IMPLEMENTAÇÃO DE CONTROLADORES A EVENTOS DISCRETOS USANDO DIAGRAMA LADDER COM MÓDULO SINCRONIZANTE

Gustavo da Silva Viana*, Marcos Vicente Moreira*, João Carlos Basilio*

* *Universidade Federal do Rio de Janeiro*
COPPE - Programa de Engenharia Elétrica
21949-900, Rio de Janeiro, R.J., Brasil

Emails: gustavo.viana@poli.ufrj.br, moreira.mv@poli.ufrj.br, basilio@poli.ufrj.br

Abstract— We propose in this paper a method for discrete-event controller implementation using ladder diagrams that is capable to synchronize all the component behaviors of a discrete-event system modeled with control interpreted Petri nets and, therefore, avoids the need to previously find the parallel composition of the Petri net models of the individual elements, which would lead to a considerably larger ladder diagram. In order to do so, we extend a previously proposed method that converts control interpreted Petri nets in ladder diagrams by adding a synchronization module in such a way that the resulting ladder diagram obtained from the method proposed here executes, at each event observation, a synchronous composition of the modules that form the system.

Keywords— Discrete event systems, discrete-event controller, control interpreted Petri net.

Resumo— Neste artigo propomos um método de implementação de controladores a eventos discretos usando diagramas ladder que é capaz de sincronizar todos os comportamentos dos componentes do sistema modelados por redes de Petri interpretadas para controle, evitando, portanto, que seja necessário realizar a composição paralela dos modelos individuais de cada componente, o que levaria a um diagrama ladder consideravelmente maior. Para tanto, estendemos um método previamente proposto na literatura que converte redes de Petri interpretadas para controle em diagramas ladder adicionando-se um módulo de sincronização, de tal sorte que o diagrama ladder obtido utilizando-se o método proposto neste trabalho passa a executar, a cada observação de um evento, uma composição síncrona dos módulos que compõem o sistema.

Palavras-chave— Sistemas a eventos discretos, controlador a eventos discretos, redes de Petri interpretadas para controle.

1 Introdução

Redes de Petri (Peterson, 1981; Murata, 1989; Cassandras and Lafortune, 2008; David and Alla, 2005) podem ser utilizadas tanto para descrever o comportamento desejado de um sistema de automação quanto para descrever o modelo do controlador a eventos discreto que leva ao comportamento desejado em malha fechada. Na prática, contudo, não se implementa o comportamento desejado do sistema realimentado mas, sim, o controlador a eventos discretos. Em geral, controladores a eventos discretos são temporizados e sincronizados com eventos externos. Dessa forma, estruturas formais devem ser adicionadas às redes de Petri para lidar com esses requisitos. Tem-se, portanto, as chamadas redes de Petri interpretadas para controle (RPIC) (David and Alla, 1995; Uzam and Jones, 1998; Uzam et al., 1996; Jimenez et al., 2001; Venkatesh et al., 1994).

Segundo Holloway et al. (1997), as três abordagens principais para o projeto de controladores a eventos discretos são: (*i*) abordagem usando a teoria de controle (Cassandras and Lafortune, 2008; Iordache and Antsaklis, 2006); (*ii*) projeto direto do controlador lógico (Zhou, DiCesare and Rudolph, 1992; David, 1995) e; (*iii*) projeto do controlador a partir do comportamento controlado desejado do sistema (Jeng and DiCesare, 1993; Suzuki and Murata, 1983; Zhou, DiCesare and Desrochers, 1992).

O projeto do controlador a eventos discretos a partir do comportamento desejado do sistema é a técnica mais comumente adotada quando se utiliza modelos de redes de Petri para sistemas de manufatura. Essa abordagem consiste na modelagem do comportamento desejado do sistema, ou seja, da planta e do controlador juntos em malha fechada. Uma vez obtido o comportamento desejado, o controlador deve, então, ser extraído, isto é, deve ser obtida uma rede de Petri que modela o controlador para implementação, em geral, em um controlador lógico programável (CLP). Entre as linguagens existentes para a programação de CLPs (IEC 1131-3, 1993), o diagrama ladder é o mais utilizado na indústria.

Entre os métodos de obtenção do comportamento controlado desejado do sistema está a técnica de bottom-up, que consiste na obtenção do comportamento completo do sistema a partir do comportamento desejado de cada um de seus componentes. Para tanto, redes de Petri que descrevem o comportamento dos componentes do sistema são inicialmente obtidas, e uma composição é realizada para obtenção do comportamento desejado completo. Essa composição pode ser feita mesclando-se e criando-se alguns lugares e/ou transições para interligar os componentes (Cassandras and Lafortune, 2008; David and Alla, 2005). Contudo, essa forma de composição

entre subsistemas não constitui um método sistemático e a solução é obtida caso a caso. Uma outra forma de composição seria realizar a composição paralela entre as redes de Petri dos sistemas individuais (Iordache and Antsaklis, 2006). Contudo, esse procedimento, em geral, leva a um aumento significativo no número de transições da rede composta, incluindo até mesmo transições que nunca ficarão habilitadas e, portanto, nunca irão disparar.

Em um trabalho recente, Moreira and Basilio (2013) propõem um método para a construção do diagrama ladder diretamente a partir da rede de Petri interpretada para controle que modela a dinâmica do controlador a eventos discretos. O diagrama ladder consiste dos seguintes módulos: *(i)* módulo de inicialização; *(ii)* módulo dos eventos; *(iii)* módulo das condições para o disparo das transições; *(iv)* módulo da dinâmica; *(iv)* módulo das ações. Contudo, o método proposto por Moreira and Basilio (2013) exige que as composições paralelas entre os elementos individuais do controlador sejam executadas a priori. Neste artigo, estendemos o método proposto por Moreira and Basilio (2013) adicionando-se um módulo de sincronização e alterando-se o módulo das condições para o disparo das transições de tal sorte que o novo diagrama ladder seja capaz de executar a cada observação de um evento, uma composição síncrona dos módulos que compõem o sistema, isto é, permitindo que os eventos comuns somente ocorram quando ambas as transições de cada um dos modelos associadas a esse evento estejam habilitadas.

Este artigo está estruturado da seguinte forma. Na seção 2 apresenta-se uma breve revisão de redes de Petri com o objetivo de definir redes de Petri interpretadas para controle, que é o formalismo utilizado para descrever a dinâmica do controlador a eventos discretos. Na seção 3, o método de conversão de RPIC em diagrama ladder para implementação em um CLP, apresentado em Moreira and Basilio (2013), é estendido, sendo propostas algumas alterações para permitir a sincronização dos componentes do sistema. Finalmente, na seção 4, são apresentadas as conclusões. Um exemplo é utilizado ao longo do texto para ilustrar todos os passos do método proposto neste artigo.

2 Fundamentos teóricos: redes de Petri interpretadas para controle

Antes de apresentarmos a definição de redes de Petri, vamos, inicialmente, apresentar a definição de grafo de uma rede de Petri.

Definição 1 *(Grafo de uma rede de Petri) Um grafo de uma rede de Petri é um grafo bipartido ponderado* $(P, T, Pre, Post)$*, em que* P *é um conjunto finito de lugares,* T *é um conjunto finito de transições,* $Pre : (P \times T) \to \mathbb{N}$ *é a função dos arcos ordinários que ligam lugares a transições,* $Post : (T \times P) \to \mathbb{N}$ *é a função dos arcos ordinários que ligam transições a lugares.*

Para indicar quando as condições para o disparo das transições são satisfeitas, atribui-se fichas aos lugares da rede. O número de fichas atribuídas a um lugar p_i é representado por $x(p_i) \in \mathbb{N}$. Uma marcação de uma rede de Petri é o vetor coluna $\underline{x} = [x(p_1)\ x(p_2)\ \ldots\ x(p_n)]^T$, formado pelo número de fichas em cada lugar p_i, para $i = 1, \ldots, n$, em que n denota a cardinalidade de P.

Podemos, agora, definir formalmente redes de Petri.

Definição 2 *(Redes de Petri) Uma rede de Petri* N *é uma quíntupla* $N = (P, T, Pre, Post, \underline{x}_0)$*, em que* $(P, T, Pre, Post)$ *é, de acordo com a definição 1, o grafo de uma rede de Petri e* $\underline{x}_0$ *é a marcação inicial do conjunto de lugares.*

Uma extensão das redes de Petri é feita acrescentando-se um novo tipo de arco: o arco inibidor. Os arcos inibidores são representados por um arco cuja extremidade final possui um círculo. Um arco inibidor de peso y, ligando um lugar p_i a uma transição t_j, habilita t_j quando o número de fichas em p_i é inferior ao peso y e, quando t_j dispara, o número de fichas de p_i permanece inalterado. Uma rede de Petri com arcos inibidores é denominada de rede de Petri estendida. Podemos, então, apresentar a seguinte definição.

Definição 3 *(Rede de Petri estendida) Uma rede de Petri estendida é uma sêxtupla* $N_e = (P, T, Pre, Post, In, \underline{x}_0)$*, em que* P*,* T*,* Pre*,* $Post$ *e* $\underline{x}_0$ *são definidos da mesma forma que na definição 2, e* $In : (P \times T) \to \mathbb{N}$ *é a função dos arcos inibidores que ligam lugares a transições.*

As redes de Petri estendidas são mais apropriadas para descrever o comportmanento desejado do sistema. Contudo, para modelar o comportamento do controlador a eventos discretos, torna-se necessário adicionar à rede de Petri estruturas capazes de lidar com sensores e atuadores. Assim, os sinais enviados pelos sensores e que são destinados a informar a ocorrência de eventos, assim como as condições da planta, são associados às transições, e as ações impulsionais ou operações, como, por exemplo, a contagem de peças em um sistema de automação, ou ações de nível comandadas pelo controlador são associadas aos lugares. Além disso, deve-se incluir possíveis atrasos de disparo das transições, ou seja, o conjunto de transições será particionado da seguinte forma: $T = T_0 \dot{\cup} T_D$, em que T_0 denota o conjunto de transições sem atraso de disparo e T_D denota o conjunto de transições com atraso de disparo. A rede

de Petri que inclui todas essas estruturas é denominada rede de Petri interpretada para controle, sendo formalmente definida da seguinte forma.

Definição 4 *(Rede de Petri interpretada para controle) Uma rede de Petri interpretada para controle é dada por* $N_C = (P, T, Pre, Post, In, \underline{x}_0, C, E, l, D, l_D, A_I, A_N, l_A)$ *em que* $(P, T, Pre, Post, In, w, \underline{x}_0)$ *é uma rede de Petri estendida segura,* C *e* E*, são os conjuntos de condições e eventos externos associados às transições não temporizadas pertencentes a* T_0*,* $l : T_0 \rightarrow (E \times C)$ *é a função que associa a cada transição não temporizada um evento e uma condição para o disparo da transição,* D *é o conjunto de atrasos de disparo associados às transições temporizadas* T_D*,* $l_D : T_D \rightarrow D$ *é a função que associa a cada transição de* T_D *um atraso do conjunto* D*,* A_I *denota o conjunto de ações impulsionais e de operações,* A_N *denota o conjunto de ações de nível, e* $l_A : P \rightarrow (A_I \cup A_N)$ *a função que associa aos lugares da rede uma ação pertencente a* $A_I \cup A_N$*.* □

Note que na definição 4 acima, supõe-se que cada transição $t_j \in T_0$ está associada a uma condição e a um evento. Se a condição associada a t_j não for explicitamente especificada, então essa condição será igual a um, *i. e.*, a condição lógica será verdadeira. Além disso, se o evento externo de entrada e_j não for especificado, então t_j será igual ao evento sempre ocorrente λ (David and Alla, 2005), utilizado para indicar que a transição t_j deve disparar assim que habilitada, se a condição associada c_j for verdadeira. Ainda de acordo com a definição 4, ações impulsionais, operações e ações de nível são associadas aos lugares. Contudo, em alguns casos, lugares podem não possuir ações ou operações associadas, ou seja, o lugar representa apenas um estado parcial do sistema.

3 Implementação do controlador a eventos discretos modelado por uma RPIC utilizando diagramas ladder particionados em módulos

A obtenção da RPIC que descreve o comportamento completo do sistema a partir de cada um de seus subsistemas, também descritos por RPICs, requer a realização de uma operação de composição. Sejam N_{C_1} e N_{C_2} duas redes de Petri interpretadas para controle. Então, a operação de composição, denominada aqui de composição paralela entre RPICs, deve gerar uma rede de Petri N_C tal que N_C contém: (i) as transições de N_{C_1} rotuladas pelo evento sempre ocorrente λ, as transições temporizadas de N_{C_1}, e as transições de N_{C_1} rotuladas por eventos que não pertecem a N_{C_2}; (ii) as transições de N_{C_2} rotuladas pelo evento sempre ocorrente λ, as transições temporizadas de N_{C_2}, e as transições de N_{C_2} rotuladas por eventos que não pertencem a N_{C_1}; (iii) as transições que modelam a sincronização de transições rotuladas por eventos em comum de N_{C_1} e N_{C_2}. É importante ressaltar ainda que a ocorrência de um evento σ que rotula transições de N_{C_1} e N_{C_2} somente poderá levar a uma evolução dos seus estados, se existir pelo menos uma transição rotulada por σ habilitada em cada uma das RPICs, N_{C_i}, para $i = 1, 2$, e se as condições associadas a essas transições são ambas verdadeiras, permitindo o disparo simultâneo das transições.

Um algoritmo para obtenção da composição paralela entre duas RPICs pode ser obtido a partir do algoritmo apresentado em Iordache and Antsaklis (2006) para composição síncrona entre duas redes de Petri rotuladas. Contudo, a composição entre redes de Petri interpretadas para controle leva, em geral, a um número excessivamente grande de transições, sendo que algumas nunca poderão disparar. Isso resultaria em um diagrama ladder excessivamente grande e, acima de tudo, com linhas que nunca seriam disparadas.

Um método para a obtenção de diagramas ladder para implementação de controladores a eventos discretos modelados por redes de Petri interpretadas para controle foi proposto por Moreira and Basilio (2013). Entre as vantagens desse método estão a sua capacidade de evitar o chamado efeito avalanche, além de fornecer um diagrama ladder que permite tanto uma fácil visualização da estrutura da rede de Petri quanto o acompanhamento da evolução das fichas ao longo da rede à medida que os eventos ocorrem. O método proposto por Moreira and Basilio (2013) consiste na divisão do código de programação em cinco módulos, quais sejam: *(i)* módulo de inicialização; *(ii)* módulo dos eventos; *(iii)* módulo das condições para o disparo das transições; *(iv)* módulo da dinâmica; *(iv)* módulo das ações. Essa estrutura garante que cada estado da rede de Petri permanece inalterado por pelo menos um ciclo de varredura.

Neste artigo, iremos estender o método proposto por Moreira and Basilio (2013) para permitir que a sincronização entre os componentes do sistema seja feita na etapa de implementação do controlador, evitando, assim, a necessidade de se calcular, a priori, a composição paralela entre as RPICs. Isso será feito da seguinte forma:

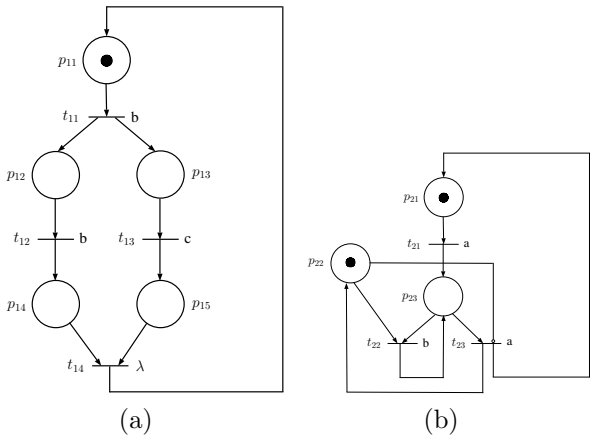
1. Inicialmente será criado um módulo de sincronização para assegurar que as transições de RPICS diferentes associadas a um mesmo evento somente dispare quando estiverem habilitadas em ambas as RPICs.

2. Em seguida, o módulo das condições para o disparo das transições será alterado no sentido de garantir que as linhas associadas aos eventos comuns somente fiquem energizadas quando as condições de sincronização se tornem verdadeiras no módulo de sincronização.

Como resultado, o diagrama ladder estendido terá a seguinte estrutura:

- Módulo de inicialização;
- Módulo de eventos;
- Módulo de sincronização;
- Módulo das condições para o disparo das transições;
- Módulo da dinâmica;
- Módulo das ações.

Nas subseções a seguir, iremos detalhar como cada um dos seis módulos acima pode ser obtido a partir das RPICs de vários elementos individuais. Para tornar mais clara a construção do diagrama ladder, usaremos ao longo do texto o sistema composto de dois subsistemas modelados pelas RPICs N_{C_1} e N_{C_2} mostradas nas figuras 1(a) e (b). Iremos supor que $E_1 = \{b, c, \lambda\}$ e $E_2 = \{a, b\}$. Assim, b será o único evento em comum entre N_{C_1} e N_{C_2} e, portanto, os disparos das transições rotuladas por b devem ser sincronizados na ocorrência desse evento. Iremos supor ainda que os eventos a, b e c são eventos externos associados à borda de subida dos respectivos sinais dos sensores.



(a) (b)

Figura 1: Redes de Petri interpretadas para contole: N_{C_1} (a) e N_{C_2} (b).

3.1 Módulo de inicialização

O primeiro módulo é formado por uma única linha que contém um contato normalmente fechado (NF) associado a uma variável binária interna $B0$ que, no primeiro ciclo de varredura, energiza logicamente as bobinas associadas aos lugares que possuem uma ficha no estado inicial. Após o primeiro ciclo de varredura, o contato NF é aberto e não é mais fechado. Na figura 2 é mostrado o módulo de inicialização para as RPICs da figura 1.

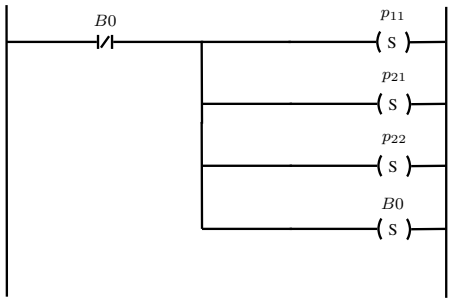


Figura 2: Módulo de inicialização para o ladder sincronizante das RPICs N_{C_1} e N_{C_2}.

3.2 Módulo de eventos

O módulo de eventos possui um número de linhas igual ao número de eventos externos do sistema. Eventos da planta são associados à borda de subida ou de descida de sinais de sensores. Para detectar essas transições de nível de variáveis binárias, alguns CLPs possuem contatos especiais que são fechados quando a expressão booleana anterior ao contato, na mesma linha de programação, muda de falsa para verdadeira, e mantém o contato fechado por apenas um ciclo de varredura. Em alguns CLPs essa instrução recebe o nome de *one shot rising* (OSR). Neste artigo, esse tipo de contato é utilizado para observar a ocorrência de um evento, como mostrado na figura 3, em que há três eventos a serem observados.

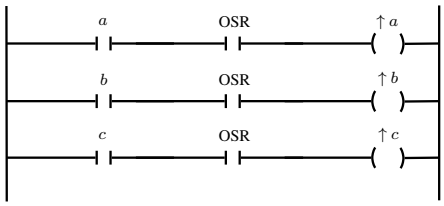


Figura 3: Módulo de eventos para o ladder sincronizante das RPICs N_{C_1} e N_{C_2}.

3.3 Módulo de sincronização

O módulo de sincronização serve para verificar se um dado evento comum a duas ou mais redes de Petri pode ocasionar o disparo de transições nessas redes. Para tanto, cada subsistema precisa ser verificado separadamente. Isso é feito colocando-se em paralelo as condições de habilitação de todas as transições que estão associadas ao evento que se deseja verificar de um dado subsistema. Caso uma dessas condições seja verdadeira, uma variável binária recebe o valor um para indicar que o evento pode ocorrer naquele subsistema. É importante ressaltar que as condições c_j, associadas às transições $t_j \in T_0$, também devem ser implementadas conectadas em série com as condições de habilitação das transições. Isso se deve ao fato

de que uma transição t_j é disparável se e somente se é habilitada pela marcação de seus lugares de entrada e a condição associada c_j for verdadeira. Esse módulo possui, no pior caso, o número de linhas igual ao número de subsistemas multiplicado pelo seu número de eventos externos.

Na figura 4 é mostrado o módulo de sincronização para o único evento em comum das redes da figura 1. Note que a primeira linha do módulo de

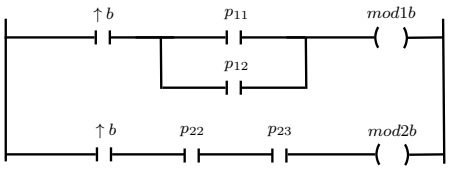


Figura 4: Módulo de sincronização para o ladder sincronizante das RPICs N_{C_1} e N_{C_2}.

sincronização assegura que as transições associadas ao evento b somente poderão disparar quando um dos lugares, p_{11} ou p_{12}, tenha ficha, e a segunda linha impõe a necessidade conjunta de que os lugares p_{22} e p_{23} também tenham fichas.

3.4 Módulo das condições para o disparo das transições

Nesse módulo são verificadas as condições para o disparo das transições e cada linha de programação está associada a uma única transição. No caso do evento externo estar associado a apenas um único módulo, as condições são obtidas pela associação em série de contatos normalmente abertos, representando a conexão aos lugares de entrada da transição por arcos ordinários, e de contatos normalmente fechados, representando a ligação aos lugares de entrada da transição por arcos inibidores.

Quando o evento externo está associado a mais de um subsistema, como é o caso do evento b na figura 1, contatos normalmente abertos devem ser inseridos em série para verificar se o evento pode ocorrer em todos os subsistemas para os quais é definido.

Na figura 5, o módulo das condições para o exemplo da figura 1 é mostrado. Note que os contatos rotulados por *mod1b* e *mod2b*, inseridos em série nas duas primeiras linhas e na penúltima linha do diagrama ladder da figura 5, asseguram o disparo síncrono das transições t_{11} e t_{22} ou t_{12} e t_{22}, dependendo dos lugares p_{11} ou p_{12} terem ou não fichas.

3.5 Módulo da dinâmica da RPIC

O módulo da dinâmica fornece a evolução das fichas quando uma transição dispara. Como primeiramente a habilitação das transições é verificada no módulo das condições, para depois as

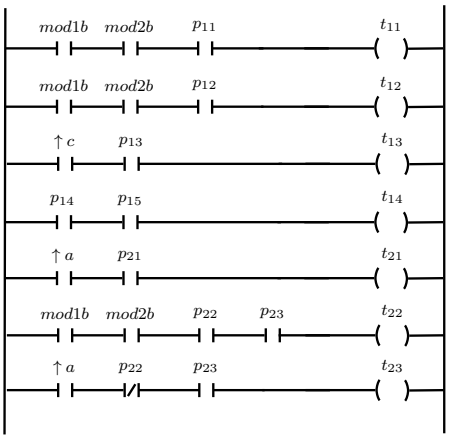


Figura 5: Módulo das condições para o disparo das transições para o ladder sincronizante das RPICs N_{C_1} e N_{C_2}.

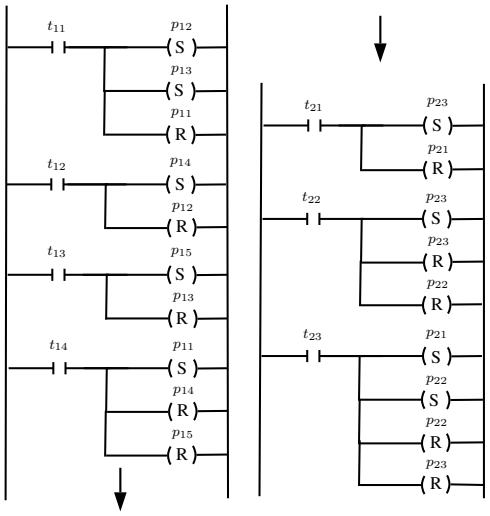


Figura 6: Módulo da dinâmica para o ladder sincronizante das RPICs N_{C_1} e N_{C_2}.

transições serem disparadas no módulo da dinâmica, o disparo de uma transição não é capaz de habilitar uma outra transição e provocar o seu disparo em um mesmo ciclo de varredura. Isso evita o chamado efeito avalanche (Fabian and Hellgren, 1998). O módulo da dinâmica da RPIC possui o número de linhas igual a soma do número de transições dos subsistemas. Na figura 7 é mostrado o módulo da dinâmica dos subsistemas da figura 1.

3.6 Módulo das ações

No último módulo, as ações são definidas. Quando um lugar possui uma ação impulsional associada a ele, um contato OSR é utilizado para energizar uma bobina de saída por apenas um ciclo de varredura, e quando um lugar possui uma ação de nível associada, uma bobina normal é utilizada. Na figura 7 o módulo de ações para os subsistemas da

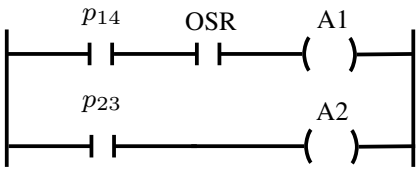


Figura 7: Módulo das ações para o ladder sincronizante das RPICs N_{C_1} e N_{C_2}.

figura 1 é mostrado supondo-se que existem apenas uma ação impulsional e uma ação de nível associadas aos lugares p_{14} e p_{23}, respectivamente.

4 Conclusão

Neste artigo apresentamos uma maneira sistemática de efetuar a sincronização de sistemas modelados por RPICs diretamente no diagrama ladder sem a necessidade de se efetuar, a priori, as composições paralelas das redes individuais e que representa uma extensão do método proposto em Moreira and Basilio (2013). A metodologia aqui proposta pode ser utilizada com vantagens sobre aquela proposta em Moreira and Basilio (2013) quando o sistema for formado por vários componentes individuais.

Referências

Cassandras, C. G. and Lafortune, S. (2008). *Introduction to Discrete Events Systems*, 2nd edn, Springer, New York, NY : USA.

David, R. (1995). Grafcet: a powerful tool for specification of logic controllers, *IEEE Transactions on Control Systems Technology* **3**: 253–268.

David, R. and Alla, H. (1995). Petri nets for modeling of dynamic systems - a survey, *Automatica* **30**: 175–202.

David, R. and Alla, H. (2005). *Discrete, Continuous and Hybrid Petri Nets*, Springer.

Fabian, M. and Hellgren, A. (1998). PLC-based implementation of supervisory control for discrete event systems, *Proceedings of the 37th IEEE Conference on Decision and Control*, pp. 3305–3310.

Holloway, L. E., Krogh, B. H. and Giua, A. (1997). A survey of petri net methods for controlled discrete event systems, *Discrete Event Dynamic Systems: Theory and Applications* **7**: 151–190.

Iordache, M. V. and Antsaklis, P. J. (2006). *Supervisory Control of Concurrent Systems: A Petri Net Structural Approach*, Birkhauser, Boston : USA.

Jeng, M. D. and DiCesare, F. (1993). A review of synthesis techniques for petri nets with applications to automated manufacturing systems, *IEEE Transactions on Systems, Man, and Cybernetics* **23**: 301–312.

Jimenez, I., lopez, E. and Ramirez, A. (2001). Synthesis of Ladder diagrams from Petri nets controller models, *Proceedings of the 2001 IEEE International Symposium on Intelligent Control*, pp. 225–230.

IEC 1131-3 (1993). *Programmable controllers - Part 3: Programming Languages*, International Electrotechnical Commission.

Moreira, M. V. and Basilio, J. C. (2013). Bridging the gap between design and implementation of discrete event controllers, *IEEE Transactions on Automation Science and Engineering* (to appear) .

Murata, T. (1989). Petri nets: Properties, analysis and applications, *Proceedings of the IEEE* **77**(4): 541–580.

Peterson, J. L. (1981). *Petri net theory and the modeling of systems*, Prentice Hall, Upper Saddle River, NJ.

Suzuki, I. and Murata, T. (1983). A method for stepwise refinement and abstraction of petri nets, *Journal of Computer and System Sciences* **27**: 51–76.

Uzam, M. and Jones, A. H. (1998). Discrete event control system design using automation Petri nets and their Ladder diagram implementation, *Int J Adv Manuf Technol* **14**: 716–728.

Uzam, M., Jones, A. H. and Ajlouni, N. (1996). Conversion of Petri nets controllers for manufacturing systems into Ladder logic diagrams, *IEEE Conference on Emerging Technologies and Factory Automation*, pp. 649–655.

Venkatesh, K., Zhou, M. and Caudill, R. J. (1994). Comparing ladder logic diagrams and Petri nets for sequence controller design through a discrete manufacturing system, *IEEE Transactions on Industrial Electronics* **41**: 611–619.

Zhou, M. C., DiCesare, F. and Desrochers, A. (1992). Hybrid methodology for the synthesis of petri net models for manufacturing systems, *IEEE Transactions on Robotics and Automation* **8**: 350–361.

Zhou, M. C., DiCesare, F. and Rudolph, D. L. (1992). Design and implementation of a petri net based supervisor for a flexible manufacturing system, *Automatica* **28**: 1199–1208.