, such a net cannot put marker into the place or extract it from there. In that case, the state of the marker in this place is defined only by the initial marking, which is defined externally. However, the structure of *control place* is not a useless formation. This structural formation can be considered as a facility, which helps to realize the interactions between nets or between functional components of net. For example, with the help of such a place we could implement a goal-directed action (influence) to the net from external environment or other nets. In other words we can influence the functioning of the net externally. Availability of the *place-indicator* in a D-net means that we can change the state of token in this place but token of place itself can't change token of other places in net. The *place-indicator* can be used as facility of the modeling of output signal to afford information or render influence on external environment or other nets. If the *place-indicator* is not interpreted as such facility then it may be remove from the net.
Except of this, it is also possible to form structures in D-nets that contain two places for which all the associated transitions are mutual (fig.4c). Herewith, one of these places is conditional, and the second is active. It's easy to see that the marker in such a

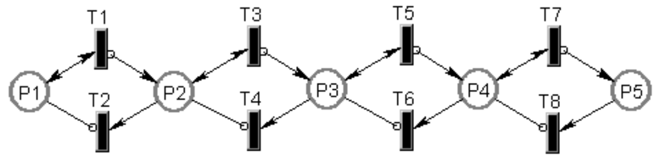
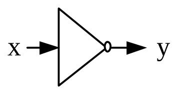
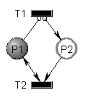
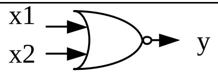
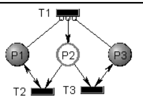
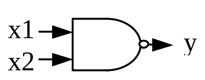
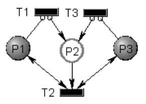


Fig.5. Structural chains of the passer of marker.

structural formation is passed from one place to the other. We will call such structural fragments *structure of passers*. Such structures can form chains as it is shown in fig.5.
As a rule, all modern digital devices are built on basic standard integral microcircuits (IC) of small, middle, large and very-large-scale of integration (VLSI). Each of these microchips can contain one or more functional elements. In spite of large number of different kinds of modern functional devices, the number of base components from that they are built limited by a few units (NO, AND, OR and some other). In Table 1 D-nets models of main functional devices of elemental basis are shown. Then the each of these D-net models can be used for building models of more complex functional devices. Majority of functional elements for modern microcircuits with combinatorial or consequent logic can be built from such basic base elements.

Table 1. An elements which form the basis for building functional elements and their D-net models

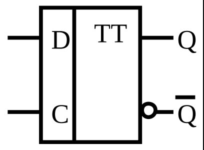
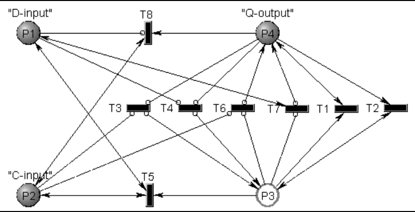
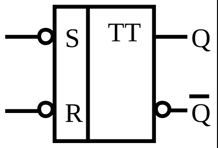
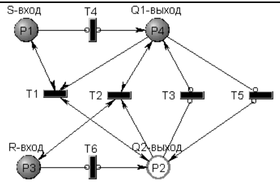
Order Number	Functional element	D-net model	Logical function
Elements with combinatorial logic			
1			"NOT" $y = \bar{x}$;
2			"NOR" $y = \overline{x1 \vee x2}$
3			"NAND " $y = \overline{x1 \wedge x2}$

Composition of D-net models

As a rule, forming the circuit of any digital device consists of choosing the required functional element, putting it in the needed place on the schema and creating the corresponding links between functional components. Each functional element contains input and output pins. The forming link means the uniting some from them in something whole (indivisible). Onto circuit this links are represented in the manner of corresponding graphical image of connection or bus. In this context it is helpful to represent the object as a "black box". In this case modeled object is considered as the sets of inputs, internal states and outputs. Analogously with notion "multipoles" in electrical engineering, we'll call their corresponded subsets in D-net as input poles, output poles and internal states.
It is evident that the control places very suit for the role of the input poles, which associate with transitions only with the help of conditional links (fig.3a). As already mentioned they can affect the marking of the net but the net itself cannot do the same with their marking. The presence of only conditional links beside such places is single formal way to differ them from rest. Places associated with the output signals of the

modeled object perfectly can to play the role of output poles. Indicator-places are very well suited for this purpose. Places, which do not get nor in one of enumerated groups, automatically fall into the category the places represented the only internal states of the object. Will be names their usual places. In some cases, a role of output places can be played by the usual places.
The constructing a model from components consists of two stages: composing and establishing links between its component parts. In a first stage we consecutively add models of the functional circuit elements to the designed model. This process is very similar to syntheses of matrix by adding sub-matrices that are presented in the manner of diagonal elements of this matrix. As result we get a model composed from a components that are no linked between itself else. In a second stage we set the links between input and output poles D-nets models of diagonal elements in a resulting model. If don't take into account the time delays in connected wires, then the linked poles can merge in one place. As a result instead of input and output poles we get new place that inherit all the links of united poles. This methodic has been used for building models for standard functional elements of integral microcircuits. Some of them are presented in table 2.

Table 2. A functional elements with sequential logic and their D-net models.

Order number	Functional element	D-net model
	Elements with sequential logic	
1		 "D-input" T8 "Q-output" P1 P4 T3 T4 T6 T7 T1 T2 "C-input" T5 P2 P3
2		 S-вход T4 Q1-выход P1 P4 T1 T2 T3 T5 R-вход T6 Q2-выход P3 P2

The conditional division of the places into poles is presented very useful. Because it gives formal possibility to link (unify) the process of synthesizing the model with the process of designing of the real digital device by its circuit. Except of this notions of "poles" can use for building hierarchical representation of the D-nets model in the same way as this marketed CPN-tools [J97].

Implementation

Thereby though D-nets have similarities to no pure timed PN with inhibitor arcs, they also has the evident discriminating particularities, which are unique only for them. D-nets allow better take into account particularities of the behavior of digital devices and offer kit of some new additional facilities and possibilities that allow to relieve interface between circuital presentation of device and model during its designing, simulating and analysis. First of all this relates to particularities of structural formations into D-nets. Presence a formal correspondence between places of model and pins of circuit allows us easy to convert marking of the D-net to ensemble of logical levels of electrical signals on pins in a circuit and on the contrary. Further this will allow hide model itself from designer and afford to him almost unique possibility fulfill experimental researching with worked out device directly through his circuital image. In result, designer may easy to choose and change states of electrical signals and to observe the corresponding reaction of model directly on the circuital picture. The similarity of processes of constructing a model and forming digital circuits allows during designing circuit automatically to get its model.

For checking main conceptual positions mortgaged in base of D-extended PN, for study of properties and particularities and for evaluation the prospects of using this extension was developed tool that got a name of DPN-tool. Herewith main task of this development was create the visual environment of designing which could ensure the possibility create D-nets models implicitly (not explicitly) through their graphical representation in the manner of graph. Under its development were taken into account results of the analysis of modern tools for modeling by Petri nets and used recommendations on the building of similar facilities, worded in works [YK98].

Development was carried out on C++ language, in integrated development's environment BorlandC++ Builder-4.0. The principle of the object-oriented approach was put in a basis of development of DPN-tool [Ve00]. The submitted DPN-tool is intended for operation in environment of operational system MS Windows-95 and higher and ensures the functioning in following basic modes:

– Creation new and editing of already existing D-nets models.
– Performance simulating experiments with models.
– Performing the transformations of D-net models (reduction).
– Construction of reachable trees and their analyze.

General structure of DPN-system is shown on fig. 6.

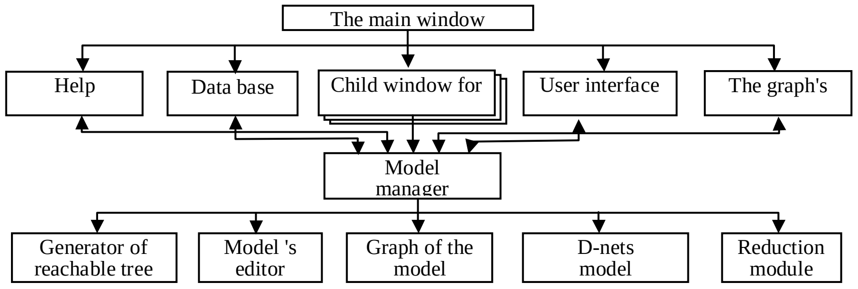


Fig.6. General structure of the instrumental simulating DPN-system.

The manager of the models built-in in each window of editing and presents itself main an element of system. It contain of itself facilities for creating and editing of D-nets models in the manner of correspond graph, its transformation onto internal (into computer) presentation of D-net models, executing of model with simultaneously displaying of its current states and preserving of constructed models and the results of

researches on disk. Besides the manager provides an interaction with a database, controls of working modes of the subsystems subordinated for it and carries out interactions with the graphic buffer. As it was already spoken, graph was used as the main visual form of models image. So to the manager, among other duties were added a duty of observation for change of graph and well-timed corresponded correcting of D-net's model.
Main window of application exhibit as it shown on fig.7 and contain menu, main and secondary control panels and status bar in which the current state of modeling system is displayed. The developed tool simultaneously allows the user to work with several models, each of which is displayed in its separated (child) window.

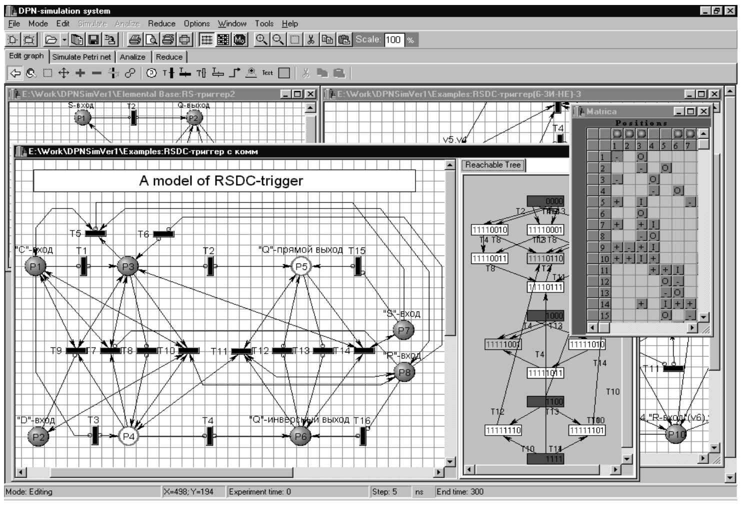


Fig. 7. Exterior of main window of an application.

Application provides enough a simple and convenient in use graphic interface. The main purpose of it is to performance of two tasks. First, it allows the user to draw D-net in a most natural fashion and intuitively clear for researcher the form of displaying of model (in form of graph). Second, it allow to raise efficiency of DPN-tool due to automation such operations as operations of work with the graphic buffer, interactions with file system and databases, scaling and other.

Conclusion

Thereby, having saved simplicity of general PN, D-nets enables to more adequately represent behavior of digital devices and more better take into account particularities of the technology of their construction. Generated from a no pure timed PN with inhibitor arcs, D-nets also have clearly the discriminating particularities, which are unique only for them. First of all this relates to rules of execution of D-nets and behavioral particularities of their structural formations.
D-extended PN ensures more natural interpretation of properties and notions from PN domain into same properties and notions from domain of digital circuits. The results of conducted studies of D-nets extension and the usage of developed DPN-tool have shown the following:

– correctness of basic principles and of strategy of building of D-net extension;
– high adequacy of representation for real discrete systems with help D-net expansion;
– an opportunity of modeling of not only hardware, but also and of software (algorithms, protocols, interfaces etc.);
– reliable operation of developed tool in various regular of its modes.

Hereinafter, a given system is supposed to be used as a component part in the frameworks of more general system for modeling of digital devices. The conducted researching DPN-system allow to hope that functional scheme of digital device (circuit) can be used as unique language for interfacing with modeled object.

Bibliography

[PJ81] Peterson, James L. Petri net Theory and the Modeling of Systems. Englewood Cliffs, New Jersey: Prentice Hall, Inc., 1981.
[RR98a] Wolfgang Reisig, Grzegorz Rosenberg(ed.). Lectures by Petri nets: advances in Petri nets. Springer. Berlin. 2. Applications. 1998.
[W98] Jiacun Wang. Timed Petri nets. Theory and Application. Florida International University. Kluwer Academic Publisher. 1998.
[RR98b] Lectures on Petri Nets I: Basic Models. Advanced in Petri Nets/Volfgang Reisig; Grzegorz Rozenberg(ed.). -Springer-Verlag Berlin Heidelberg New York. 1998.
[Va90] Victor I. Varshavsky, editor. Self-Timed Control of Control Processes: The Design of Aperiodic Logical Circuits in Computers and Discrete Systems. Kluwer Academic Publisher, Dordrecht, The Niderlands, 1990.
[NP86] Nikonov V.V., Podgursky U.E. Application. Using of Petri nets. – Foreign radioelectronica, 1986г., №11.(In Russian)
[SP01] Enrique Soto, Miguel Pereira. Implementing a Petri Net Specification in FPGA using VHDL. The International Workshop on Discrete-Event System Design, DESDes'01, 27-29 June, 2001.
[M89] Murogov V.N. The system of simulating on E-nets base. – Microprocessor's facilities and systems, 1989г. №1. (In Russian)
[G00]Gultiaev A. Visual modeling onto MATLAB environment. Course for teaching – SPb: Piter. 2000. (In Russian)
[Ve90] Veselov A.A. The system for simulating of digital devices of automation by Petri nets. Material to international research conferences "New information's technologies". Tver, "CenterProgramSystem", "NovInTech" firm. 1990. (In Russian)
[Ve99] Veselov A.A. The extention of Petri nets for modeling digital and computing technique. Material to unionrussian conference " Prospects of development of Volga's region". Tver, TSTU (Tver State Technical University), 1999. pp 131-134. (In Russian)
[BD98] Berthomieu B., Diaz M. Modelling and verification of time dependent system using time Petri nets // International Journal on Software tools for technology Transfer – 1998.-P.98-132
[J97] Kurt Jensen. Colored Petri nets. Basic concepts, analisis methods and practical use/ Vol.1, Second ed.. Springer-Verlag Berlin Heidelberg New York. 1997.
[Br96] Richard Scott Brink A PETRI NET DESIGN, SIMULATION, AND VERIFICATION TOOL. A Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of MS in CE. Department of Computer Engineering College of Engineering Rochester Institute of Technology Rochester, New York. 1996. On-line version: www.csh.rit.edu/~rick/thesis/doc/PetriThesis.html.
[YK98] A.V.Yakovlev, A.M.Koelmans. Petri nets and Digital Hardware Design. Lectures on Petri nets: advances on Petri nets / Wolfgang Reisig; Grzegolz Rosenberg (ed). Berlin. 2: Applications.-1998.
[Ve00] Veselov A.A. Program's implication simulation's models on base D-extension of Petri nets. InterInstitute collection of scientific works “Designing technological and medico-biological systems”. TSTU (Tver State Technical University), 2000. pp.130-135. (In Russian)