An Extended Petri Net EPRES for Embedded System Modeling

Sen Liu, Chundi Mu
Department of Automation, Tsinghua University, Beijing 100084, China
lius00@mails.tsinghua.edu.cn

Abstract

An appropriate formal model should be established to perform simulation analysis in prophase of embedded system design. This paper analyzes the current Petri net models and presents an EPRES model which gives its structural definition, graphical representation and behavioral rules concretely. The places were extended to two types such as control places and data places in EPRES. Transition function and transition time delay were introduced. Then EPRES can specify and analyze the function realization, resource consumption and time constraint. Finally, an example of the EPRES representation of an embedded system is given to show that EPRES is suitable for embedded system modeling.

1. Introduction

Embedded systems are characterized by their dedicated functions, real-time behaviors, high requirements on reliability and correctness based on limited hardware and software resource. Designing embedded systems is a difficult task. Moreover, the increasing complexity combined with short time-to-market poses great challengers for the designers. In order to reduce risks, an appropriate formal model should be established so that the simulation analysis for reliability and correctness can be carried out systematically in prophase of embedded system design.

Since C.A.Petri introduced Petri net in 1962, Petri net have been widely applied for system modeling in many fields of science. But now with the development of embedded system, Petri net are more and more used to model embedded system [1-3]. Due to the inability to deal with the characters of embedded system, a number of extensions to the classical Petri net have been proposed such as ETPN, PRES+ and DTPN. Extended Timed Petri Nets (ETPN) consists of two separate but related parts: The control part, captured as a Timed Petri Net (TPN), and the data path, represented as a directed graph where nodes are used to capture data manipulation and storage [4]. However, data and control flow are not very well linked in this model. To overcome the lack of union between these two domains, PRES+ (Petri Net based Representation for Embedded System) employs a single graph to model the control functions and data operations of the embedded system. It can be used to model a system at different levels of abstraction using hierarchy. But it can't describe the control flow adequately [5-6]. DTPN (Dual Transitions Petri Net) uses two sets of transitions to deal with control and data. However, it lacks expressiveness for complex computations and formulation for time characteristic [7-8]. The aim of this paper is to present a new efficient modeling technique suitable for the formal representation of embedded systems.

The rest of this paper is organized as follows. Section 2 presents the formal definition of the model that we use to represent embedded systems. The approach and efficiency of the model are described through an example in section 3. Finally, some conclusions are drawn in Section 4.

2. Extended PRES（EPRES）

The proposed modeling technique in dealing with embedded system is an extension to Petri net, called EPRES (Extended Petri Net based Representation for Embedded System). This section presents the new model and describes its structural definition, behavioral rules and graphical representation.

2.1 Structural Definitions

Definition 1. The structure of EPRES is a six-tuple $N = (P, Q, T, F, W, G)$

Where:

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

$Q = \{q_1, q_2, \ldots, q_m\}$ is a finite set of data places.

$T = \{t_1, t_2, \ldots, t_k\}$ is a finite non-empty set of transitions.

$F = F_C \bigcup F_D$ is a finite non-empty set of arcs which define the flow relation between places and transitions. $F_C \bigcap F_D = \varnothing$. $F_C \subseteq (P \times T) \bigcup (T \times P)$ is a set of arcs describing the control flow relation. $F_D \subseteq (Q \times T) \bigcup (T \times Q)$ is a set of arcs describing the data flow relation.

W is a weight function. $W \subseteq (P \times T) \bigcup (T \times P) \rightarrow R^+$

G is a set of the guard functions describing how the data places influence the control flow.

Definition 2. An EPRES is a seven-tuple

$EPRES = \left(P, Q, T, F, W, G, M^0\right) = (N, M^0)$,

M^0 is the initial marking of the net.

2.2 Description of System

Definition 3. Control places are associated with control signals or physical resource in the embedded system (e.g. switch signal, memory storage, energy, etc). Data places are associated with data such as constants and variables. The marking function M is a mapping function from the set of places $P \cup Q$ to the set of pairs. $\forall q_m \in Q$, $M(q_m) = (v, r)$; $\forall p_n \in P$, $M(p_n) = (u, r)$. v is the token value in the data place. This value may be of any type, e.g. Boolean, integer, etc., or user-defined type of any complexity (for instance a structure, a set, a record); $u \in \overline{R^-}$ represent the number of token in the control place. r is the token time, a finite positive real number representing the time stamp of the token.. In the step k , the marking of EPRES is $M^k = \{M^k(p_1), M^k(p_2), \ldots, M^k(p_m),$ $M^k(q_1), M^k(q_2), \ldots, M^k(q_n)\}$

Definiton 4. $x, y \in P \bigcup T \bigcup Q$, ${}^\bullet x = \{y \mid (y, x) \in F_C\}$ is the control pre-set of x , $x^\bullet = \{y \mid (x, y) \in F_C\}$ is the control post-set of x . ${}^\circ x = \{y \mid (y, x) \in F_D\}$ is the data pre-set of x , $x^\circ = \{y \mid (x, y) \in F_D\}$ is the data post-set of x .

Definition 5. The guard function G_j of a transition t_j is the set of Boolean conditions that must be asserted in order to enable that transition. A condition of a transition $cond_i : \tau(q_1) \times \tau(q_2) \times \ldots \times \tau(q_a) \rightarrow$ $\{0,1\}$ is function of the token values in the data places of the data pre-set of t_j ($q_1, q_2, \ldots, q_a \in {}^\circ t_j$). G_j is the conjunction of all conditions of the transition t_j . If all conditions are asserted G_j=1, otherwise G_j=0. If no guard is explicitly defined, it will be assumed constantly asserted. Let L be a finite set of predicates. G is composed of a number of sub-functions G_j . Formally:

$G : 2^P \rightarrow \{0,1\}$, $2^P = \{{}^\circ t_j \mid j < m, \exists l_j \in L\}$

$\forall j < m \mid \exists l_j \in L \Rightarrow G_j : {}^\circ t_j \rightarrow \{0,1\}$

Definition 6. For every transition t , there exits a transition time delay d , a finite positive real number, which represents the execution time of that transition. $\forall t \in T$, $\exists d \in \overline{R^-}$. If no transition time delay is explicitly defined, it will be assumed 0.

Definition 7. The type function $V : Q \rightarrow U$ is a relation that associates the token value in a data place with a token type. Thus, we will call $V(q)$ the token type associated with data place q .

Definition 8. For some transition t with a non-empty data pre-set (${}^\circ t \neq \varnothing$), there exists a transition function h associated to t . Transition functions are used to capture the behavior of the system to be modeled. They allow system to be modeled at different levels of granularity with transitions representing simple arithmetic operations or complex algorithms. This is:

$h : V(q_1) \times V(q_2) \times \ldots \times V(q_i) \rightarrow V(s)$

Where ${}^\circ t = \{q_1, q_2, \ldots, q_i\}$ and $s \in t^\circ$.

2.3 Graphical representation

Formal analysis of an EPRES can be made in terms of graph theory. A directed, weighted, tripartite graph is used. The graph has three types of vertices and two types of arcs. Control places are graphically represented by single-line circles, data places by dual-line circles, transitions by rectangles, arcs of control flow relations by light arrows, and arcs of data flow relations by thick arrows. Transition functions are inscribed on left and transition time delays on right inside transition rectangles. The conditions are inscribed close to the respective transition rectangle. Inscriptions are used on the input data arcs of a transition in order to denote the arguments of its transition function. The weight value is written on the control arcs of a transition to describe how many tokens are consumed and created by the transition.

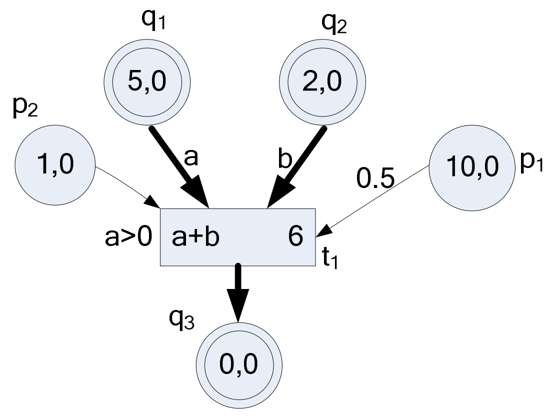

Figure 1. An EPRES graphical model

A simple EPRES is shown in Figure 1 where $P = \{p_1, p_{2,}\}$, $Q = \{q_1, q_2, q_3\}$ and $T = \{t_1\}$ respectively. The initial marking $M^0 = \{(10,0), (1,0), (5,0), (2,0), (0,0)\}$ .The transition of t_1 is $h(t_1) = a + b$, the transition time delay is 6 and the condition is $a > 0$.

2.4 Behavioral rules

Definition 9. The guard function G_j is:

$$G_j(q_i \in {}^\circ t_j) = \begin{cases} 1 & if(\forall q_i)[l_j(v_{q_i})] = true \\ 0 & if(\exists q_i)[l_j(v_{q_i})] = false \end{cases}$$

Definition 10. A transition t_j is said to be enabled if each control place in its control pre-set holds $W(p_i, t_j)$ or more tokens, and its guard function is asserted. Formally, for a given marking *M*, a transition $t \in T$ is enabled iff (if and only if)

$$[\forall p_i \in {}^\bullet t_j \mid u_{p_i} \geq W(p_i, t_j)] \wedge [G_j = 1]$$

Definition 11. Every enabled transition t_j has a trigger time tt_j that represents the time instant at which the transition may fire. Each token in the pre-set of an enabled transition has, in general, a different token time. From the point of view of time, the transition could not fire before the tokens are ready. The concept of trigger time is needed to describe how token times are handled when the transition is fired. The trigger time of an enabled transition is the maximum token time of the tokens in its input places,

$$tt_j = \max(r_{p_1}, r_{p_2}, \ldots, r_{p_a}, r_{q_1}, r_{q_2}, \ldots, r_{q_b})$$

Where ${}^\bullet t = \{p_1, p_2, \ldots, p_a\}$ and ${}^\circ t = \{q_1, q_2, \ldots, q_b\}$

Definition 12. The firing of an enabled transition changes the marking M^k into a new marking M^{k+1} .

$$\begin{cases} \forall p_i \in {}^\bullet t_j, u_{p_i}^{k+1} = u_{p_i}^k - W(p_i, t_j) \\ \forall p_i \in t_j{}^\bullet, u_{p_i}^{k+1} = u_{p_i}^k + W(t_j, p_i), r_{p_i}^{k+1} = tt_j^k + d_{t_j} \\ \forall q_i \in t_j{}^\circ, v_{q_i}^{k+1} = h_{t_j}, r_{q_i}^{k+1} = tt_j^k + d_{t_j} \end{cases}$$

Note that after the transition fires, the token numbers in the control places of the control pre-set and post-set change. It represents the acceptance and creation of control signal, the consumption and release of resource, etc. But the token values in the data places of the data pre-set don't change since the read operations don't change the store data. It is consistent with the known facts.

Figure 1 shows the EPRES model of an add manipulation. In this example if $a > 0$, $c = a + b$, and the operation consumes 0.5 energy unit every time. The data places q_1 , q_2 and q_3 correspond to the variables a , b and c . The control place p_1 corresponds to the energy resource. An assistant control place p_2 is introduced to denote the operation only executes one time. The initial marking $M^0 = \{(10,0), (1,0), (5,0), (2,0), (0,0)\}$. The transition t_1 is enabled since $u_{p_1} = 10 > 0.5$, $u_{p_2} = 1 \geq 1$ and $a = v_{q_1} = 5 > 0 \Rightarrow G_1 = 1$. The trigger time $tt_1 = 0$ and transition time delay d_1 =6. After t_1 fires the marking changes into $M^1 = \{(9.5,0), (0,0), (5,0), (2,0), (7,6)\}$ shown in Figure 2.

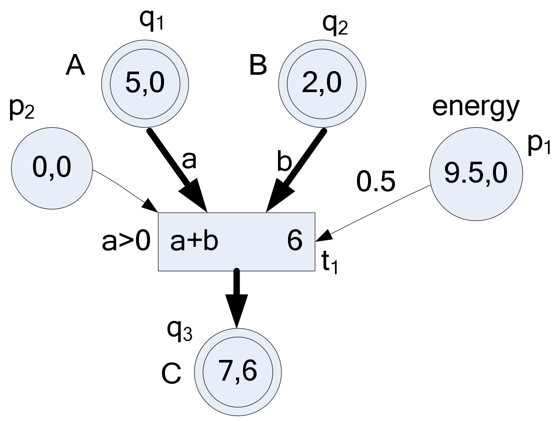

Figure 2. M^1 of the EPRES in Figure 1

3. Application Example

When EPRES is used to model an embedded system, the places and transitions are assigned to represent control signals, resource, data and operations firstly. Then the control and data flow relations, various parameters such as weight and transition time delay are specified actually. Finally based on the former preparatory model, some assistant control places are added to complete the final EPRES.

In paper [5] a multiplier example is proposed. The algorithm calculates the product of two positive integers by means of iterative sums. PRES+ can't describe the energy consumption and DTPN can't capture the time character. EPRES can be used to model the multiplier to overcome their weaknesses. The C description corresponding to this algorithm and the relations between the places, transitions and operations are shown as follows.

```
int mult(int a,int b)        -- q2，q3
{
  int x,y,z;                 --q4，q5，q6
  z=0;                       --t1
x=a;                         --t2
  y=b;                       --t3
  while(y>0)  {
     z=z+x;                  --t4
     y=y-1;                  --t5
  }
  return z;
}
c=mult(a,b);                 --t6
```

Figure 3 illustrates the EPRES model for the multiplier. Some assistant control places such as p_1, p_2, p_3, p_4, p_5 are added to control the execution sequence of operation to guarantee the accurateness of the model.

The initial marking of the system:

$$M^0 = \{(1,0),(0,0),(0,0),(0,0),(0,0),(10,0),(0,0),(5,0),(2,0),(0,0),(0,0),(0,0),(0,0)\}$$

The latter markings can be obtained step by step according to the firing rules aforementioned. They are as follows:

$$M^1 = \{(0,0),(1,1),(0,0),(0,0),(0,0),(9.9,0),(0,0),(5,0),(2,0),(0,0),(0,0),(0,1),(0,0)\}$$

$$M^2 = \{(0,0),(0,1),(1,2),(0,0),(0,0),(9.8,0),(0,0),(5,0),(2,0),(5,2),(0,0),(0,1),(0,0)\}$$

$$M^3 = \{(0,0),(0,1),(0,2),(1,3),(0,0),(9.7,0),(0,0),(5,0),(2,0),(5,2),(2,3),(0,1),(0,0)\}$$

$$M^4 = \{(0,0),(0,1),(0,2),(0,3),(1,9),(9.2,0),(0,0),(5,0),(2,0),(5,2),(2,3),(5,9),(0,0)\}$$

$$M^5 = \{(0,0),(0,1),(0,2),(1,14),(0,9),(8.7,0),(0,0),(5,0),(2,0),(5,2),(1,14),(5,9),(0,0)\}$$

$$M^6 = \{(0,0),(0,1),(0,2),(0,14),(1,20),(8.2,0),(0,0),(5,0),(2,0),(5,2),(1,14),(10,20),(0,0)\}$$

$$M^7 = \{(0,0),(0,1),(0,2),(1,25),(0,20),(7.7,0),(0,0),(5,0),(2,0),(5,2),(0,25),(10,20),(0,0)\}$$

$$M^8 = \{(0,0),(0,1),(0,2),(0,25),(0,20),(7.6,0),(0,0),(5,0),(2,0),(5,2),(0,25),(10,20),(10,26)\}$$

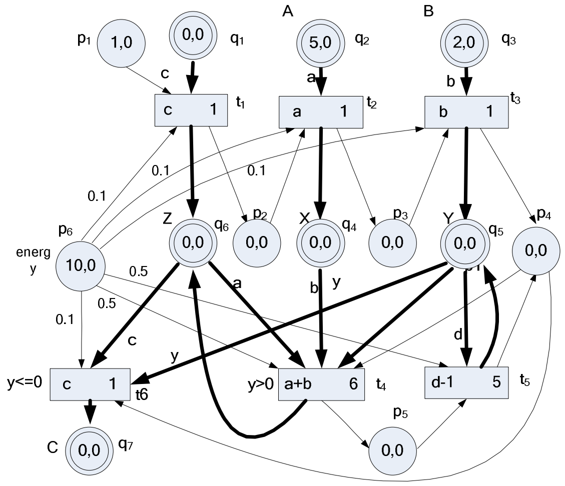

Figure 3. EPRE representation of the multiplier

Finally, the marking $M^8(q_7) = (10,26)$ shows the output result of the multiplication (10) and the total time needed for the operation (26 time units). $M^8(p_6) = (7.6,0)$ shows the 7.6 energy units are left and the whole operation consumes 2.4 energy units.

4. Conclusions

This paper has proposed an EPRES, a Petri net based model suited to embedded systems modeling. In EPRES places are extended to two types as control places and data places. Transition function, trigger time and transition time delay are introduced. EPRES allows capture and analyze relevant information characteristic of function realization, resource consumption and time constraint. It can deal easily with concurrency and sequential behavior, nested loops and conditional operations, which may lead to better results in the co-design of embedded systems specification. In the future we will use this representation to develop a formal modeling tool for the simulation analysis of complex embedded systems.

5. References

[1] T. Murata, Petri net: properties, “analysis and applications”, *Proceeding of the IEEE*, 1989, 77(4), pp. 541-580.

[2] L.P.M Benders, M.P.J Stevens, “Petri net modeling in embedded system design”, *CompEuro’92.Computer Systems and software Engineering Proceedings*, Hague, Netherlands, 1992, pp. 612-617.

[3] P. Maciel, E. Barros, W. Rosenstiel, “A Petri net model for hardware/software codesign”, *Design Automation for Embedded Systems*, 1999, 4(10), pp. 243-310.

[4] Z. Peng, K. Kuchcinski, “Automated transformation of algorithms into register-transfer level implementations”, *IEEE Transactions on Computer-Aided Design of integrated Circuits and Systems*, 1994, 13(1), pp. 150-166.

[5] L.A. Cortés, Eles P, Peng Z, “A Petri net Based Model for Heterogenous Embedded Systems”, *IEEE NORCHIP Conference*, Oslo, Norway, 1999, pp. 248-255.

[6] L.A. Cortés, P. Eles, Z. Peng, “Modeling and formal verification of embedded systems on a Petri net representation”, *Journal of Systems Architecture*, 2003, 49, pp. 571-589.

[7] M. Varea, B. AI-Hashimi, “Dual Transitions Petri Net Based Modeling Technique for Embedded Systems Specification”, *Proceedings of Design, Automation and Test in Europe*, Munich, Germany, 2001, pp. 566-571.

[8] Y.G. Luo, J.Z. Gu, “A New Kind of Petri Net: Double Transition Timed Petri Net”, *Computer Engineering*, 2002, 28(8), pp. 45-47. (in Chinese)