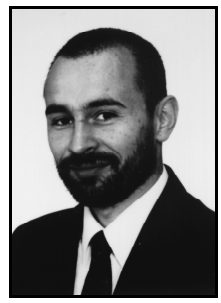
Grzegorz ANDRZEJEWSKI, Piotr MRÓZ
UNIWERSYTET ZIELONOGÓRSKI, INSTYTUT INFORMATYKI I ELEKTRONIKI

Realizacja hierarchicznych sieci Petriego z wykorzystaniem sterowników klasy PLC

Dr inż. Grzegorz ANDRZEJEWSKI

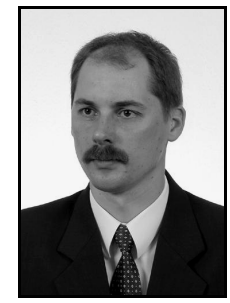
Absolwent Wydziału Elektrycznego Politechniki Poznańskiej, dyplom w specjalności elektronicznej aparatury i systemów pomiarowych uzyskał w roku 1995. Stopień doktora z zakresu informatyki uzyskał na Wydziale Informatyki Politechniki Szczecińskiej w roku 2002. Jego zainteresowania naukowe ukierunkowane są na zagadnienia modelowania i syntezy systemów sterowania cyfrowego.



e-mail: g.andrzejewski@ iie.uz.zgora.pl

Dr inż. Piotr MRÓZ

Absolwent Wydziału Elektrycznego Wyższej Szkoły Inżynierskiej w Zielonej Górze, dyplom (1990) w zakresie automatyki i metrologii elektrycznej. Stopień doktora nauk technicznych uzyskał na Wydziale Elektrotechniki, Informatyki i Telekomunikacji Uniwersytetu Zielonogórskiego (2002). Zainteresowania naukowe koncentrują się w zakresie mikroprocesorowych urządzeń i systemów testujących aparaturę pomiarową i sterującą.



e-mail: p.mroz@iie.uz.zgora.pl

Streszczenie

W artykule przedstawione zostały zasady realizacji hierarchicznych, interpretowanych sieci Petriego (HPN) w sterownikach klasy PLC. Jako język implementacji wybrano graficzny język drabinkowy (LD), ze względu na jego uniwersalny charakter. Proponowaną metodę realizacji zaprezentowano na szeregu przykładach, przedstawiających różne aspekty opisu, wykorzystywane w sieciach hierarchicznych.

Słowa kluczowe: hierarchiczne sieci Petriego, programowalne sterowniki logiczne PLC.

Realization of Hierarchical Petri net by means of PLC

Abstract

The paper presents rules of implementation of interpreted Hierarchical Petri nets (HPN) by means of PLC. The Ladder Diagram (LD) language has been used because of its universal character. The proposed method is shown on a set of examples, which are describing various aspects of formal description used in HPN model.

Keywords: hierarchical Petri nets, programmable logic controllers PLC.

1. Wstęp

Rozwój narzędzi do modelowania zachowania systemów sterujących pociąga za sobą konieczność opracowywania nowych metod realizacji tych modeli w istniejących pakietach wspomagających projektowanie takich systemów. Jedną z bardziej popularnych platform realizacyjnych w praktyce inżynierskiej jest platforma programowalnych sterowników logicznych PLC (ang. *Programmable Logic Controller*). Opracowano dla niej i znormalizowano kilka języków programowania, m.in. graficzny język drabinkowy LD (ang. *Ladder Diagram*) [4]. Jest on chyba najczęściej wykorzystywanym, ze względu na swoją elastyczność opisu i długotrwałe przyzwyczajenia inżynierów do projektowania w kategoriach schematów przekaźnikowych. Jakkolwiek jest to dobry język do realizacji skomplikowanych procesów sterowania, to już niekoniecznie stanowi on czytelny i zrozumiały model opisu zachowania projektowanego systemu. Istnieje wiele modeli pozwalających na opisywanie różnych aspektów sterowania cyfrowego. W artykule przedstawiono zasady realizacji modelu hierarchicznej sieci Petriego (ang. *Hierarchical Petri Net*), ze względu na bogaty zasób elementów opisu tego modelu: zależności czasowe, hierarchię behawioralną, pamięć stanu wybranych fragmentów sieci oraz obsługę sytuacji wyjątkowych.

2. Hierarchiczna sieć Petriego

Formalna definicja modelu hierarchicznej sieci Petriego została przedstawiona w [1, 2]:

$$HPN = \{P,T,F,S,\mathcal{T},\chi,\psi,\lambda,\alpha,\varepsilon,\tau\} \tag{1}$$

gdzie: P jest skończonym niepustym zbiorem miejsc; T jest skończonym niepustym zbiorem tranzycji; F jest skończonym niepustym zbiorem łuków, takim że $F = F_o \cup F_e \cup F_i$, gdzie F_o, F_e, F_i oznaczają odpowiednio zbiory łuków zwykłych, zezwalających oraz zabraniających; S jest skończonym niepustym zbiorem sygnałów, $\mathcal{T}$ jest dyskretną skalą czasu; χ jest funkcją hierarchii, określającą zbiór bezpośrednich podwęzłów miejsca p; ψ jest dwuwartościową funkcją historii, przyporządkowującą każdemu miejscu p takiemu że: $\chi(p) \neq \varnothing$ atrybut historii, λ jest funkcją etykietującą, przypisującą do węzłów wyrażenia (*cond*, *abort*, *action*) budowane z elementów zbioru S, α jest funkcją znakowania początkowego, miejsca początkowe graficznie wyróżnione są znacznikiem wewnątrz okręgu reprezentującego miejsce; ε jest funkcją znakowania końcowego miejsca końcowe graficznie wyróżnione są znakiem × wewnątrz okręgu reprezentującego miejsce; τ jest funkcją czasu, przypisującą liczbę z dyskretnej skali czasu do zbioru węzłów N.

Zbiorem miejsc końcowych w podsieci skojarzonej z makromiejscem p jest taki zbiór P_p^{end}, że:

$$\underset{p' \in \chi(p)}{\forall} \varepsilon(p') = true \Rightarrow p' \in P_p^{end} \tag{2}$$

Funkcją zbioru miejsc końcowych nazywamy taką funkcję $\xi: P \rightarrow P$, która dla makromiejsca p zwraca jego zbiór miejsc końcowych:

$$\xi(p) = P_p^{end} \tag{3}$$

Najbliższym przodkiem węzła n' (ang. *lowest ancestor*) nazywamy miejsce p, takie że: $n' \in \chi(p)$, co zapisać można: la(n')=p.

Działanie tak przedstawionej sieci opisać można z wykorzystaniem warunków zezwolenia tranzycji oraz akcji związanych z jej wykonaniem.

Warunki zezwolenia tranzycji *t*:

$$\underset{p \in P}{\exists} la(t) = p \Rightarrow ac(p) = true \tag{4a}$$

$$\underset{p \in P_t^{in(o)} \cup P_t^{in(e)}}{\forall} ac(p) = true \tag{4b}$$

$$\underset{p \in P_t^{in(i)}}{\forall} ac(p) = false \tag{4c}$$

$$cond(t) = true \tag{4d}$$

$$\underset{p \in P_t^{in(o)}}{\forall} \chi(p) \neq \varnothing \Rightarrow \underset{p' \in \xi^*(p)}{\forall} (ac(p') = true \; oraz \; \tau(p') = 0) \tag{4e}$$

$$\underset{p \in P_t^{in(o)}}{\forall} \tau(p) = 0 \tag{4f}$$

$$abort(t) = true \tag{4g}$$

Warunki te tworzą warunek ogólny zezwolenia tranzycji:

$$gc = a*b*c*(d*e*f+g).$$

Do opisu akcji związanych z wykonaniem tranzycji t użyto oznaczeń, gdzie ac(p,t_e) oznacza stan posiadania (lub nie posiadania) znacznika przez miejsce p w momencie czasu, w którym nastąpiło opuszczenie znacznika z makromiejsca la(p). Akcje *e-j* wykonywane są dopiero, gdy τ(t) = 0. Akcja *k* jest wykonywana przez cały okres aktywności tranzycji t.

$$\forall_{p\in P_t^{in(o)}} ac(p) := false \tag{5a}$$

$$\forall_{p\in P_t^{in(o)}} \chi(p)\neq\varnothing \Rightarrow \forall_{p'\in\chi^*(p)} ac(p') := false \tag{5b}$$

$$\forall_{p\in P_t^{in(o)}} \forall_{s\in action(p)} s := false \tag{5c}$$

$$\forall_{p\in P_t^{in(o)}} \chi(p)\neq\varnothing \Rightarrow \forall_{p'\in\chi^*(p)} \forall_{s\in action(p')} s := false \tag{5d}$$

$$\tau(t)=0 \Rightarrow \forall_{p\in P_t^{out}} ac(p) := true \tag{5e}$$

$$\forall_{p\in P_t^{out}} \chi(p)\neq\varnothing \Rightarrow (\forall_{p'\in\chi^*(p)} ac(la(p'))=true \ oraz \ \psi(la(p'))=false \Rightarrow ac(p') := \alpha(p')) \tag{5f}$$

$$\forall_{p\in P_t^{out}} \chi(p)\neq\varnothing \Rightarrow (\forall_{p'\in\chi^*(p)} ac(la(p'))=true \wedge \psi(la(p'))= true \wedge \exists_{p''\in\xi(la(p'))} ac(p'')=false \Rightarrow ac(p') := ac(p', t_e)) \tag{5g}$$

$$\forall_{p\in P_t^{out}} \chi(p)\neq\varnothing \Rightarrow (\forall_{p'\in\chi^*(p)} ac(la(p'))=true \wedge \psi(la(p'))= true \wedge \forall_{p''\in\xi(la(p'))} ac(p'')=true \Rightarrow ac(p') := \alpha(p')) \tag{5h}$$

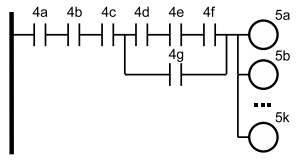
$$\forall_{p\in P_t^{out}} \forall_{s\in action(p)} s := true \tag{5i}$$

$$\forall_{p\in P_t^{out}} \chi(p)\neq\varnothing \Rightarrow (\forall_{p'\in\chi^*(p)} ac(p')=true \Rightarrow \forall_{s\in action(p')} s := true) \tag{5j}$$

$$\tau(t,t_0)=\eta \Rightarrow \forall_{s\in action(t)} s := true \ w \ przedziale <t_0, t_0+\eta+1> \tag{5k}$$

3. Realizacja sieci w języku LD

Prezentowana metoda realizacji bazuje na opisie działania sieci. Realizacja jest możliwa z wykorzystaniem dowolnego języka programowania sterowników PLC, jednakże ze względu na dużą popularność zdecydowano się na język LD. Dla każdej tranzycji tworzony jest odrębny element realizacyjny - szczebel (ang. *rung*), opisany zgodnie z zasadą: jeśli spełniony jest warunek ogólny zezwolenia tranzycji, to wykonywane są dla niej stosowne akcje (rys. 1) [3].

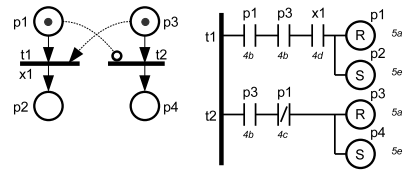


Rys. 1. Ogólna ilustracja metody realizacyjnej
Fig. 1. The general illustration of implementation method

Realizację poszczególnych warunków i akcji ukazano na szeregu prostych przykładów, obrazujących wykorzystanie w modelu konkretnych elementów opisu.

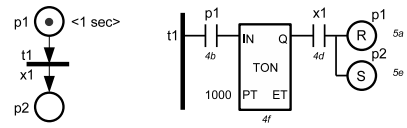
Na rys. 2 przedstawiono przykład użycia łuków zezwalających i zabraniających oraz ich realizację w języku LD.

Nie wszystkie warunki i akcje są zawsze istotne dla realizacji, np. w przykładzie z rys. 2. nie użyto opisu hierarchicznego, zależności czasowych, obsługi wyjątków, wobec czego warunki 4a, 4e, 4f i 4g mogą być pominięte. To samo dotyczy akcji 5b, 5d, 5f, 5g, 5h, 5j, 5k. Ze względu na brak etykietowania miejsc sygnałami wyjściowymi nie uwzględniono również akcji 5c, 5i.



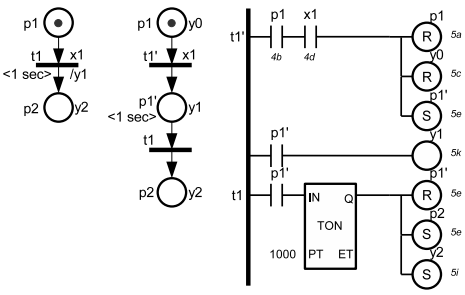
Rys. 2. Realizacja łuków zezwalających i zabraniających
Fig. 2. An implementation of enabling and prohibiting arcs

Na rys. 3 przedstawiono realizację akcji czasowych związanych z miejscem. Do tego celu wykorzystano standardowy układ czasowy TON (ang. *Timer On Delay*), na którego wejście *PT* podano zadaną wartość czasu wyrażoną w milisekundach.



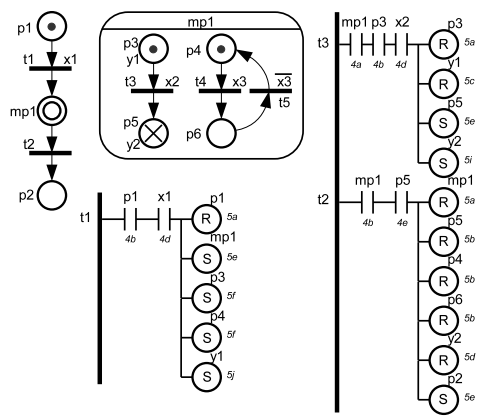
Rys. 3. Realizacja akcji czasowej związanej z miejscem
Fig. 3. An implementation of the time action assigned to a place

Na rys. 4 przedstawiono realizację etykiet nałożonych na miejsca, tranzycje oraz realizację akcji czasowych związanych z tranzycją. W tym celu wprowadzono dodatkowe miejsce *p'* oraz tranzycję *t1'* celem sprowadzenia problemu do realizacji akcji czasowej związanej z miejscem.



Rys. 4. Realizacja akcji czasowej związanej z tranzycją
Fig. 4. An implementation of the time action assigned to a transition

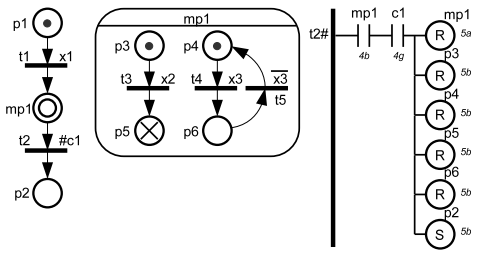
Na rys. 5 przedstawiono fragment sieci z makromiejscem *mp1* oraz ze skojarzoną z nim podsiecią. Jeśli makromiejsce nie posiada atrybutu historii, to w momencie jego aktywacji aktywność uzyskują wszystkie miejsca w podsieci oznakowane jako początkowe {*p3*,*p4*} – akcja 5f.



Rys. 5. Realizacja hierarchii
Fig. 5. An implementation of the hierarchy

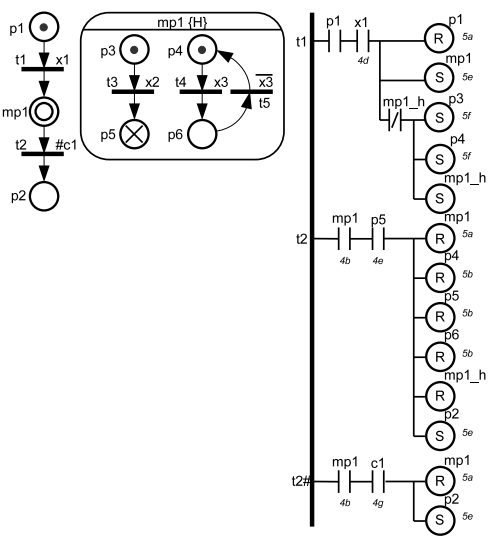
Warunkiem zezwolenia tranzycji *t3* jest m.in. oznakowanie makromiejsca *mp1* w sieci nadrzędnej – warunek 4a. Warunkiem zezwolenia tranzycji *t2* jest m.in. osiągnięcie wszystkich miejsc końcowych w podsieci {*p5*} – warunek 4e. W momencie deaktywacji makromiejsca *mp1* deaktywowane są wszystkie miejsca w podsieci – akcja 5b.

Na rys. 6 przedstawiono własność obsługi sytuacji wyjątkowej oznaczonej warunkiem wywłaszczenia makromiejsca *#c1*. W takiej sytuacji nie jest konieczne osiągnięcia miejsc końcowych w podsieci, a w momencie wywłaszczenia deaktywowana jest cała podsieć, niezależnie od stanu w jakim się znajdzie. Jeśli podsieć osiągnie wcześniej miejsca końcowe (przed wywłaszczeniem), to podsieć deaktywowana jest tak samo jak na rys.5. Wobec powyższego do drabinki przedstawionej na rys. 5 wystarczy dodać jeden szczebel obsługi wywłaszczenia makromiejsca.



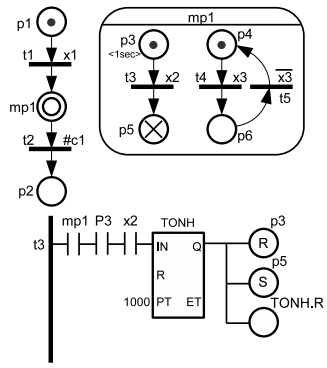
Rys. 6. Realizacja własności wywłaszczenia
Fig. 6. An implementation of the preemption

Na rys. 7 przedstawiono realizację atrybutu historii związanego z makromiejscem. Podsieć po wywłaszczeniu musi zapamiętać swój stan i po ponownej aktywacji makromiejsca aktywowane są te miejsca w podsieci, które były ostatnio aktywne – akcja 5g. Jeśli w podsieci osiągnięte zostaną miejsca końcowe, to podsieć deaktywowana jest tak samo jak na rys. 5 i przy ponownej aktywacji znakowane są miejsca początkowe – akcja 5h. Celem wyróżnienia sposobu deaktywacji makromiejsca wprowadzono dodatkową zmienną *mp1_h*.

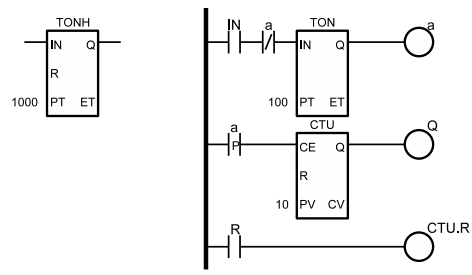


Rys. 7. Realizacja atrybutu historii związanego z makromiejscem
Fig. 7. An implementation of the history attribute assigned to a macroplace

Na rys. 8 przedstawiono przykład realizacji akcji czasowej związanej z miejscem w podsieci z atrybutem historii. W momencie wywłaszczenia makromiejsca wszystkie miejsca powinny być deaktywowane, ale stan akcji czasowych powinien zostać zapamiętany. W tym celu można wykorzystać blok układu czasowego, takiego, w którym można zatrzymać proces odmierzania czasu. Na rys. 9 zamieszczono przykład realizacji takiego układu czasowego TONH, zbudowanego z wykorzystaniem standardowych elementów TON (ang. *Timer On Delay*) oraz CTU (ang. *Count Up Counter*). Blok układu czasowego TON wykorzystywany jest do odmierzania interwałów czasowych (tu 100ms), które są następnie zliczane w liczniku CTU. Zliczanie jest wykonywane w czasie aktywnego wejścia IN. Po osiągnięciu żądanej liczby interwałów ustawiane jest wyjście Q. Kasowanie układu odbywa się z wykorzystaniem sygnału R.



Rys. 8. Realizacja akcji czasowej w makromiejscu z historią
Fig. 8. An implementation of the time action inside a macroplace with history



Rys. 9. Realizacja układu czasowego z pamięcią
Fig. 9. An implementation of a timer with a memory

Implementacja złożonego modelu systemu sterowania, przygotowanego z wykorzystaniem hierarchicznej sieci Petriego, wymaga odpowiedniego złożenia przedstawionych w artykule elementów realizacyjnych.

4. Podsumowanie

W artykule zaproponowano metodę realizacji hierarchicznej sieci Petriego HPN w sterownikach klasy PLC, z wykorzystaniem graficznego języka drabinkowego (LD). Metodę przedstawiono na szeregu bardzo prostych przykładów, opisujących charakterystyczne elementy opisu, wykorzystywane w modelu HPN. Prostota metody może być przyczynkiem do jej wykorzystania przy realizacji systemów sterowania cyfrowego, w których odnaleźć można takie aspekty sterowania jak: zależności czasowe, współbieżność procesów, hierarchia czy obsługa sytuacji wyjątkowych.

5. Literatura

[1] Andrzejewski G.: Programowy model interpretowanej sieci Petriego dla potrzeb projektowania mikrosystemów cyfrowych, Zielona Góra, Oficyna Wydawnicza Uniwersytetu Zielonogórskiego, 2003.
[2] Andrzejewski G.: Hierarchical Petri nets for digital controller design, in Design of embedded control systems, New York, Springer, 2005 pp. 27-36.
[3] Andrzejewski G., Skowroński Z.: Zrównoleglanie algorytmów sterowania w systemach klasy PLC, Pomiary Automatyka Kontrola, 2006, nr 6, wyd. spec., s. 62-64.
[4] IEC 61131 Programmable Controllers, New York, 1993.

Artykuł recenzowany