SAFE PETRI NETS MODELS MAPPING INTO FPGA USING HDL CODE

V. Ababii, V. Sudacevschi

Technical University of Moldova
avv@mail.utm.md, svm@mail.utm.md

Abstract: In this paper a hardware implementation method of Safe Petri Nets (SaPN) models is proposed. Mapping of SaPN models into FPGA is based on creating of the connections between selected functional elements places P and transitions T according to the incidence matrix. The method allows the implementation of SaPN models of any complexity by reason of flexible architecture of functional elements. Proposed method makes possible the economical and structured FPGA implementation according to AHDL description of processing elements.

Keywords: Safe Petri Nets, FPGA, Hardware implementation of Petri Nets models, parallel data processing.

1. Introduction

Petri nets are often used for modeling and verification of distributed control system behavior. The most essential features of such systems are event connections between components, scheduling of algorithms within a device, communications between devices and resources and asynchronous execution of algorithms in different devices. Petri net represent a powerful formal basis for the representing of such systems [1] and can be used to check for potential deadlocks in the system, competition, cooperation and synchronization of processes.

A hardware implementation of Petri net model opens new possibilities in system analysis. It allows to diminish the simulation time [2] and to reduce the number of steps necessary for system design [3, 4].

In this paper is analyzed only 1-bounded Petri net which is called Safe Petri Net (SaPN). A new method of SaPN mapping into Field Programmable Gate Area (FPGA) structure is presented, based on AHDL description of functional elements of the analyzed SaPN model – places and transitions.

2. Safe Petri Net Model

In general, a Petri net is a five-tuple, $PN=\{P,T,In,Out,M\}$,

where: $P=\{p_1,p_2,\ldots,p_n\}$ is a finite set of places,

$T=\{t_1,t_2,\ldots,t_m\}$ is a finite set of transitions,

$n(t)=\{p \mid (p,t)\in F\}$ the set of input places of t,

$Out(t)\ )=\{p \mid (t,p)\in F\}$ the set of output places of t,

$M=\{\mu(p_1),\ \mu(p_2),\ldots,\ \mu(p_n))$ is the set of markings of places,

where: $F\subseteq(P\times T)\cup(T\times P)$ is a set of all arcs of Petri net and represent the incidence matrix. The number of columns is equal to the number of places n, the number of rows is equal to the number of transitions m. The matrix elements f_{ij} are defined according to the following relations:

$$f_{ij}=\begin{cases}1,\ \textit{if exist an arc from } p_j \textit{ to } t_i,\\ -1,\ \textit{if exist an arc from } t_i \textit{ to } p_j,\\ 2,\ \textit{if exist an arc from } p_j \textit{ to } t_i \textit{ and an arc from } t_i \textit{ to } p_j,\\ 0,\ \textit{if do not exist any arc between } p_j \textit{ and } t_i,\end{cases}$$

$$\forall\ i=\overline{1,m},\ j=\overline{1,n}.$$

For Safe Petri net $\mu(p_i) \in \{0,1\}$.

The initial marking of places can be defined as $M_0=\{\mu_0(p_1), \mu_0(p_2),\ldots, \mu_0(p_n)\}$

The marking describes the state of the adequate dynamic system, and dynamic changes are modeled as tokens movement from one place to another.

Transition $t \in T$ is enabled in marking M_k, if $\forall p \in P : \mu_k(p) = (p,t)$. Any transition enabled in M_k can fire, changing the marking of any $p \in P$ to marking M_{k+1}, according to conditions

$$M_{k+1}(p) = M_k(p) - (p,t) + (t,p),$$

k - it is a step of data processing.

3. FPGA based design

The advent of Field-Programmable Gate Array circuits allows using them as hardware solutions for simulation of Petri Net models with a large number of states. FPGA circuits contain thousands of gates equivalents and provide enough logic to implement several small processors on a single chip. There are many available tools for their programming as high level hardware description languages (e.g. VHDL, AHDL, Verilog, ABEL, PALASM, Max+Plus [5]) or traditional schematics. These languages also can be used to check the model before it is loaded on to the hardware. The possibility of a run-time reconfiguration allows the use of adoptive algorithms that can reduce the time that is necessary for Petri Net simulation.

In the literature several studies of FPGA implementation of Petri Net have been described. In [3] reachability analyze of Petri Net by using an FPGA based accelerator is proposed. An FPGA implementation to execute the Petri Net transition firing algorithm is described in [4].

Mapping of a SaPN model into FPGA is based on creating of the connections between selected functional elements places P and transitions T according to the incidence matrix. Functional elements are selected from a Standard Library in dependence of their number of inputs and outputs. Standard library contents AHDL description of these elements. Logical structure for FPGA design is presented on fig. 1.

4. Hardware implementation

Hardware block is connected to the ISA system bus and consist of Interface and FPGA-based SaPN model (fig. 2).

Interface is built on the basis of CPLD (MAX 3000 A) circuit. For SaPN model FPGA implementation was used SPAM FPGA FLEX10K technology, which allows on-line structure configuration. As global states of SaPN model are stored in internal RAM memory of FLEX10K, it is not necessary to transmit processed data after each operation that increases the processing speed of the system.

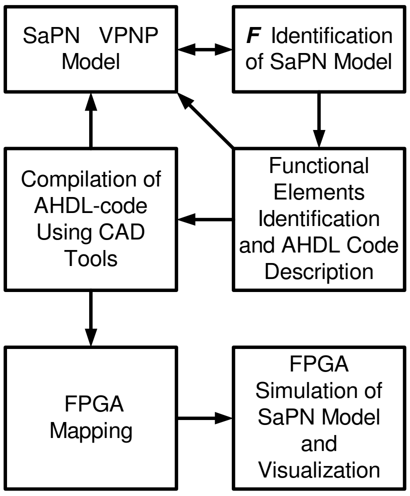
SaPN VPNP Model
F Identification of SaPN Model
Compilation of AHDL-code Using CAD Tools
Functional Elements Identification and AHDL Code Description
FPGA Mapping
FPGA Simulation of SaPN Model and Visualization

Figure 1. Steps for FPGA Design

Interface block consist of:

- **DB** (Data Bus) – bidirectional data bus;
- **I/O** Control (Input/Output Control) – the input/output control block;
- **DC** (Decoder) – Address bus decoder.

The FPGA-based SaPN models consists of:

- ***T*** – set of Transitions;
- ***P*** – set of Places;
- **Data RAM** – memory where is stored the reachability graph;
- **B SYN** – synchronization block for data processing;
- **RAM SYN** – memory where is stored the synchronization signals for data processing.

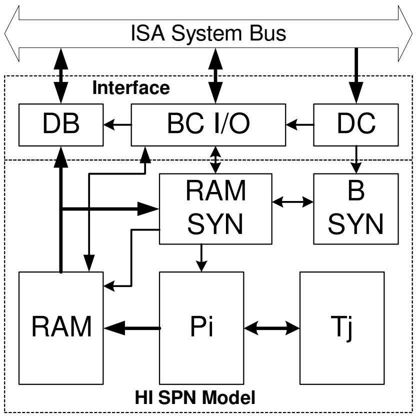

Fig. 2. Interface and Hardware Safe Petri Net Architecture

Via JTAG interface is written the configured file which determines the SaPN model structure. Synchronization file for data processing is loaded in RAM SYN. Synchronization block **B SYN** read the information from **RAM SYN** and enables the corresponding places where the data are processed in parallel. At each synchronization step the SaPN state is written in **Data RAM**. When **Data RAM** memory become full, a special signal DRQ is generated which initialize DMA process and allows writing the information from **Data RAM** to PC memory.

5. AHDL description of functional elements

AHDL description of the internal PN structure consists of several blocks that can be easily modified. The concrete PN model is done by interconnection of these blocks.

AHDL-code for place P.

```
SUBDESIGN place1
(         Dec0, Dec1, Dec2, Dec3, Inc0, Inc1, Inc2,
Inc3,  SYN, nCLR, nPR  : input;
          Mout : output; )
VARIABLE      FF       : SRFF;
                         Dec : NODE;
                         Inc : NODE;
BEGIN
          Dec = (Dec0 # Dec1 # Dec2 # Dec3);
          FF.R = Dec;
          Inc = (Inc0 # Inc1 # Inc2 # Inc3);
          FF.S = Inc;
          FF.clk = SYN & (Dec $ Inc);
          FF.clrn = nCLR;
          FF.prn = nPR;
          Mout = FF.q;
END;
```

Suggested AHDL code satisfies the following conditions

$$\mu_{k+1}(p_i)=\begin{cases}\mu_k(p_i)+1,\ if\ \left(\mu_k(p_i)=0\ \&\ \left(I\ \&\ \overline{D}\right)\right),\\ \mu_k(p_i)\ if\ \left(\overline{I}\ \&\ \overline{D}\right),\\ \mu_k(p_i)\ if\ \left(I\ \&\ D\right),\\ \mu_k(p_i)-1\ if\ \left(\mu_k(p_i)=1\ \&\ (\overline{I}\ \&\ D)\right).\end{cases}\quad \forall\, i=\overline{1,n}$$

AHDL-code for transition T.

```
SUBDESIGN transition4
(         Min0, Min1, Min2, Min3: input;
          Dec_Inc : output;
)
BEGIN
          Dec_Inc = (Min0 & Min1 & Min2 &
Min3);
END;
```

6. Conclusions

In this paper a new method of SaPN mapping into FPGA structure is presented, based on AHDL description of functional elements of the analyzed SaPN model – places and transitions. This method allows the parallel data processing in places, storage of all states of SaPN model into RAM and their transmission to PC memory. The future research direction consists in conflict identification that appears during parallel processing and in dynamic memory organization for data storage in order to optimize the internal FPGA memory utilization.

References

[1]. Peterson, J. *Petri Net theory and modeling of systems.* New York, 1984.

[2]. Garlic A., Gutuleac E., Enicov I. *A Software Tool for Distributed System Evaluation* // Proceedings of the Symposium on Electronics and Telecommunications, vol. 3, Romania, Timisoara, Sept., 1994.

[3]. G. A. Bundell. *An FPGA implementation of the Petri Net firing algorithm.* In Proc. 4th Australasian Conf. on Parallel and Real-Time Systems, pp. 434-445, 1997.

[4]. John Morris et al. *A Re-configurable Processor for Petri Net Simulation.* Proceedings of the 33rd Hawaii International Conference on System Sciences – 2000.

[5]. http://www.altera.com/.