UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
CÂMPUS CORNÉLIO PROCÓPIO
DIRETORIA DE PESQUISA E PÓS-GRADUAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA
MESTRADO EM ENGENHARIA ELÉTRICA

HUGO VINICIUS DIAS SILVA

METODOLOGIA DE PROJETO DE AUTOMAÇÃO INDUSTRIAL VISANDO A CONVERSÃO AUTOMÁTICA DE REDES DE PETRI INTERPRETADAS EM CÓDIGOS IMPLEMENTÁVEIS

DISSERTAÇÃO

CORNÉLIO PROCÓPIO
2013

HUGO VINICIUS DIAS SILVA

METODOLOGIA DE PROJETO DE AUTOMAÇÃO INDUSTRIAL VISANDO A CONVERSÃO AUTOMÁTICA DE REDES DE PETRI INTERPRETADAS EM CÓDIGOS IMPLEMENTÁVEIS

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Tecnológica Federal do Paraná como requisito parcial para obtenção do título de “Mestre em Engenharia Elétrica”.

Orientador: Prof. Dr. Marcos Banheti Rabello Vallim

CORNÉLIO PROCÓPIO
2013

Dados Internacionais de Catalogação na Publicação

S586 Silva, Hugo Vinícius Dias
Metodologia de projeto de automação industrial visando a conversão automática de redes de Petri interpretadas em códigos implementáveis / Hugo Vinícius Dias Silva. – 2013.
62 p. : il. ; 30 cm

Orientador: Prof. Dr. Marcos Banheti Rabello Vallim.
Dissertação (Mestrado) – Universidade Tecnológica Federal do Paraná. Programa de Pós-graduação em Engenharia Elétrica. Cornélio Procópio, 2013.
Bibliografia: p. 59-62.

1. Sistemas não-lineares. 2. Automação industrial. 3. Petri, Redes de. 4. Engenharia elétrica – Dissertações. I. Vallim, Marcos Banheti Rabello, orient. II. Universidade Tecnológica Federal do Paraná. Programa de Pós-graduação em Engenharia Elétrica. III. Título.

CDD (22. ed.) 621.3

Biblioteca da UTFPR, Câmpus Cornélio Procópio

Minstério da Educação
Universidade Tecnológica Federal do Paraná
Câmpus Cornélio Procópio
Diretoria de Pesquisa e Pós-Graduação
Programa de Pós-Graduação em Engenharia Elétrica
Mestrado em Engenharia Elétrica

TERMO DE APROVAÇÃO

Metodologia de projeto de automação industrial visando a conversão automática de redes de petri interpretadas em códigos implementáveis

por

Hugo Vinicius Dias Silva

Esta Dissertação foi julgada adequada para obtenção do Título de “Mestre em Engenharia Elétrica” e aprovado em sua forma final pelo Programa de Pós-Graduação em Engenharia Elétrica da Universidade Tecnológica Federal do Paraná.
Cornélio Procópio, 08/04/2013.

Prof. Dr. Alessandro Goedtel,
Coordenador do Curso

Banca Examinadora:

Prof. Dr. Marcos Banheti Rabello Vallim,
Orientador

Prof. Dr. Bruno Augusto Angélico,
Universidade Tecnológica Federal do Paraná - UTFPR

Prof. Dr. André Bittencourt Leal,
Universidade do Estado de Santa Catarina - UDESC

“A Folha de Aprovação assinada encontra-se na Coordenação do Curso do Programa”

Dedico este trabalho aos meus pais, Iraci e Salvador pessoas fundamentais na minha formação.

à minha filha Alice Dias,

à minha esposa Natália Cristina,

à minha irmã Bruna Dias.

AGRADECIMENTOS

À Deus pela ajuda nos momentos importantes.

À minha família pelo apoio e compreensão nos períodos de ausência.

Ao meu orientador e amigo Prof. Dr. Marcos Banheti Rabello Vallim pelas orientações acadêmicas e conselhos pessoais.

Aos companheiros estudandes do Programa de Pós-Graduação em Engenharia Elétrica da UTFPR, em especial aos alunos Wyllian Salviano, Marcelo Pedroso, Thiago Henrique, Felipe Postali, Celso Kawamura, Leonardo Campanhol, Paulo Broniera entre tantos outros.

Aos estudantes que participam do Projeto “Ninho de Pardais” pelo apoio e a amizade durante a minha passagem neste brilhante projeto.

Aos professores que ajudaram com esta pesquisa, em especial ao Prof. Dr. Bruno Angélico, Prof. Dr. José Augusto Fabri e Prof. Dr. Rodrigo Sumar.

Ao coordenador do PPGEE Prof. Dr. Alessandro Goedtel pela ajuda e orientações acadêmicas.

À UTFPR pelo suporte acadêmico e tecnológico.

À Capes pelo apoio financeiro concedido através da bolsa de mestrado.

A mente que se abre a uma nova ideia
jamais voltará ao seu tamanho original.

(Albert Einstein)

RESUMO

SILVA, Hugo Vinicius Dias. **Metodologia de projeto de automação industrial visando a conversão automática de redes de petri interpretadas em códigos implementáveis**. 2013. 70 f. Dissertação – Programa de Pós-Graduação em Engenharia Elétrica, Universidade Tecnológica Federal do Paraná. Cornélio Procópio, 2013.

O objetivo deste trabalho é apresentar uma proposta de metodologia para o desenvolvimento de projetos de automação industrial com vistas a agregar as vantagens da utilização dos formalismos para representação dos Sistemas a Eventos Discretos com a necessidade de implementar os algoritmos utilizados em equipamentos de controle industrial. Tal metodologia propõe um conjunto de etapas para a construção do algoritmo de controle, baseando-se na Rede de Petri Interpretada utilizada para representação da planta controlada e da lógica de controle. Através desta metodologia é possível realizar o controle de um sistema de automação industrial garantindo que a lógica de controle não possua erros quando for executada. A utilização da metodologia foi exemplificada aplicado-a ao projeto de automação de dois casos práticos, onde o primeiro caso é utilizado para detalhar as etapas contidas na metodologia, já o segundo projeto prático possui uma especificação de funcionamento mais complexa, corroborando a utilização dos métodos formais para garantir a corretude lógica do processo. Buscando evidenciar a independência da metodologia frente à tecnologia utilizada para controlar tais processos, os algoritmos resultantes da execução da metodologia foram implementados em CLP e FPGA, sendo que a utilização de equipamentos de controle com características paralelas aponta uma tendência dos projetos de automação industrial.

Palavras-chave: Sistemas a Eventos Discretos. Projetos de Automação Industrial. Redes de Petri Interpretadas.

ABSTRACT

SILVA, Hugo Vinicius Dias. **Design methodology targeting industrial automation automatic conversion of Interpreted Petri Nets code implementable.** 2013. 70 f. Dissertação – Programa de Pós-Graduação em Engenharia Elétrica, Universidade Tecnológica Federal do Paraná. Cornélio Procópio, 2013.

The objective of this work is to present a proposed methodology for the development of industrial automation projects by aggregating the advantages of using formalisms for representing Discrete Event Systems with the need to implement algorithms considered in industrial control equipment. This methodology proposes a set of steps for the construction of the control algorithm, based on Interpreted Petri Net used to represent the plant to be controlled and the control logic. Through this method it is possible to control an industrial automation system ensuring that the control logic contains no errors when executed. The use of the methodology was exemplified by automating two study cases, where the former is used to detail the steps in the methodology, and the second presents a practical function with more complex specification, corroborating the use of formal methods to ensure the correctness of the process logic. In order to demonstrate the independence of the methodology regarding to the technology used to control such processes, algorithms resulting from the application of the methodology were implemented in PLC and FPGA, and the use of control equipment with parallel features shows a trend of industrial automation projects.

Keywords: Discrete Event Systems. Project Industrial Automation. Interpreted Petri Nets.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE QUADROS

LISTA DE SIGLAS

DAQ	*Data aquisition*.
EEPROM	*Electrically Erasable Programmable Read-Only Memory*.
FPGA	Field-programmable gate array.
IEEE	*Institute of Electrical and Electronics Engineers*.
IOPT	*Input/Output Place/Transition*.
ITS PLC	*Interactive Training System for Programmable Logic Controller*.
RdPIs	Redes de Petri Interpretadas.
RdPs	Redes de Petri.
RTL	*Register Transfer Level*.
SEDs	Sistemas a Eventos Discretos.
SFC	*Sequential Function Chart*.
SRAM	*Static Random Access Memory*.
VHDL	*Very High-Speed Integrated Circuit Hardware Description Language*.

LISTA DE SÍMBOLOS

P	conjunto de lugares
T	conjunto de transições
M	marcação da Rede de Petri
$\mathbb{Z}^+$	conjunto dos numeros inteiros e positivos
Σ	alfabeto dos eventos de entrada
Φ	alfabeto dos eventos de saída
$\mathcal{L}(Q,M_0)$	linguagem
M_0	marcação inicial
$\mathcal{L}_{in}(Q,M_0)$	linguagem das entradas
$\mathcal{L}_{out}(Q,M_0)$	linguagem das saídas
$\cap$	interseção
$\emptyset$	conjunto vazio
$\mathbb{N}$	conjunto dos números naturais
$\subseteq$	subconjunto
$\rightarrow$	aplicação. Ex: $f : x \rightarrow y$, f é uma aplicação de x em y
$\forall$	para todo
$\in$	pertence

SUMÁRIO

1 INTRODUÇÃO

A otimização dos processos e a busca pelo aumento da produção em sistemas de automação industrial geram um aumento expressivo na complexidade deste sistemas, devido ao alto grau de integração entre os processos e a expansão dos arranjos produtivos já existentes. Devido a este aumento na complexidade, surge a necessidade da utilização de ferramentas para análise e validação das estratégias de controle em tempo de projeto (CASSANDRAS; LAFORTUNE, 2008).

Vários métodos formais são utilizados para análise e simulação de sistemas de automação. Tais sistemas possuem a característica de evoluir com a ocorrência de eventos discretos e assíncronos no tempo, logo são classificados como Sistemas a Eventos Discretos (SEDs) (CASSANDRAS; LAFORTUNE, 2008). Um formalismo largamente utilizado para a representação dos SEDs são as Redes de Petri (RdPs), devido ao seu alto poder de representação de processos com dinâmicas paralelas, característica fortemente presente nos sistemas de automação industriais (MURATA, 1989; CARDOSO; VALETTE, 1997).

Porém, a utilização dos métodos formais esbarra na dificuldade de transcrever as propriedades presentes no modelo para os equipamentos que realizaram o controle destes processos, diminuindo a aplicabilidade dos métodos de análise nos ambientes industriais (IORDACHE; ANTSAKLIS, 2009). A ausência da validação das rotinas de controle dos SEDs proporciona inúmeros prejuízos, onde o principal problema é a necessidade de realizar a implementação da rotina de controle e a sua execução, para só depois identificar e corrigir possíveis erros.

1.1 Objetivos

Os objetivos deste trabalho são divididos em objetivo geral e objetivo específicos.

1.1.1 Objetivo Geral

O objetivo deste trabalho é contribuir para o desenvolvimento dos projetos de automação industrial apresentando uma metodologia que visa, através de etapas bem definidas, à síntese de algoritmos de controle legíveis e reutilizáveis. Metodologia que aborda todas as etapas para construção destes algoritmos de controle, buscando definir e propor melhorias para a forma que se executa os projetos de automação industrial.

1.1.2 Objetivos Específicos

Os objetivos específicos deste trabalho são:

- Realizar um estudo sobre a aplicação das Redes de Petri Interpretadas na representação dos Sistemas a Eventos Discretos;
- Estudar as técnicas de implementação das Redes de Petri Interpretadas em equipamentos de controle;
- Propor uma metodologia para desenvolvimento de projetos de automação industrial;
- Validar a metodologia proposta projetando o controle de um processo virtual, para dois casos práticos e implementando os algoritmos resultantes da metodologia em duas tecnologias distintas (SFC para o CLP e VHLD para o FPGA); e
- Documentar os resultados das pesquisas em forma de uma dissertação.

1.2 Contribuição e relevância do trabalho

Este trabalho busca contribuir para síntese de projetos de automação industrial, propondo uma metodologia com etapas definidas com o objetivo de criar algoritmos que garantem a adequada execução do processo, atendendo as especificações de funcionamento e a corretude lógica.

Atualmente, devido a ausência de uma metodologia unificada, os projetos de automação industrial são desenvolvidos sem a utilização dos métodos formais para validação, onde o projetista repete o processo de implementação e correção do algoritmo de controle incessantemente até alcançar uma certa confiabilidade, sendo que apenas os estados mais frequentes são validados.

A principal deficiência deste método de validação é que em qualquer instante da execução do processo o sistema pode alcançar um estado não validado e, consequentemente, não previsto em sua etapa de validação, podendo gerar violações nas especificações de funcionamento e de segurança do processo. Outro aspecto importante é que os erros lógicos contidos no algoritmo só serão visíveis após a implementação no equipamento de controle, gerando riscos aos operadores, danos ao próprio sistema e prejuízos financeiros.

O objetivo desta metodologia é propor a utilização de um formalismo matemático para auxiliar a síntese de projetos de automação industrial, formalismo que possibilita a representação, análise e simulação da lógica de controle sem a necessidade da implementação de algoritmo para sua correção, dando ao projetista condições de prever os possíveis estados que o sistema alcançará , gerando algoritmos de controle livres de *deadlocks*, *livelocks* e conflitos, aumentado a confiabilidade do processo.

1.3 Organização do trabalho

O texto desta dissertação possui a seguinte disposição:

- **Capítulo 2** - REPRESENTAÇÃO DOS SISTEMAS A EVENTOS DISCRETOS: são apresentados os conceitos sobre sistemas e as definições matemáticas das Redes de Petri e suas extensões.
- **Capítulo 3** - DISPOSITIVOS LÓGICOS PROGRAMÁVEIS: detalhes sobre a plataforma FPGA e conceitos sobre linguagem de especificação de *hardware* VHDL.
- **Capítulo 4** - MÉTODOS DE IMPLEMENTAÇÃO DAS RDP EM CLPS: neste capítulo são descritos os métodos de implementação de Redes de Petri.
- **Capítulo 5** - METODOLOGIA PARA DESENVOLVIMENTO DE PROJETOS DE AUTOMAÇÃO: descrição das etapas que compõe a metodologia e detalhes da implementação dos algoritmos aplicados ao controle de dois casos práticos.
- **Capítulo 6** - VALIDAÇÃO DA METODOLOGIA: detalhes da aplicação da metodologia na síntese dos algoritmos de controle.
- **Capítulo 7** - CONCLUSÃO: são destacados os resultados principais desta pesquisa e sugestões para trabalhos futuros.

1.4 Resultado Preliminar

A metodologia proposta para conversão das Redes de Petri Interpretada em Grafcet (SILVA; VALLIM, 2012) foi submetida para avaliação e aprovada pela Conferência Internacional de Aplicações Industriais (IEEE/IAS) - INDUSCON 2012. Sendo apresentada no dia 05/11/2012 na cidade de Fortaleza-CE. Tal trabalho é referênciado abaixo:

- SILVA, H. V. D. e VALLIM, M. B. R. “Metodologia para conversão de Rede de Petri Interpretada para Grafcet visando projetos de automação industrial”, 10^{th} IEEE/IAS International Conference on Industry Applications, IEEE/IAS, Fortaleza - CE,Nov, 2012.

2 REPRESENTAÇÃO DOS SISTEMAS A EVENTOS DISCRETOS

2.1 Introdução

Neste capítulo é apresentado um estudo sobre as definições das Redes de Petri abordadas neste trabalho, descrevendo suas propriedades matemáticas fundamentais para a compreensão da dinâmica da rede. Inicialmente, são descritas alguns conceitos sobre os sistemas a eventos discretos.

2.2 Sistemas a Eventos Discretos

Um sistema pode ser definido como um conjunto de elementos, que separadamente não seria possível, sendo representado matematicamente por um conjunto de variáveis interdependentes. Na literatura, tal grupo é dividido em duas subclasses os sistemas estáticos e dinâmicos, onde a saída atual dos sistemas dinâmicos dependem dos valores das entradas anteriores apresentadas ao sistema e podem ser descritos por equações diferenciais (CASSANDRAS; LAFORTUNE, 2008). Os SED possuem a característica que sua evolução depende do acontecimento de eventos, desprezando a passagem do tempo. Na Figura 1 é apresentado um resumo da classificação dos sistemas.

2.3 Redes de Petri

A Rede de Petri é um formalismo que possibilita a representação gráfica com um alto nível de abstração, apresentada por Carl Adam Petri em sua tese de doutorado em 1962, inicialmente aplicada para avaliações de desempenho e representação do fluxo de dados nos protocolos de comunicação (MURATA, 1989). Este formalismo é aplicado

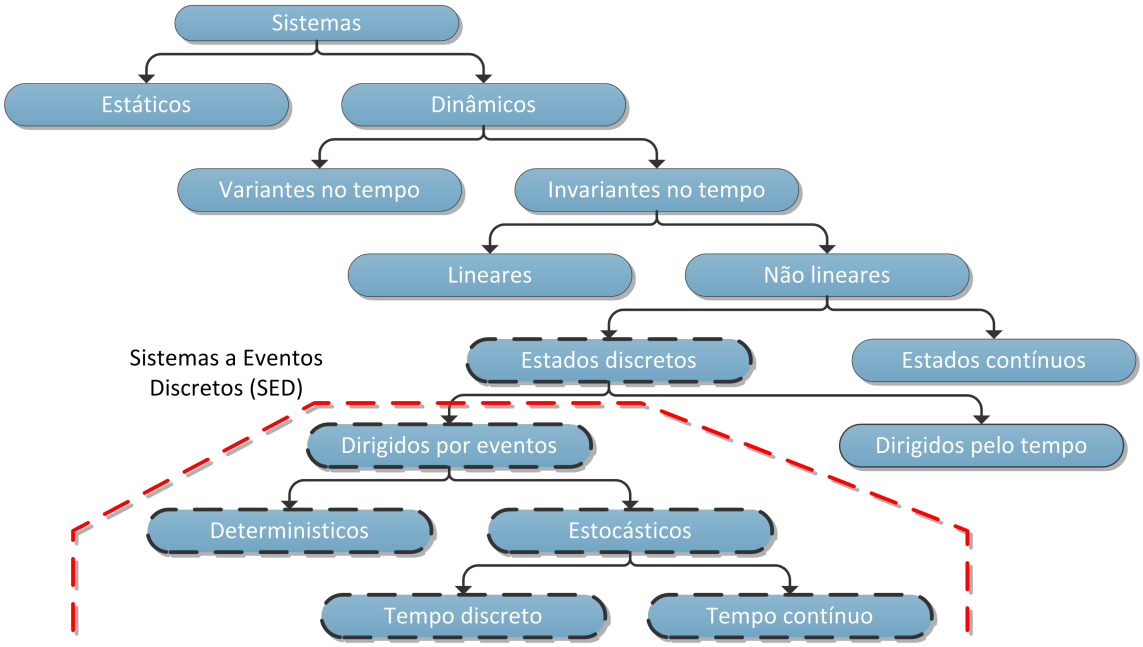


Figura 1 – Classificação de sistemas.
Fonte: Adaptado de Cassandras e Lafortune (2008).

na modelagem e análise de sistemas de automação industrial, representando de forma eficiente o comportamento assíncrono presente nos SEDs.

2.3.1 Conceitos de RdP

Definição 1. A estrutura da Rdp é uma quádrupla (RAMIREZ-TREVINO; RIVERA-RANGEL; LOPEZ-MELLADO, 2003), mostrado na equação (1):

$$G = (P, T, I, O) \tag{1}$$

O conjunto de lugares do modelo é definido por $P = \{p_1, p_2, \ldots, p_n\}$, e o conjunto de transições por $T = \{t_1, t_2, \ldots, t_m\}$ onde $I : P \times T \to \mathbb{Z}^+$ é a função que representa os pesos dos arcos que chegam às transições e $O : P \times T \to \mathbb{Z}^+$ é a função que representa os pesos dos arcos que saem das transições.

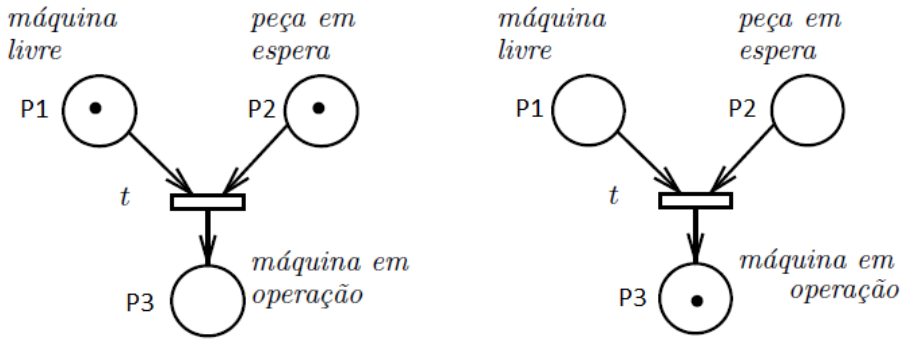
Os lugares p_n representam os possíveis estados que o sistema modelado pode assumir, e são descritos graficamente por um círculo.

As transições t_m são representadas graficamente por uma barra horizontal, associando os lugares que antecedem a transição com os lugares sucessores, sendo ligados através de arcos que possuem pesos individuais.

A matriz de incidência para G é uma matriz i x j cujo elemento $C = [c_{ij}]$, onde $c_{ij} = O(p_i, t_j) - I(p_i, t_j)$. A função de marcação $M : P \to \mathbb{Z}^+$ é a função que define a marcação atual da rede.

Definição 2. Uma RdP é um par $N = (G, M_0)$, onde G é a estrutura da RdP e M_0 é a marcação inicial da rede. A transição t_j está habilitada na marcação M_k se $\forall p_i \in P, M_k(p_i) \geq I(p_i, t_j)$; uma transição habilitada t_j pode ser disparada gerando uma nova marcação M_{k+1} podendo ser calculada por $M_{k+1} = M_k + Cv_k$ onde $v_k(j) = 1$. Tal equação é chamada de equação de estados da RdP.

As alocações das fichas nos lugares do modelo representam o estado geral do sistema. Já os eventos do sistema estão associados às transições. Para que uma transição seja habilitada, é necessário que as condições associadas às transições sejam atendidas. Uma vez disparada, a transição muda as posições das fichas entre os lugares conectados pelos arcos, retirando a ficha do lugar marcado p_1 e a deposita no lugar p_2 (CARDOSO; VALETTE, 1997), conforme apresentado na Figura 2(a).



(a) antes do disparo **(b) após o disparo**

Figura 2 – Dinâmica das Redes de Petri.
Fonte: Adaptado de Murata (1989).

Após o disparo da transição t as fichas são retiradas de p_1 e p_2 e depositadas em p_3, conforme mostrado na Figura 2(b).

Uma RdP pode ser utilizada para representar diversas entidades abstratas, dependendo do nível de abstração utilizado no seu desenvolvimento. Sendo assim, um lugar poderá representar uma entidade física ou não, como uma peça a ser processada ou uma máquina quebrada.

2.4 Grafcet

Originado na década de 70, o Grafcet surgiu com uma ferramenta gráfica para representação da lógica de controle de sistemas de automação industrial. Utilizado com

base para a criação da *Sequential Function Chart* (SFC) uma linguagem de programação de CLPs regulamentada através da IEC 1131-3, sendo atualizada em 2003 (INTERNATIONAL ELECTROTECHNICAL COMMISSION,). O Grafcet é utilizado principalmente na soluções de problemas cuja as ações de comando são sequenciais ou de tempo dependente, problemas que a solução utilizando representação Ladder, poderia se tornar confusa e muitas vezes inviáveis (DAVID, 1995).

Também considerado um caso particular da RdP, o Grafcet é definido como uma RdP binária, onde as matrizes Pre e $Post$ só podem assumir valores de 0 ou 1, correspondente às variáveis booleanas que simboliza a presença ou ausência em uma etapa (DAVID, 1995). Logo, um Grafcet é defino por um quadrupla mostrada na equação (2).

$$N =< E, T, Pre, Post > \tag{2}$$

onde E é o conjunto de etapas finitas, T é o conjunto de transições finitas, Pre é uma matriz com dimensões E x T, que relaciona as etapas anteriores com as transições precedentes e $Post$ é uma matriz com dimensões E x T, que relaciona as etapas posteriores com as transições anteriores.

Na Figura 3 são apresentados os elementos que compõem um Grafcet.

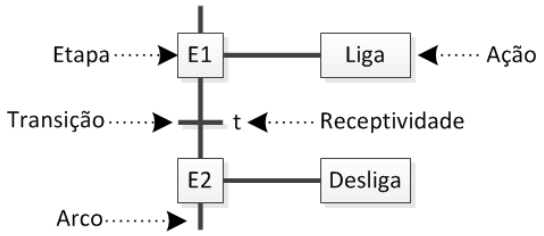


Figura 3 – Elementos do Grafcet.
Fonte: Traduzido de David (1995).

Através da organização dos elementos do Grafcet, o comportamento dinâmico é formado e seus elementos são divididos da seguinte maneira (DAVID, 1995):

- **Etapas:** simbolizam um estado no qual o comportamento da ação não se relaciona com as entradas e saídas atuais do sistema. Por definição, uma etapa é representada graficamente por um retângulo, sendo identificada por números. Durante a evolução do Grafcet, uma etapa pode estar ativa ou inativa. A etapa inicial é definida por uma retângulo duplo e representa a etapa que será ativada logo após o início do funcionamento do sistema.

- **Transições:** representadas graficamente por um traço perpendicular aos *arcos orientados* e conectando a etapa anterior com a etapa posterior à transição.

- **Arcos Orientados:** responsáveis por indicar a sequência de evolução do Grafcet, interligando etapas e transições, sendo interpretados da parte superior para inferior, representada por uma linha contínua.
- **Ações:** representam os efeitos que devem ser realizados quando alcançada uma determinada etapa do Grafcet. Cada ação é representada graficamente por um retângulo alocado à direta da etapa em que está associado. Alguns qualificadores podem ser associados às ações, alterando a maneira em que as ações são executadas. Como, por exemplo, a inserção dos qualificadores S (Set) ou R (Reset), utilizados como elementos memorizadores de ações.
- **Receptividade:** condição lógica associada à cada transição, e seus estados lógicos, habilitado ou não habilitado, ativam ou não as transições associadas.

Devido ao maior nível de representatividade, o Grafcet se popularizou no ambiente industrial, possibilitando a construção de lógicas de controle mais legíveis.

2.5 Modelagem de processos utilizando Redes de Petri

Um processo industrial pode evoluir de várias maneiras e possuir diversas formas de interações. Logo, serão descritas algumas estruturas importantes para a compreensão do trabalho.

Uma modelagem sequencial pode representar um processo de fabricação, onde a finalização da primeira etapa é essencial para evolução do sistema, conforme mostrado na Figura 4.

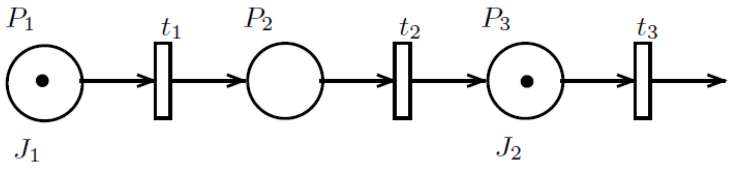
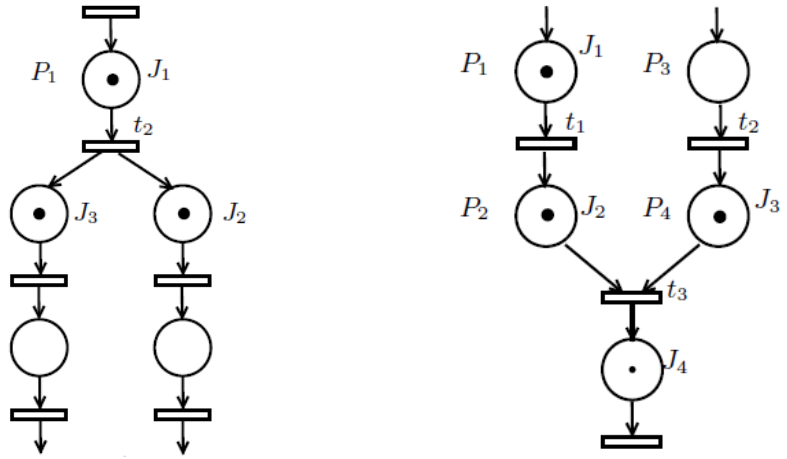


Figura 4 – Estrutura de uma Rdp para modelar comportamento sequencial.
Fonte: Adaptado de Peterson (1981).

A Figura 4 pode representar vários processos, sendo que conforme o nível de abstração empregado utilizado, um lugar pode representar um estado da máquina ou condições de funcionamento da mesma.



(a) Divisão **(b) Junção**

Figura 5 – Exemplo de estruturas de Rede de Petri.
Fonte: Adaptado de Peterson (1981).

No caso da *divisão*, logo após uma transição existem dois lugares distintos dando origem a duas fichas, onde os lugares podem simbolizar dois processos diferentes sendo executados ao mesmo tempo, introduzindo a ideia de simultaneidade. Após a criação das novas fichas, a rede pode evoluir de modo independente conforme visto na Figura 5(a).

Por outro lado, na *junção* a ideia é contraria, antes de uma transição existem dois lugares marcados (P_2 e P_4) e após o disparo da transição as duas fichas são retiradas de P_2 e P_4 e depositadas em J_4 conforme mostrado na Figura 5(b).

Um comportamento comum nos processos industriais é a condição de *repetição*, pois demonstra a característica do sistema em repetir um processo até que uma condição seja satisfeita, conforme mostrado na Figura 6.

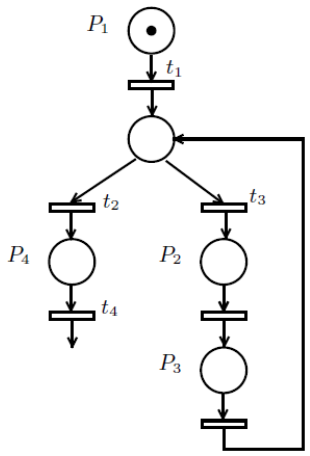


Figura 6 – Repetição.
Fonte: Adaptado de Peterson (1981).

Os processos associados aos lugares p_2 e p_3 serão realizados até que uma con-

dição seja alcançada, conforme mostrado na Figura 6.

Sistema que possuem processo que devem ser executados de forma individual são representados por uma estrutura conhecida com exclusão mutua, que garante a prioridade ao acesso do elemento compartilhado. E esta estrutura é mostrada na Figura 7.

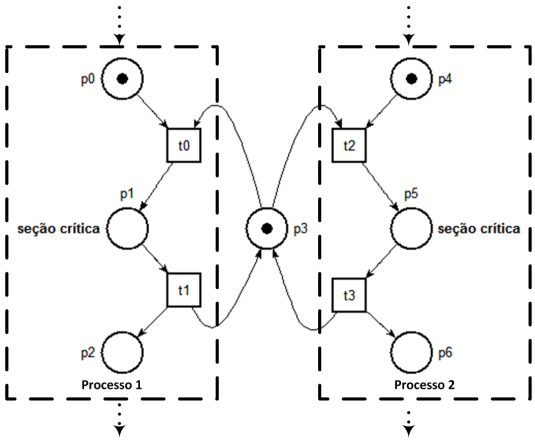


Figura 7 – Exclusão mútua.
Fonte: Adaptado de Peterson (1981).

Outras estruturas podem ser encontradas na literatura, como a alocação de recurso, caminhos alternativos, entre outros.

2.5.1 Rede de Petri Interpretada

As Redes de Petri Interpretadas (RdPIs) são uma extensão das Redes de Petri ordinárias (DAVID; ALLA, 1992), podendo descrever sinais e eventos de entrada e saída originados do meio externo. Tais eventos são associados às transições através de expressões booleanas. As RdPI podem ser aplicadas para o diagnóstico de falhas em SED (SANTOYO-SANCHEZ; RUIZ-BELTRAN; AGUIRRE-SALAS L.I.AND ORTIZ-MURO, 2008), sintonização e sincronização (RAMIREZ-PRADO et al., 2000; AGUIRRE-SALAS; SANTOYO-SANCHEZ, Mallorca: IEEE, 2009.), bem como na representação da lógica de controle (GUSTIN, 1999).

Definição 3. Formalmente, uma RdPI é uma sêxtupla, conforme apresentado na equação 3.

$$Q = (N, \Sigma, \Phi, \lambda, D, \varphi) \tag{3}$$

onde:

1. $N = (G, M_0)$ é uma RdP;

2. $\Sigma = \{\sigma_1, \sigma_2, ..., \sigma_r\}$ é o alfabeto das entradas para a rede onde σ_i é um símbolo de entrada;

3. $\Phi = \{\phi_1, \phi_2, ..., \phi_s\}$ é o alfabeto das saídas para a rede onde ϕ_i é um símbolo de saída;

4. $\lambda : T \rightarrow \Sigma \cup \{\varepsilon'\}$ é a função que rotula as transições e respeita as seguintes especificações: $\forall t_j, t_k \in T, j \neq k$ se $I(p_i, t_j) = I(p_i, t_k) \neq 0$ e tanto $\lambda(t_j), \lambda(t_k) \neq \varepsilon'$, quanto $\lambda(t_j) \neq \lambda(t_k)$. Neste caso, ε' representa um evento interno do sistema;

5. $D : T \rightarrow \top$ é a função de avanço da rede, onde $\top = \tau \cup \{\varepsilon\}$ e $\tau \subset T$. Quando a transição t_j é disparada, o nome de j_j ou o símbolo ε são associados a uma saída da rede. Neste caso, ε representa a ausência do sinal de saída;e

6. $\varphi : R(G, M_0) \rightarrow \{\Phi\}^q$ é uma função de saída, onde $R(G, M_0)$ é o conjunto de alcançabilidade definido para uma RdP e q é o número de saída válidas associadas ao lugar.

Definição 4. A sequência de disparo de uma RdPI (Q, M_0) é a sequência $\sigma = t_i, t_j, ..., t_k$ de tal forma que $M_0 \xrightarrow{t_i} M_1 \xrightarrow{t_j}, ..., M_w \xrightarrow{t_k}$. O conjunto de todas as sequências de disparo são chamados de linguagem de disparo $\mathcal{L}(Q, M_0) = \{\sigma | \sigma = t_i, t_j, ..., t_k$ e $M_0 \xrightarrow{t_i} M_1 \xrightarrow{t_j}, ..., M_w \xrightarrow{t_k}\}$.

Com base na linguagem $\mathcal{L}(Q, M_0)$ pode-se definir as linguagens de entrada $\mathcal{L}_{in}(Q, M_0)$ e de saída $\mathcal{L}_{out}(Q, M_0)$, apresentadas na definição 5.

Definição 5. Considerando a linguagem de entrada sendo $\mathcal{L}_{in}(Q, M_0) = \{\lambda(t_i), \lambda(t_j), ..., \lambda(t_k) | t_i, t_j, ..., t_k \in \mathcal{L}(Q, M_0)\}$ e a linguagem de saída sendo $\mathcal{L}_{out}(Q, M_0) = \{\varphi(M_0)\varphi(M_1), ..., \varphi(M_w) | M_0 \xrightarrow{t_i} M_1 \xrightarrow{t_j}, ..., M_w \xrightarrow{t_k}, ...$ e $t_i, t_j, ..., t_k \in \mathcal{L}(Q, M_0)\}$

As RdPI associam as variáveis do sistema às transições da rede, que descrevem as condições e ações existentes no processo. Estas variáveis podem indicar os estados dos atuadores ou sensores do sistema, permitindo a modelagem das interações com o ambiente externo (RAMIREZ-TREVINO; RIVERA-RANGEL; LOPEZ-MELLADO, 2003).

Uma RdPI é dividida em duas partes, sendo elas: controle e dados. O controle descreve todas as evoluções possíveis do processo relacionadas aos eventos. Os dados descrevem as estruturas de dados internas ao sistema e as informações recebidas do mundo externo. Os elementos que constituem os dados de uma Rede de Petri Interpretada são as condições e as ações .

As condições C_i e as ações A_i são expressas como disjunção e conjunção *booleanas* e são associadas às respectivas transições. As condições e ações são representadas graficamente por (C_i, A_i) ao lado da transição pertencente.

Resumidamente, os passos para modelar um sistema utilizando RdPI são (CARDOSO; VALETTE, 1997):

1. Encontrar a estrutura da Rede de Petri que representa as atividades concorrentes, paralelas e sequenciais, e descrever os eventos associados ao meio externo.
2. Analisar a rede gerada verificando as propriedades. Caso necessário, corrigir os eventuais problemas.
3. Simular a Rede Interpretada (controle e dados) para extrair o comportamento do sistema.

2.5.2 Rede de Petri *Input*/*Output Place*/*Transition*

Outras extensões das Redes de Petri ordinarias são utilizada para a representação da lógica de controle dos processo de automação. Pais, Barros e Gomes (2005) apresenta a Rede de Petri *Input/Output Place/Transition* (IOPT) sendo uma melhoria das Redes de Petri *Place/Transition* (DESEL; REISIG, 1998). A Rede de Petri IOPT possui a característica de associar as expressões lógicas diretamente nos lugares ou transições do modelo. Trabalhos recentes apresentam ferramentas computacionais que automatizam o processo de conversão do modelo em algoritmos de controle (GOMES et al., 2007).

Em Pais, Barros e Gomes (2005) são apresentadas as definições matemáticas que descrevem a modelagem e a dinâmica das Redes de Petri IOPT, são elas:

Definição 6 (Interface do sistema)**.** A base para o controle dos sistemas utilizando as Redes de Petri IOPT é uma quádrupla:

$$ICS = (IS, IE, OS, OE) \tag{4}$$

Com base na equação (4) algumas condições são requeridas:

1. IS é o conjunto finito de sinais de entrada;
2. IE é o conjunto finito de eventos de entrada;
3. OS é o conjunto finito de sinais de saída;
4. OE é o conjunto finito de eventos de saída;e

5. $IS \cap\ IE \cap OS \cap OE = \emptyset$.

Os eventos de entrada e saída são representados, sendo que a composição de vários sinais dão origem aos eventos.

Definição 7 (Estado de entrada do sistema)**.** Os estados de entrada do sistema são definidos pelo par mostrado na equação (5).

$$SIS = (ISB, IEB) \tag{5}$$

1. ISB é o conjunto finito de sinais de entrada associados: $ISB \subseteq IS \times \mathbb{N}$;e
2. IEB é o conjunto finito de eventos de entrada associados: $IEB \subseteq IE \times \mathbb{B}$.

Definição 8 (Rede IOPT)**.** Uma Rede de Petri IOPT é uma N-upla $N = (P, T, A, TA, M, weight, weightTest, priority, isg, ie, oe, osc)$ que satisfaça as seguintes condições:

1. P é o conjunto finito de lugares;
2. T é o conjunto finito de transições;
3. A é o conjunto de arcos, tal que $A \subseteq ((P \times T) \cup (T \times P))$;
4. TA é o conjunto de arcos de teste, tal que $TA \subseteq (P \times T)$;
5. M é a função de marcação de: $M : P \to \mathbb{N}_0$;
6. $weight : A \to \mathbb{N}_0$;
7. $weightTest : TA \to\ \mathbb{N}_0$;
8. $priority$ é uma função parcial aplicada às transições para resolução de conflitos: $priority : T \to \mathbb{N}_0$;
9. isg é uma função parcial de monitoramento de sinal de entrada aplicada às transições através de expressões booleanas: $isg : T \to BE$, onde $\forall eb \in igs(T), Var(eb) \subseteq IS$;
10. ie é uma função parcial de monitoramento do evento de entrada aplicadas às transições dos eventos de entrada: $ie : T \to IE$;
11. oe é uma função parcial de monitoramento do evento de saída aplicadas as transições dos eventos de saída: $oe : T \to OE$;e

12. osc é uma função condicional do sinal de saída para os lugares contidos no conjunto de regras: $osc : P \rightarrow P(RULES)$, onde $RULES \subseteq (BES \times OS \times \mathbb{N}_0)$, $BES \subseteq BE$ e $\forall e \in BES, Var(e) \subseteq ML$.

Definição 9 (Condições ativas)**.** Tomando como base $N = (P, T, A, TA, M, weight, weightTest, priority, isg, ie, oe, osc)$ e as equações (4) e (5), uma transição t é habilitada para ser disparada se, e somente se, as condições a seguir forem satisfeitas:

1. $\forall\ p \in \bullet t, M(p) \geq weight(p,t)$;
2. $\forall p \in \Diamond t, M(p) \geq weightTest(p,t)$;
3. A transição t evolui $(ie(t), true) \in IEB$

Definição 10 (Etapas da rede IOPT)**.** Considerando $ET \subseteq T$ sendo o conjunto de todas as transições habilitadas mostrada na definição 9, e Y sendo as etapas em N se a condição apresentada na equação (6) for satisfeita.

$$\begin{aligned} &Y \subseteq ET \wedge \forall t_1 \in (ET \backslash Y), \exists SY \subseteq Y, (\bullet t_1 \cap \bullet SY) \neq \emptyset \wedge \exists p \in (\bullet t_1 \cap \bullet SY), \\ &\left(weight(p,t_1) + \sum_{t \in SY} weight(p,t) > M(p) \right) \end{aligned} \tag{6}$$

Definição 11 (Etapas de evolução e marcação posterior)**.** O acontecimento da etapa Y e uma rede no estado N leva a uma rede $N' = (P, T, A, TA, M', weight, weightTest, priority, isg, ie, oe, osc)$, sendo que a marcação posterior da rede segue a expressão apresentada na equação 7.

$$M' = \left\{ \left(p, m - \sum_{t \in Y \wedge (p,t) \in A} weight(p,t) + \sum_{t \in Y \wedge (p,t) \in A} weight(t,p) \right) \in (P \times \mathbb{N}_0) \,|\, (p,m) \in M \right\} \tag{7}$$

2.6 Conclusão

Neste capítulo apresentou-se alguns conceitos básicos sobre as RdP, ressaltando os detalhes da modelagem de SEDs e as definições matemáticas presente no formalismo. Também apresentou-se algumas propriedades das RdPI e dos RdP IOPT, onde foi possível

identificar algumas relacionadas a forma de descrever o processo, sendo que a RdP IOPT não utiliza Tabela de Dados para relacionar as transições às condições, já para a RdPI o termo evento não está presente do modelo, gerando um aumento nos termos que são associados às transições.

Analisando a legibilidade do modelo, a RdPI se mostra superior descrevendo apenas os estados dos elementos que compõem o processo, possibilitando a representação do real estado do sistema.

3 DISPOSITIVOS LÓGICOS PROGRAMÁVEIS

3.1 Introdução

Neste capítulo são apresentados conceitos sobre a arquitetura de um Field-programmable gate array (FPGA), incluindo aspectos físicos e a linguagem de especificação de *hardware*.

3.2 Conceitos sobre FPGA

Segundo Bobda (2007) o FPGA foi desenvolvido e apresentado pela empresa Xilinx em 1985, sua arquitetura é composta por uma matriz de blocos lógicos ou células lógicas, sendo envolvidos por blocos de entradas/saídas (E/S) alocados individualmente. Os blocos lógicos são ligados aos blocos de E/S através de interligações programáveis, sendo que cada bloco lógico pode ser programado individualmente, conforme mostrado na Figura 8.

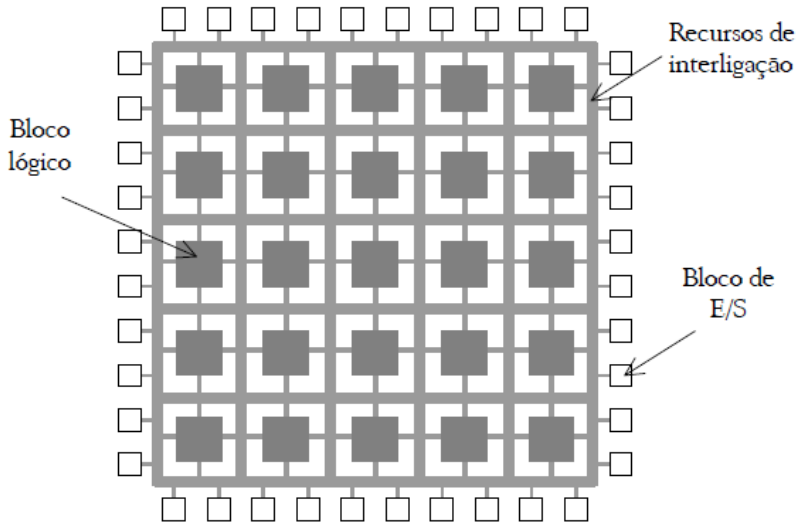


Figura 8 – Arquitetura interna do FPGA.
Fonte: Traduzido de Rose, Gamal e Sangiovanni-Vincentelli (1993).

Cada bloco de E/S pode ser programado como entrada, saída ou bidirecional. Os componentes de interligação permitem a distribuição do sinal responsável pela sincronia de todos os blocos lógicos.

Este equipamento pode ser desenvolvido utilizando duas tecnologias existentes: a *antifuse* e a *memory-based*. As propriedades da tecnologia *antifuse* são utilizadas para as ligações elétricas. Por outro lado, a *memory-based* é utilizada nos cálculos no período de processamento. A tecnologia *memory-based* é baseada nas memórias do tipo *Static Random Access Memory* (SRAM), *Electrically Erasable Programmable Read-Only Memory* (EEPROM) e *Flash*.

Os *antifuse* realizam as conexões elétricas das entradas e saídas dos FPGAs, sendo facilmente modificados devido as suas características construtivas. Já as *memory-based* são utilizadas para armazenar dados durante a execução do algoritmo.

Os dispositivos lógicos programáveis possuem a propriedade de serem reconfigurados dinâmicamente, sem a necessidade de interromper a execução das rotinas. Isto é realizado com o uso de geradores de função, para isso são utilizados os multiplexadores e os *Look-up tables*. O multiplexadores gerenciam as linhas que ativam as conexões de entrada e saída, mantendo a integridade do hardware. O *look-up tables* representam um grupo de células de memória que contêm todos os resultados possíveis de uma dada função para um determinado conjunto de valores de entrada considerando todas as combinações possíveis.

As estruturas de um FPGA podem variar para cada fabricante e modelo, sendo classificadas em : *Symmetrical array*, *Row-based*,*Hierarchy-based* e *Sea of gates*, conforme mostradas na Figura 9.

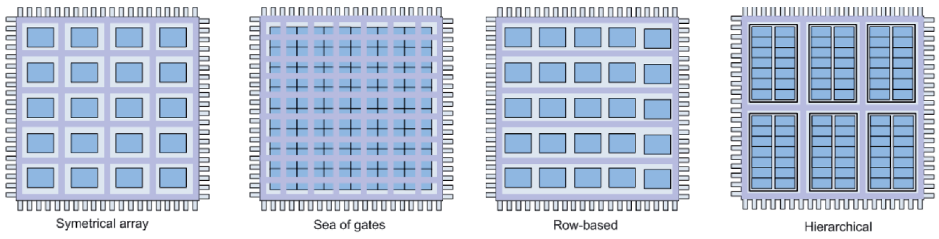


Figura 9 – Estruturas básicas de um FPGA.
Fonte: Bobda (2007).

A estrutura *Symmetrical array* é composta de uma matriz bidimensional de blocos lógicos inserida em um conjunto de linhas verticais e horizontais e seus elementos de comutação estão inseridos nas intersecções das linhas verticais e horizontais, permitindo as ligações de linhas verticais com as horizontais.

A estrutura *Row-base* consiste em linhas alternadas de bloco de lógicos e canais.

Estes canais são os espaços livres entre os blocos lógicos utilizados para o envio de sinais de controle.

A *Hierarchy-based* possui a característica de organizar a inserção das células de forma hierárquica, colocando os elementos com a menor granularidade em níveis mais baixos.

O *Sea of gates* são semelhantes as *Symmetrical arrays*, onde as células são dispostas em uma matriz bidimensional sendo que uma entrada na matriz corresponde à coordenada de uma célula. A principal diferença é que não existem canais entre os blocos lógicos.

3.3 Linguagem de especificação de *hardware*

Para especificar as conexões destes elementos lógicos, pode-se utilizar as linguagens de especificação de *hardware* Verilog ou VHDL. Estas linguagens possuem um alto nível de abstração, proporcionando algumas vantagens quando utilizadas, são elas:

- A linguagem de especificação é independente do *hardware* que será implementado, logo em fase de projeto é possível mudar de plataforma sem comprometer o algoritmo;
- Aumento da legibilidade do algoritmo, devido a utilização de processos para organização;
- Diminuição das documentações geradas ao final da programação, devido aos comentários inseridos no próprio algoritmo desenvolvido;e
- Permite a simulação dos algoritmos construídos, sendo possível a visualização dos sistemas digitais que serão executados.

3.3.1 VHDL

Very High-Speed Integrated Circuit Hardware Description Language (VHDL) é a linguagem de especificação mais popular, utilizada para descrever os sistemas lógicos digitais, especificando as conexões dos blocos contidos no FPGA, suportando também as descrições de *hardware Register Transfer Level* (RTL). Surgiu na década de 80, da tentativa de unificar as descrições das estruturas associadas aos circuitos integrados. Alguns anos depois se tornou uma padrão definido pela *Institute of Electrical and Electronics Engineers* (IEEE) (HAUCK; DEHON, 2008).

A linguagem de especificação VHDL possui diferenças quando comparada com a linguagem C, devido a sua semântica paralela, mas possuem algumas semelhanças com as linguagens orientadas a objetos nos aspectos construtivos do algoritmo. A linguagem de especificação VHDL pode ser dividida em três elementos: *Package*, *Entiry* e *Achitecture*. Na Figura 10 são apresentados tais elementos.

LIBRARY IEEE; **USE IEEE.STD_LOGIC_1164.all;** **USE IEEE.STD_LOGIC_UNSIGNED.all;**	**PACKAGE** **(BIBLIOTECAS)**
ENTITY exemplo **IS** **PORT (** < descrição dos pinos de entrada e saída > **);** **END** exemplo;	**ENTITY** **(PINOS DE I/O)**
ARCHITECTURE teste **OF** exemplo **IS** **BEGIN** **PROCESS(** <pinos de entrada e signal > **)** **BEGIN** < descrição do circuito integrado > **END PROCESS;** **END** teste;	**ARCHITECTURE** **(ARQUITETURA)**

Figura 10 – Estrutura básica do algoritmo VHDL.
Fonte: Rose, Gamal e Sangiovanni-Vincentelli (1993).

Na parte conhecida como *Package*, são descritas as bibliotecas disponíveis que serão utilizadas durante a execução do algoritmo, tipicamente são chamadas as bibliotecas std_logic_1164, que permite a utilização dos operadores booleanos, as std_logic_arith e $std\_logic\_unsigned$, que possibilitam a utilização dos operadores matemáticos.

Na seção *Entity* são descritos os pinos de entrada e saída e suas especificações de funcionamento, como tamanho do bit e tipo de dados.

Na *Achitecture* são montadas as estruturas que serão executadas, *functions* e *procedures* são utilizados para especificar as rotinas que serão executadas de forma paralela.

3.4 Conclusão

Neste capítulo foram apresentadas algumas propriedades dos FPGAs, descrevendo sua arquitetura de *hardware* e a principal linguagem de especificação utilizada. A facilidade na reorganização dos blocos lógicos associada à linguagem de especificação que dispõe de um alto poder de abstração, garantem a aplicação dos FPGA em vários projetos.

Devido à flexibilidade deste equipamento e suas propriedade paralelas, alguns trabalhos propõem sua utilização como equipamentos de controle de processos com base nas RdP (FERNANDES; ADAMSKI; PROENCA, 1997; SILVA et al., 2007; SILVA et al., 2010).

4 MÉTODOS DE IMPLEMENTAÇÃO DAS RDP EM CLPS

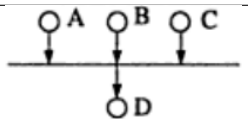
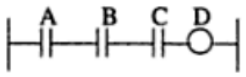
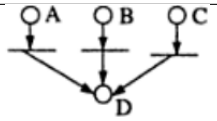
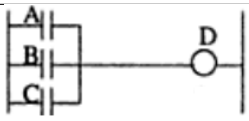
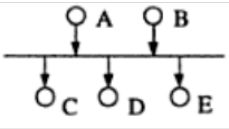
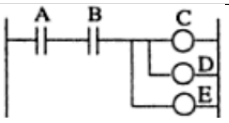
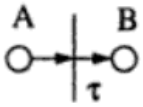
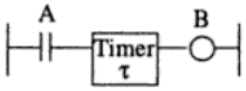
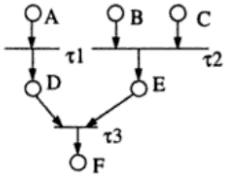
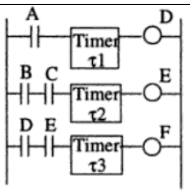
4.1 Introdução

Neste capítulo serão apresentados alguns métodos de conversão das RdP para as linguagens de programação de CLP, explorando os aspectos relacionados a forma em que as RdP são traduzidas para a linguagem de controle. Ao final do capítulo há uma breve discussão sobre a aplicação e as vantagens dos métodos apresentados.

4.2 Métodos de conversão de RdP para *Ladder*

Devido ao grande número de aplicações industriais que utilizam como equipamento de controle os CLPs e a grande aceitação da linguagem de Diagrama *Ladder* para especificação das rotinas de controle, surgiram técnicas de implementação das RdP diretamente nesta linguagem. Técnicas que propõem a tradução direta dos elemento contidos na RdP para estruturas específicas descritas em *Ladder*, conforme mostrado no Quadro 1.

Em Peng e Zhou (2004) são descritas algumas estruturas e técnicas para a implementação das RdP, detalhando os esquemas utilizados para cada implementação e os *hardwares* empregados ao controle. Em Zhou e Venkatesh (1999) é descrita uma metodologia para implementação das RdP onde as estruturas básicas presentes nos modelos são diretamente representadas na linguagem Ladder. No Quadro 1 tais estruturas são mostradas.

Estrutura lógica	Redes de Petri	Diagrama *Ladder*
Lógica “E” (AND)		
Lógica “OU” (OR)		
Modelo Concorrente		
Modelo sequencial temporizado		
Modelo síncrono		

Quadro 1 – Tabela de conversão das Redes de Petri em Diagrama *Ladder*
Fonte: Traduzido de Zhou e Venkatesh (1999, p. 262).

Utilizando as estruturas de conversão mostradas no Quadro 1 é possível implementar uma RdP em um CLP, sendo que em alguns casos, o Diagrama *Ladder* gerado pode-se tornar extenso e ilegível, devido ao tamanho da rede, dificultando o processo de implementação e expansão das rotinas de controle.

4.3 Métodos de conversão de RDP para SFC

Utilizando as definições da RdPI apresentadas em (DAVID; ALLA, 2010) e utilizando como base a proposta de construção do Grafcet apresentada em (DAVID; ALLA, 1992; DAVID, 1995), podem ser estabelecidas as seguintes etapas de tradução:

1. Criar para cada elemento do sistema uma RdPI que associa as condições e ações essenciais para a evolução do processo;

2. Agregar todas as RdPI criadas com os elementos auxiliares (temporizadores e contadores) e organizar as condições e ações de cada transição em uma tabela de dados;

3. Analisar as propriedades da RdPI geral e simular a rede em conjunto com a tabela de dados, buscando comprovar a correta evolução do processo;

4. Criar módulos com os elementos que possuem condições e ações compartilhadas;

5. Converter cada módulo para Grafcet, considerando as condições como a receptividade da transição e as ações como etapas;

6. Marcar as etapas iniciais de cada SFC com o estado inicial de cada elemento;

7. Implementar no CLP todos os SFCs gerados, que serão executados simultaneamente.

Na Figura 11 é apresentado um esquemático do método de conversão.

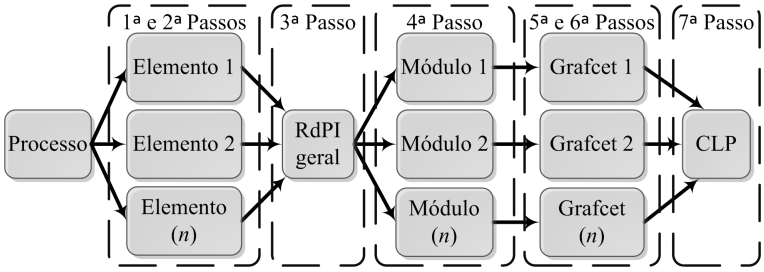


Figura 11 – **Esquemático do método utilizado.**
Fonte: Autoria própria.

4.4 Conclusão

Neste capítulo foram apresentados dois métodos para implementação das RdP em equipamentos de controle, sendo que o primeiro método propõe a conversão das estruturas contidas no modelo diretamente para o diagrama Ladder, podendo gerar um linguagem extensa e complexa, dificultando a reutilazação do algoritmo e reduzindo a sua legibilidade.

Por sua vez, a segunda metodologia descreve a conversão do formalismo em SFC através da execução de sete etapas, proporcionando a criação de módulos independentes de controle, que realizam ações específicas para cada elemento que compõem o sistema. Esta metodologia também possibilita a reutilização do algoritmo, pois gera códigos resumidos e mais legíveis quando comparados com a primeira metodologia.

5 METODOLOGIA PARA DESENVOLVIMENTO DE PROJETOS DE AUTOMAÇÃO

5.1 Introdução

Neste capítulo serão apresentados os detalhes sobre a metodologia proposta, explorando as etapas para conversão do formalismo em códigos implementáveis.

5.2 Metodologia proposta

Este trabalho propõe uma metodologia para construção de projetos de automação industrial, com base em um formalismo com alto poder de abstração e realizando etapas de conversão bem definidas. Esta metodologia é constituida de nove etapas, são elas:

1. Criar para cada elemento do sistema uma RdPI que associa às condições e ações essenciais para a evolução do processo;
2. Agregar todas as RdPI criadas com os elementos auxiliares (temporizadores e contadores) e organizar as condições e ações de cada transição em uma tabela de dados;
3. Analisar as boas propriedades da RdPI;
4. Validar a lógica de controle acompanhando a dinâmica da RdPI e sua tabela de dados;
5. Corrigir o formalismo criado, caso seja necessário;
6. Criar módulos com os elementos que possuem condições e ações compartilhadas;
7. Converter cada módulo para um Grafcet ou Máquina de Estados Finito;
8. Marcar as etapas iniciais de cada Grafcet ou Máquina de Estados Finito com o estado inicial de cada elemento do sistema;

9. Converter os Grafcets(CLP) ou Máquinas de estados finitos(FPGA) para as linguagems de programação para os equipamentos de controle em específico.

Com o objetivo de detalhar as etapas apresentadas na metodologia cria-se um esquemático conforme apresentado na Figura 12.

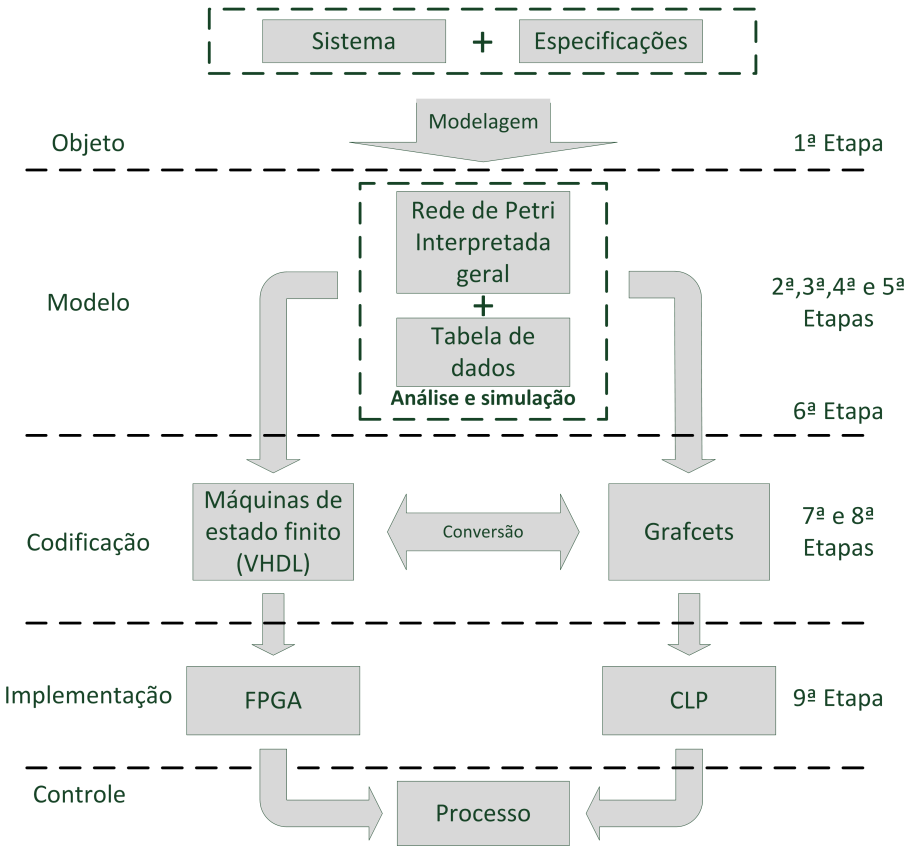


Figura 12 – Esquema da metodologia proposta.
Fonte: Autoria própria.

Com base na Figura 12 pode-se observar que a primeira etapa é o ponto inicial do processo de síntese dos algoritmos de controle, nesta etapa é gerada um RdPI para cada elemento que compõe o sistema, considerando os estados dos elementos modelados e sua especificação de funcionamento.

Na segunda etapa as RdPI geradas na primeira etapa são unidas, realizado a simplificação de estados e transições que possam existir para cada elemento do sistema, nesta etapa as condições e ações associadas a cada transição contida no modelo são repassadas e organizadas em uma tabela de dados. O resultado desta etapa é a criação da tabela de dados e da RdPI geral, e com base na marcação desta rede obtemos o estado atual do sistema modelado.

Na etapa três da metodologia proposta são realizadas as análises da RdPI buscando identificar possíveis estados indesejados que o sistema pode vir a alcançar. Análises

da estrutura e alcançabilidade também são realizadas com a pretensão de identificar os invariantes de lugar e transição que fornecem informações importantes sobre a dinâmica da RdPI e consequentemente do sistema modelado.

Na quarta etapa a RdPI geral e a tabela de dados são simuladas, sendo que a cada evolução da RdPI as condições e ações descritas na tabela de dados são testadas, possibilitando ao projetista acompanhar a dinâmica da RdPI e prever possíveis problemas lógicos.

Na quinta etapa, os erros encontrados na etapa anterior, caso existam, são corrigidos modificando o modelo representativo do sistema ou até mesmo algumas integrações entre os elementos que compõem o sistema modelado. Sendo que a cada mudança realizada no sistema e nas especificações de funcionamento faz-se necessário o reinício do processo de síntese do algoritmo de controle apresentado nesta metodologia, retornando-se à Etapa 1.

Na sexta etapa são criados módulos de controle para cada elemento, observando as condições e ações expressas na tabela de dados, buscando identificar as condições necessárias que iniciam o funcionamento do elemento e as ações a serem executadas após a validação desta condição. Nesta etapa, podem-se criar módulos que auxiliam o controle do processo, dando origem a elementos de armazenamento de informações, contadores, temporizadores entre outros.

A sétima etapa propõe a conversão dos módulos gerados pela etapa seis em algoritmos implementáveis para equipamentos de controle, onde estes módulos são convertidos para Grafcets ou Máquinas de Estados Finitos. Quando a conversão é realizada para Grafcet as condições contidas na tabela de dados são repassadas para as transições do Grafcet e as ações são associadas às etapas da linguagem. Para as Máquinas de Estado Finito as condições são associadas aos arcos que interligam os lugares. Uma vez gerado um conjunto de Grafcet é possível a conversão direta para Máquinas de Estados Finitos, representada na Figura 12 pela seta bidirecional no cetro da imagem. Com base nesta conversão, pode-se reescrever um Grafcet para um conjunto de especificações de *hardware* descritas sobre a linguagem VHDL.

Na oitava etapa da metodologia, são realizadas as marcações iniciais das etapas dos Grafcets criados e a identificação do lugares iniciais das Máquinas de Estados Finitos, sendo considerado o estado inicial de cada elemento do sistema.

Na última etapa os algoritmos de controle gerados pela execução da metodologia são gravados em seus equipamentos de controle, os SFCs gerados são descritos em um *software* e gravados no CLP. As Máquinas de Estado Finito são escristas utilizando a linguagem de especificação de *hardware* VHDL e executados pelo processador FPGA.

5.3 Conclusão

Neste capítulo foram expostas as etapas que compõem a metodologia proposta detalhando os passos para síntese do algoritmo de controle. Com base nesta metodologia apresentada é possível sintetizar um algoritmo de controle legível e reutilizável, dando ao projetista a possibilidade de validar a lógica de controle em tempo de projeto, sem a necessidade de implementar algoritmo de controle para validar o seu funcionamento.

6 VALIDAÇÃO DA METODOLOGIA

6.1 Introdução

Neste capítulo serão apresentados os detalhes da aplicação da metodologia em dois casos práticos. O primeiro projeto de automação é apresentado para exemplificar a utilização da metodologia detalhando todas as etapas que compõem a metodologia. O segundo projeto prático possui especificações de funcionamento mais complexas, onde-se buscou comprovar a aplicabilidade da metodologia proposta em processos com maior complexidade. Ao final do capítulo uma análise dos resultados obtidos é apresentada e organizada em forma de tabela, proporcionando uma comparação entre os resultados para os dois casos práticos abordados.

6.2 Sistema utilizado

Para validação da metodologia proposta, realiza-se a modelagem é o controle de um processo industrial virtual. O programa utilizado para simular estes sistemas é o *Interactive Training System for Programmable Logic Controller* (ITS PLC) *Professional Edition* que dispõe de cinco sistemas industriais para simulação. O programa oferece uma interface simples para o usuário, possibilita a mudança do ponto de vista e a interação com componentes do sistema (caixas, paletes e etc.). Seus efeitos sonoros e os detalhes gráficos aumentam a sensação de realidade (MAGALHÃES, 2009).

A utilização de sistemas industriais simulados possibilita a validação da metodologia proposta sem a necessidade da construção física dos sistemas, mantendo a dinâmica discreta dos sistemas que serão simulados (VIGÁRIO; MAGALHÃES; FREITAS, 2006; RIERA et al., 2009; MAGALHÃES; VIGÁRIO, 2009).

O ITS PLC é instalado em um computador convencional, conectado ao equipamento de controle através de uma placa de condicionamento de sinal que utiliza uma placa

Data aquisition (DAQ) e uma placa de isolamento de sinal. Logo, os sinais elétricos recebidos do sistema virtual são repassados em forma de tensão para os equipamentos de controle através da placa DAQ.

Para exemplificar a aplicação do método apresentado realiza-se o controle do processo de Triagem de Caixas. Este sistema contém um conjunto de esteiras que são acionadas individualmente para que as caixas sejam despachadas corretamente. Na Figura 13 é mostrada a alocação das esteiras transportadoras, os sensores e atuadores no ambiente.

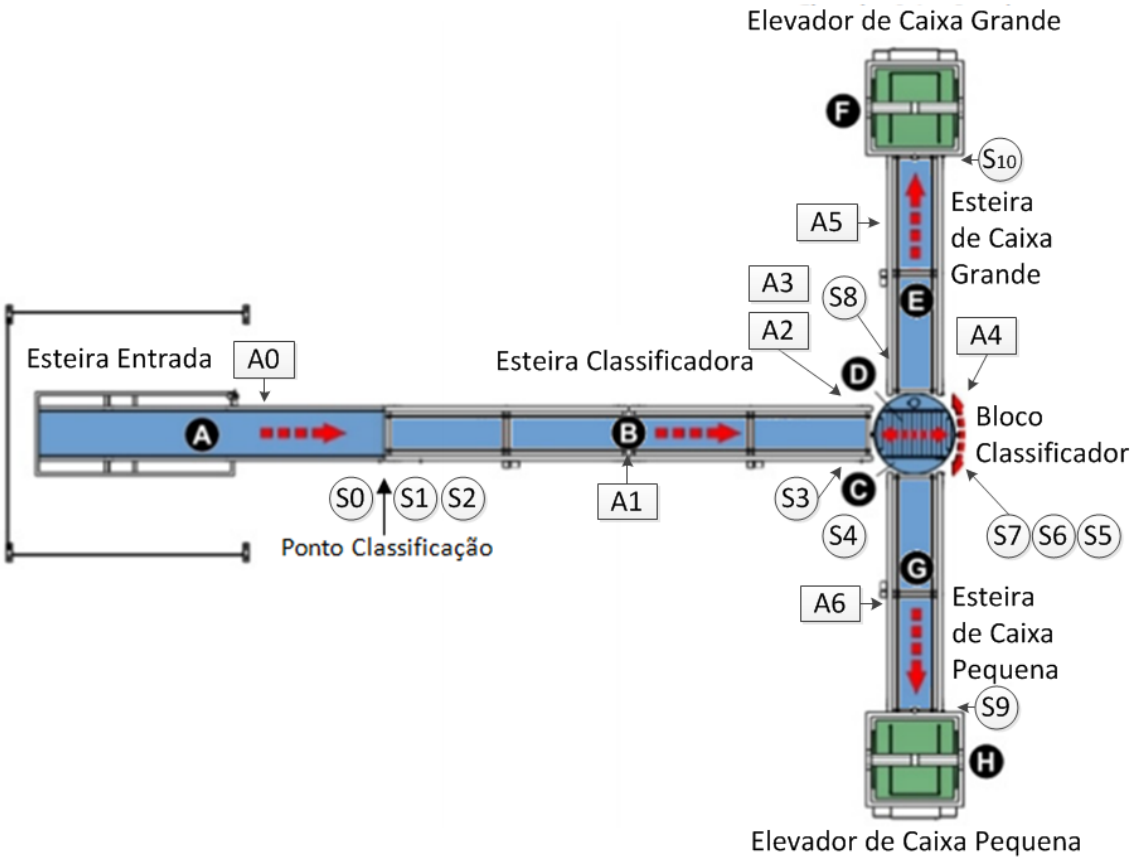


Figura 13 – Sistema de Triagem de Caixas.
Fonte: Autoria própria.

Conforme mostrado na Figura 13, o processo de Triagem de Caixas é iniciado com o acionamento da *Esteira de Entrada* (A), que transporta as caixas até a *Esteira Classificadora* (B), onde é realizada a classificação da caixa. A *Esteira Classificadora* transporta a caixa classificada até o *Bloco Classificador* (C), que realiza um giro de 90 graus. Conforme o tipo de caixa classificada, os roletes (D) do *Bloco Classificador* deverão ser acionados de modo diferente: caso a caixa seja pequena, os roletes deverão levar a caixa em direção do *Elevador de Caixa Pequena* (H) através da *Esteira de Caixa Pequena* (G); caso a caixa seja classificada como grande, os roletes deverão envia-la para o *Elevador de Caixa Grande* (F) através da *Esteira de Caixa Grande* (E).

Os sensores de presença de caixas instalados na planta virtual são responsáveis por repassar ao CLP as informações sobre o sistema. Uma relação dos sensores utilizados é apresentada na Tabela 1.

Tabela 1 – **Relação de Sensores do Sistema**

Sensor	Descrição	Endereço
S0	Saída da *Esteira Entrada*	I0.0
S1	Sensor de caixa baixa	I0.1
S2	Sensor de caixa alta	I0.2
S3	Saída da *Esteira Classificadora*	I0.3
S4	*BC* posição carregar	I0.4
S5	*BC* posição descarregar	I0.5
S6	Caixa no *Bloco Classificador*	I0.6
S7	Entrada caixa *Esteira Pequena*	I0.7
S8	Entrada caixa *Esteira Grande*	I1.0
S9	Saída caixa *Esteira Pequena*	I1.1
S10	Saída caixa *Esteira Grande*	I1.2
B1	Botão iniciar pressionado	I1.4

(BC) - Bloco Classificador

Fonte: Autoria própria.

Para que seja possível a classificação das caixas, ao final da *Esteira de Entrada* há dois sensores ($S1$ e $S2$), que adquirem a altura das caixas que passam da *Esteira de Entrada* à *Esteira Classificadora*.

Os atuadores realizam mudanças nos estados dos elementos que compõem o sistema, avançando esteiras ou realizando giros no *Bloco Classificador*, garantindo a interação direta do CLP com o ambiente controlado. A relação de atuadores utilizados no processo de Triagem de Caixas é apresentada na Tabela 2.

Tabela 2 – **Relação de Atuadores do Sistema**

Atuador	Descrição	Endereço
A0	Avança *Esteira de Entrada*	Q0.0
A1	Avança *Esteira Classificadora*	Q0.1
A2	Avança roletes *BC*	Q0.2
A3	Récua roletes *BC*	Q0.3
A4	Gira *Bloco Classificador*	Q0.4
A5	Avança *Esteira Caixa Grande*	Q0.5
A6	Avança *Esteira Caixa Pequena*	Q0.6
L1	Acende sinal luminoso	Q1.0

(BC) - Bloco Classificador

Fonte: Autoria própria.

Para o controle do processo utiliza-se um CLP modelo $S7300$ da marca *Siemens*, onde o algoritmo de controle resultante da metodologia proposta é gravado e executado. Este CLP está instalado em um módulo didático que possui, para cada entrada e saída do CLP, *bornes* para conexão da placa de condicionamento de sinal, facilitando o acesso aos módulos de *I/O (Input/Output)* do CLP. O esquema de comunicação do CLP ao computador é apresentado na Figura 14.

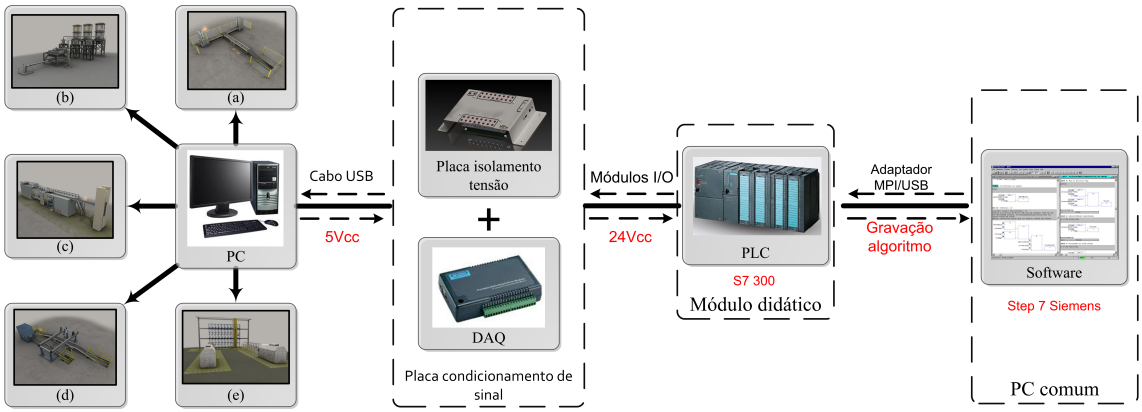


Figura 14 – Esquemático do experimento utilizando o CLP.
Fonte: Autoria própria.

Todos os sistemas simulados podem ser controlados manualmente (através do próprio programa) ou automaticamente (recebendo sinais do CLP). Após a mudança da chave para a posição automático, o controle passa a ser realizado pelo CLP conectado à DAQ. O usuário também pode simular falhas nos sensores e atuadores do sistema, melhorando assim a percepção da evolução do processo que está sendo controlado.

6.3 Aplicação da metodologia ao 1^a caso prático

Para exemplificar a utilização da metodologia na síntese de projetos de automação realiza-se o controle do sistema de Triagem de Caixas, que deverá admitir a entrada de apenas uma caixa para classificação, reiniciando o processo de classificação somente após a passagem da caixa classificada pelos elevadores de saída de caixa grande ou pequena, conforme apresentado na Figura 15. Restringindo a quantidade de caixas admitidas no processo a lógica de controle torna-se mais simples facilitando a explicação da metodologia proposta.

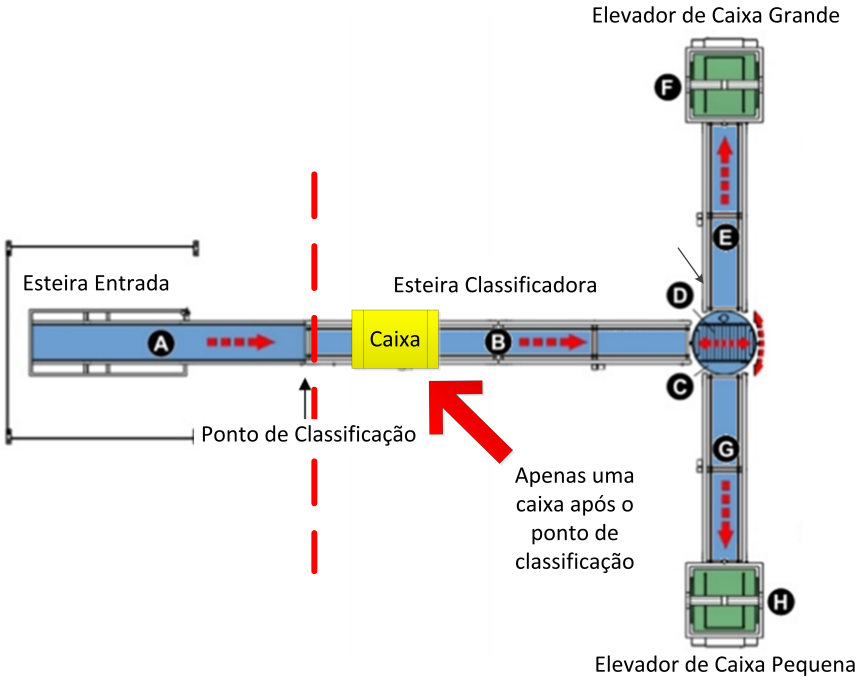


Figura 15 – **Esquemático de representação do 1º caso prático.**
Fonte: Autoria própria.

Com base no sistema e nas especificações de funcionamento, inicia-se o processo de construção do algoritmo de controle considerando as etapas propostas pela metodologia, são elas:

1ª Etapa: na primeira etapa da conversão, cada elemento que constitui o sistema de Triagem de Caixas é modelado individualmente. Para a *Esteira de Entrada* são identificados dois estados: esteira aguardando acionamento, representado no modelo pelo lugar `Esteira_Ent_Parada` e o estado em que a esteira está em funcionamento, descrito no modelo pelo lugar `Esteira_Ent_Operando`. Duas transições (t_0 e t_1) são utilizadas para conectar os estados descritos, logo, obtém-se a RdPI mostrada na Figura 16:

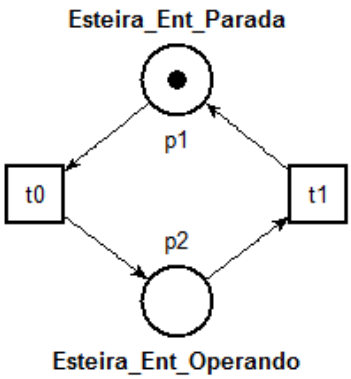


Figura 16 – **RdPI da *Esteira Entrada*.**
Fonte: Autoria própria.

Para o desenvolvimento da RdPI da *Esteira Classificadora* a informação do tipo de caixa que está sobre a esteira é relevante, logo três estados são criados: `Esteira_Class_Transp_Cx_G` para o estado em que a *Esteira Classificadora* transporta caixas do tipo grande; `Esteira_Class_Transp_Cx_P` para o transporte de caixas pequenas e `Esteira_Class_Parada` para o estado ocioso da *Esteira Classificadora*, conforme apresentado na Figura 17.

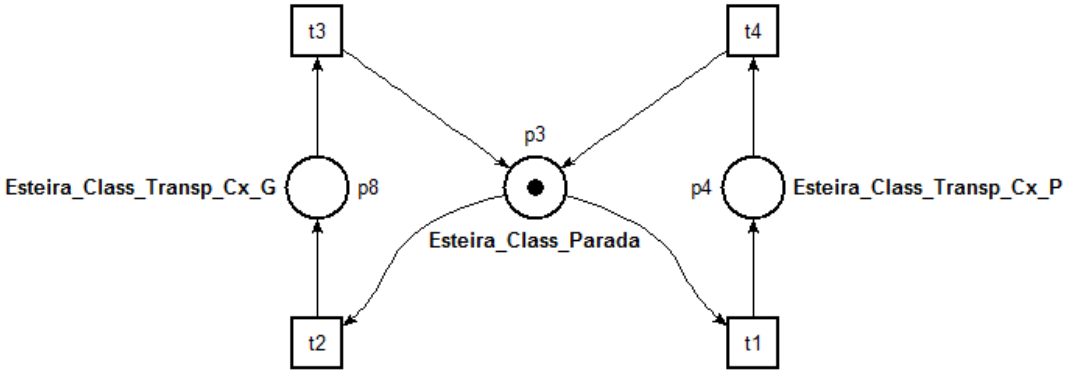


Figura 17 – RdPI da *Esteira de Classificadora*.
Fonte: Autoria própria.

O *Bloco Classificador* é semelhante à *Esteira Classificadora*, pois o tipo de caixa que está sobre o bloco define a direção em que os roletes serão acionados após a conclusão do giro. Assim, a RdPI possui estados que representam tal informação do processo, que são eles: `Bloco_gira_Elev_Grande` que é o estado em que o *Bloco Classificador* está posicionado para enviar a caixa para o *Elevador de Caixa Grande*, `Bloco_gira_Elev_Pequeno` quando o bloco está posicionado para enviar caixa para o *Elevador de Caixa Pequena* e o estado `Bloco_parado`, representando a ociosidade do *Bloco Classificador*. A RdPI do elemento descrito é mostrada na Figura 18.

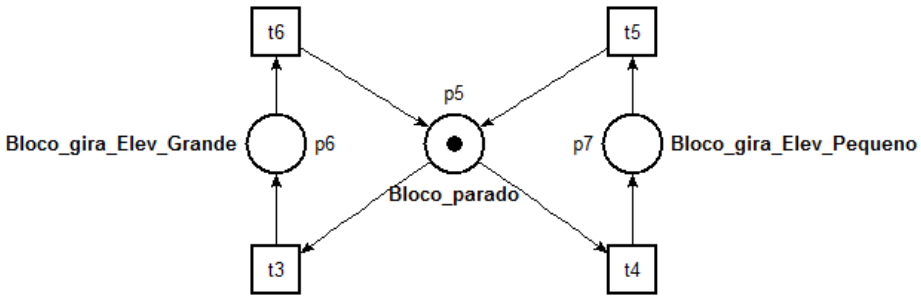
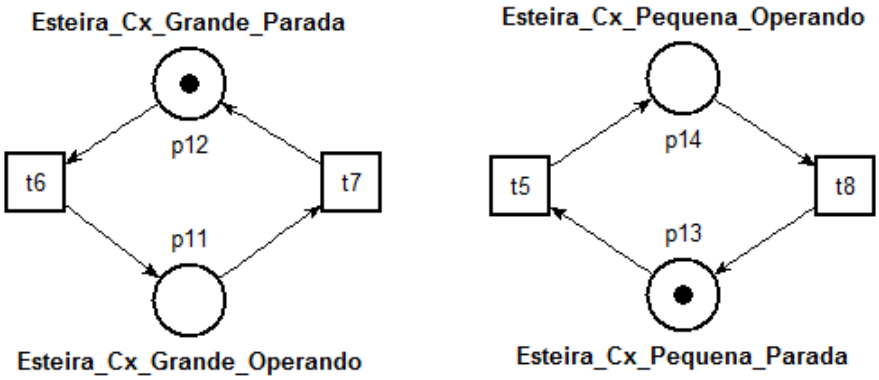


Figura 18 – RdPI do *Bloco Classificador*.
Fonte: Autoria própria.

Os elementos *Esteira de Caixa Grande* e *Esteira de Caixa Pequena* apresentam estados de funcionamento semelhantes, logo as RdPIs que representam a lógica de con-

trole possuem dois lugares: `Esteira_Cx_Grande_Parada` e `Esteira_Cx_Pequena_Parada` para os estados em que as esteiras estão ociosas e `Esteira_Cx_Grande_Operando` e `Esteira_Cx_Pequena_Operando` para quando as esteiras estão avançando. As RdPIs da *Esteira de Caixa Grande* e *Esteira de Caixa Pequena* são apresentadas na Figura 19(a) e 19(b), respectivamente.



(a) Esteira Caixas Grande **(b) Esteira Caixas Pequena**

Figura 19 – RdPI de controle das *Esteiras de Caixas Grande e Pequena*.
Fonte: Autoria própria.

2ª Etapa: após a modelagem da lógica de controle de cada elemento que compõe o sistema, realiza-se a união das RdPI considerando as transições que são compartilhadas pelos elementos e a especificação de funcionamento do processo. A RdPI de controle do processo é apresentada na Figura 20.

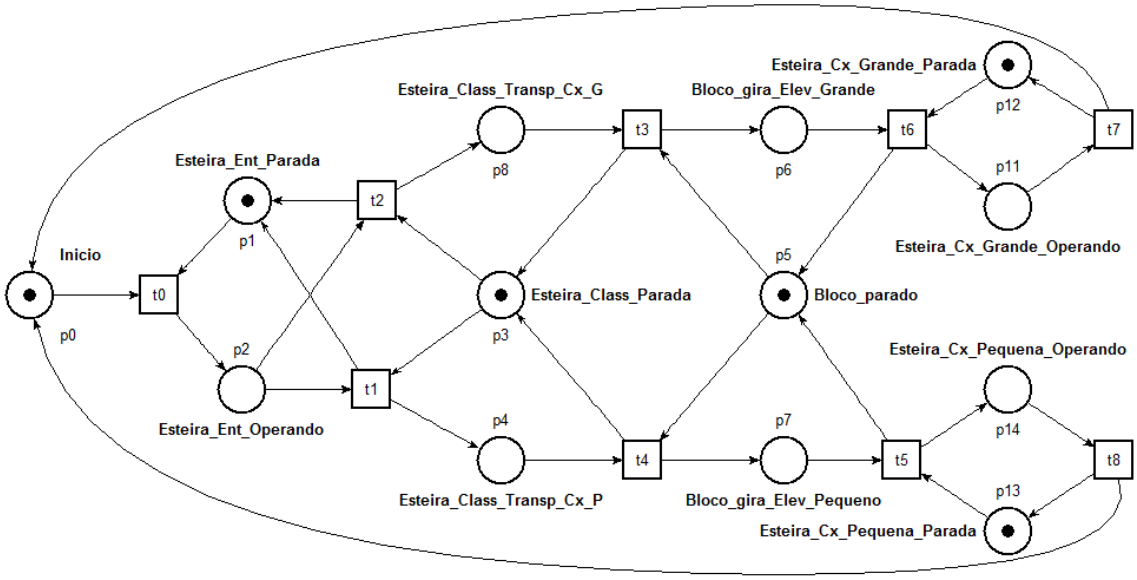


Figura 20 – RdPI do sistema de Triagem de Caixas para 1ª caso prático.
Fonte: Autoria própria.

Utilizando a RdPI de controle e as especificações de funcionamento é necessário analisar todas as transições contidas no modelo, buscando definir qual a condição que deverá ser satisfeita e a ação associada ao disparo da transição. Para exemplificar, utiliza-se a transição t_0, que quando disparada representa a mudança de estado da *Esteira de Entrada*, que inicialmente está na condição parada. Sendo assim, uma condição que deve ser satisfeita para que a *Esteira de Entrada* entre em operação é o acionamento do Botão B1 (*B1 = 1*). Logo, esta condição deve ser agregada à transição t_0. Após o acionamento do Botão B1, a *Esteira de Entrada* entrará em funcionamento e então, a ação associada ao disparo da transição t_0 será o avanço da esteira (*est_entr = 1*). Todas as transições da RdPI de controle do processo e suas condições e ações estão organizadas em uma tabela de dados, mostrada na Tabela 3.

Tabela 3 – Tabela de Dados - Descrição das Condições e Ações

t_n	**Acontecimento**	**Condições ; Ações (C_i ; A_i)**
t_0	Apertar_B1	(B1 = 1 ; est_ent = 1)
t_1	Avanc_Est_Class_P	($\downarrow$S0 = 1 $\wedge$ $\downarrow$S2 = 0 ; est_ent = 0 $\wedge$ est_class = 1 $\wedge$ Cx_P = 1)
t_2	Avanc_Est_Class_G	($\downarrow$S0 = 1 $\wedge$ $\downarrow$S2 = 1 ; est_ent = 0 $\wedge$ est_class = 1 $\wedge$ Cx_G = 1)
t_3	Carrega_Bloco_G	($\downarrow$S3 = 1 $\wedge$ Cx_G = 1 ; est_class = 0 $\wedge$ rolo_avanc = 1)
t_4	Carrega_Bloco_P	($\downarrow$S3 = 1 $\wedge$ Cx_P = 1 ; est_class = 0 $\wedge$ rolo_avanc = 1)
t_5	Descarrega_Bloco	($\downarrow$S7 = 1 ; est_peq = 1 $\wedge$ rolo_avanc = 0 $\wedge$ Cx_P = 0)
t_6	Descarrega_Bloco	(S8 = 1 ; est_grande = 1 $\wedge$ rolo_avanc = 0 $\wedge$ Cx_G = 0)
t_7	Despacha_caixa	($\downarrow$S10 = 1 ; est_grande = 0)
t_8	Despacha_caixa	($\downarrow$S9 = 1 ; est_peq = 0)

Fonte: Autoria própria.

Com base na RdPI de controle do processo de Triagem de Caixas, realiza-se a análise das boas propriedades da rede, considerando suas propriedades estruturais e marcações.

3ª,4ª e 5ª Etapas: nestas etapas a RdPI é analisada, simulada e corrigida buscando-se identificar possíveis problemas antes da implementação do algoritmo de controle.

Analisando a RdPI gerada para o controle do processo de Triagem de Caixas, observa-se que a rede é viva, logo, todas as transições contidas no modelo são sensibilizadas através de uma sequência de disparo conhecida. A RdPI analisada também é definida como limitada, pois o número de marcações da rede permanece constante consi-

derando a sua marcação inicial. A rede é reiniciável, pois partindo da marcação inicial da rede existe uma sequência de disparo que retorna a sua marcação inicial.

Analisando as propriedades estruturais do modelo é possível identificar informações relacionadas à dinâmica do sistema, como a sequência de disparo das transições, para que seja realizada uma tarefa relevante. Realizando uma análise estrutural da RdPI de controle identifica-se um invariante de transição, que descreve a sequência de disparo para classificação das caixas do tipo pequena (t_0, t_1, t_4, t_5, t_8) e do tipo grande (t_0, t_2, t_3, t_6, t_7).

Através da análise estrutural também é possível identificar o invariante de lugar, que descreve os elementos conservativos presentes no sistema. A RdPI apresentou como invariante os lugares mostrados na Tabela 4.

Tabela 4 – Relação dos invariantes de lugar

Lugares	Elemento do sistema
p_1, p_2	*Esteira de Entrada*
p_3, p_4, p_8	*Esteira Cassificadora*
p_5, p_6, p_7	*Bloco Classificador*
p_{13}, p_{14}	*Esteira Caixa Pequena*
p_{11}, p_{12}	*Esteira Caixa Grande*

Fonte: Autoria própria.

Concluída a etapa de análise, realiza-se a simulação da RdPI. Nesta etapa, para cada disparo das transições é necessário verificar na tabela de dados quais as condições e ações envolvidas, buscando avaliar a dinâmica do processo e possíveis problemas lógicos.

Caso seja necessário é possível realizar correções no modelo gerado, garantindo que as boas propriedades sejam mantidas e as especificações de funcionamento executadas.

6 e 7ª Etapas: após a análise das boas propriedades do modelo, identifica-se com o uso da RdPI de controle e da tabela de dados quais as condições e ações que estão atribuidas a cada elemento que compõe o sistema. Após a identificação das condições e ações em comum, criam-se módulos de controle descritos em Grafcet. Cada módulo é responsável pelo acionamento e desacionamento de apenas um elemento do sistema.

Para que a *Esteira de Entrada* entre em funcionamento é necessário que o Botão B1 (transição t_0) seja pressionado, sendo assim, a condição lógica ($B1 = 1$) deve ser satisfeita para que a esteira inicie o avanço (est_ent = 1). Já para que a *Esteira de Entrada* retorne à condição parada, um sinal representando que a caixa classificada saiu da esteira é necessário. As transições t_1 e t_2 representam o acontecimento deste evento, retornando a marcação da rede para o estado onde as esteiras estão ociosas, sendo necessário uma

simplificação da expressão booleana ↓S0 = 1 ∧ ↓S2 = 0 e ↓S0 = 1 ∧ ↓S2 = 1, resultando na condição ↓S0 = 1.

Com base nas condições e ações associadas a *Esteira de Entrada*, cria-se o Grafet de controle, onde as condições de início e fim de operação são as receptividades das transições. O Grafcet de controle da *Esteira de Entrada* é mostrado na Figura 21.

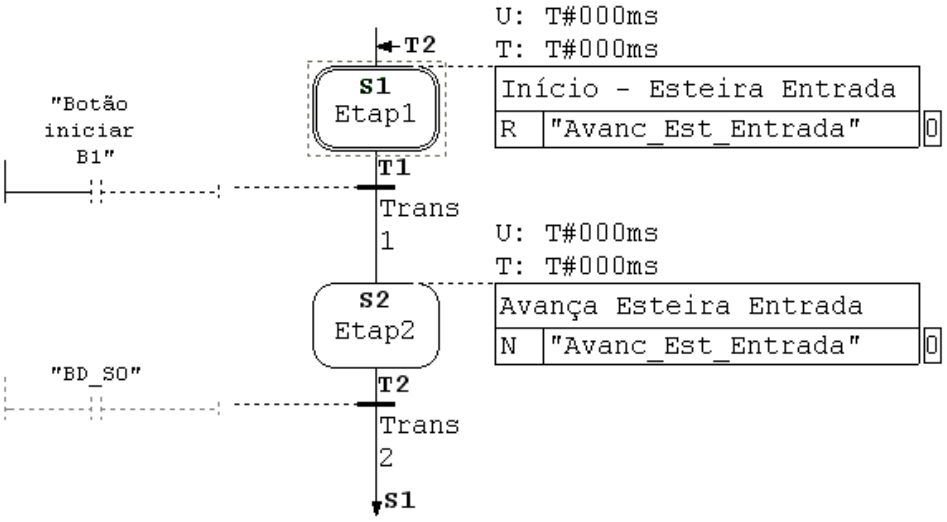


Figura 21 – SFC de controle da *Esteira de Entrada*.
Fonte: Autoria própria.

Na Figura 21 a condição "BD_S0" contida na transição $T2$ significa que o sinal S0 deve passar de 1 para 0 para que a condição seja verdadeira, isto é, uma borda de descida.

Para o acionamento da *Esteira Classificadora* estão associadas as transições t_1 e t_2, realizando uma simplificação das expressões booleanas ↓S0 = 1 ∧ ↓S2 = 0 e ↓S0 = 1 ∧ ↓S2 = 1, obtém-se ↓S0 = 1, mas para implementação desconsidera-se o sinal de borda de descida do sensor $S0$. Para voltar a *Esteira Classificadora* ao estado ocioso, a caixa que está sobre a esteira deve chegar ao *Bloco Classificador*, o sinal ↓S3=1 representa este evento. Desta forma, estas condições e ações serão descritas no Grafcet de controle, mostrado na Figura 22.

Para que o *Bloco Classificador* inicie o processo de carga da caixa classificada é necessário que a caixa passe pelo sensor de presença de caixas ($S3$). Esta condição é identificada realizando-se a simplificação da expressão booleana associada às transições t_3 e t_4. Para que o *Bloco Classificador* retorne ao estado parado, a caixa que está sobre o bloco deverá ser repassada para a *Esteira de Caixa Grande* ou *Pequena*.

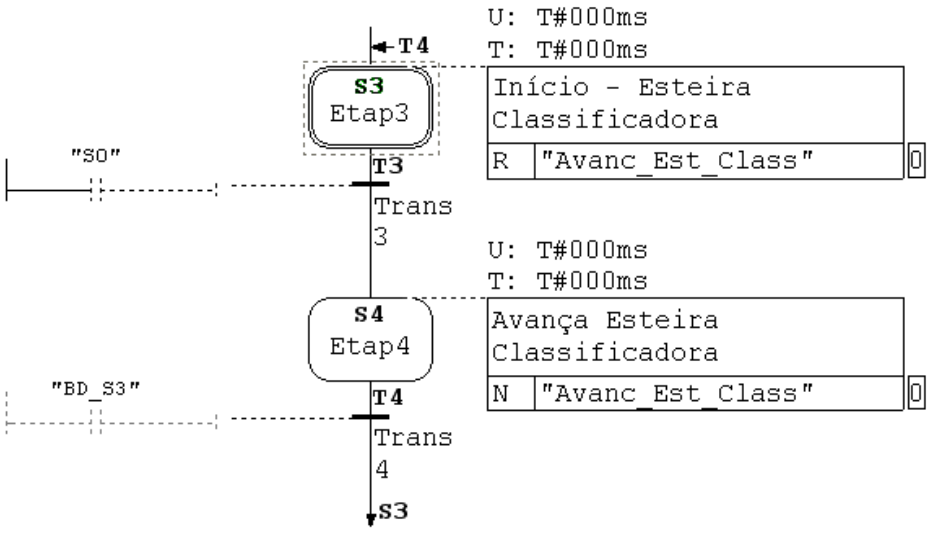


Figura 22 – **SFC de controle da *Esteira Classificadora*.**
Fonte: Autoria própria.

Tais condições são apresentadas nas transições t_5 e t_7 e, realizando-se a simplificação das expressões, tem-se que $\downarrow$S7 = 1 $\vee$ $\downarrow$S8 = 1. O Grafcet de controle baseado nestas condições e ações é mostrado na Figura 23.

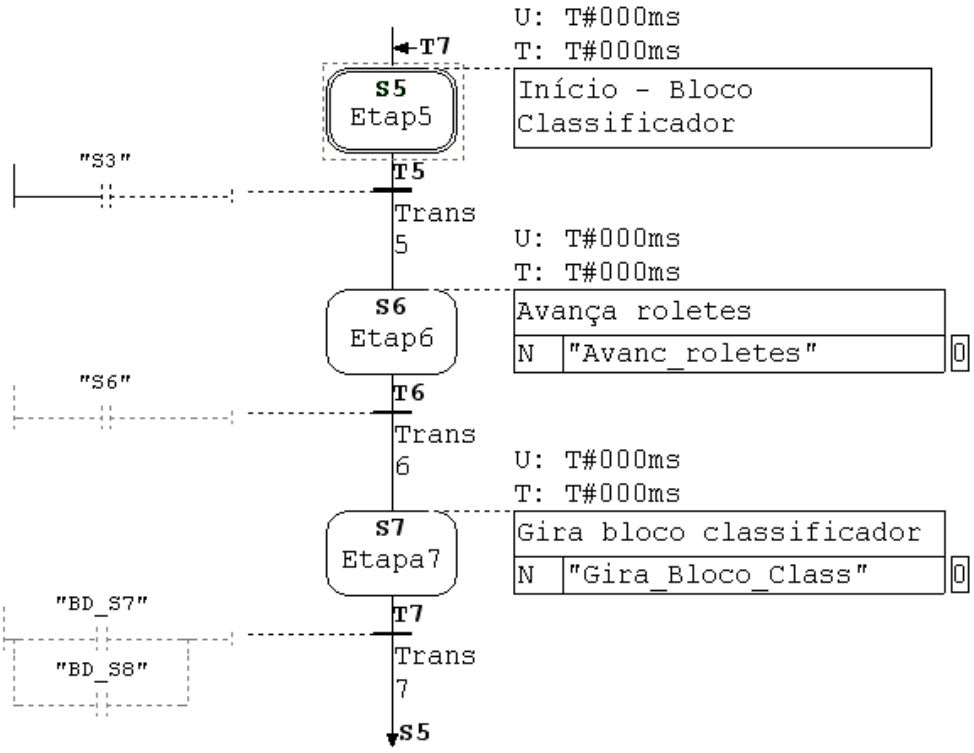


Figura 23 – **SFC de controle do *Bloco Classificador*.**
Fonte: Autoria própria.

A *Esteira de Caixa Grande* iniciará seu avanço após a entrada de uma caixa sobre

a esteira. Esta condição está agregada à transição t_6, que possui a condição S8 = 1. Para o retorno da esteira ao estado parada é necessário que a caixa saia completamente da esteira, este evento é descrito pela transição t_7, onde a expressão que garante esta condição é ↓S10 = 1. Considerando as condições e ações da *Esteira de Caixa Grande* cria-se o Grafcet de controle mostrado na Figura 24.

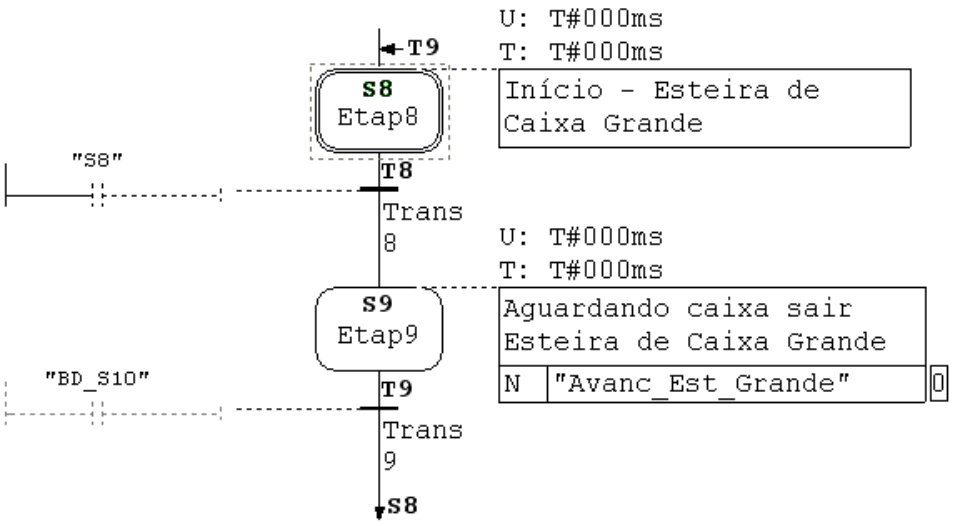


Figura 24 – Grafcet de controle *Esteira de Caixa Grande*.
Fonte: Autoria própria.

A *Esteira de Caixa Pequena* começará a avançar somente com caixa sobre a mesma, tal condição é descrita na transição t_5 que está agregada à condição S7 = 1, e para o retorno ao estado ocioso, a transição t_8, que possui a condição ↓S9 = 1, deve ser satisfeita. O Grafcet que realiza o controle da *Esteira de Caixa Pequena* considerando estas condições é mostrado na Figura 25.

Observando, a análise dos invariantes de transição da RdPI do processo, é possível identificar dois conjuntos de transições $(t_0, t_1, t_4, t_5, t_8)$ e $(t_0, t_2, t_3, t_6, t_7)$ o primeiro conjunto de transições representa a classificação de uma caixa do tipo pequena e o segundo conjunto a triagem de uma caixa grande. Com base nestas informações, cria-se um Grafcet do *Módulo de Classificação* que armazena o tipo da caixa classificada e realiza o acionamento dos roletes do *Bloco Classificador*, direcionando a caixa classificada para o elevador correto.

Este *Módulo de Classificação* é criado utilizando as transições que indicam o tamanho da caixa sobre a *Esteira de Classificação*, as transições t_1 e t_0 apresentam o contador "Cx_P" e "Cx_G", que indicam o tipo de caixa. Após a classificação da caixa, os roletes do *Bloco Classificador* deverão ser acionados conforme a especificação de funcionamento. Para a caixa pequena os roletes deverão recuar até que a caixa saia (↓S7 = 1) do *Bloco*

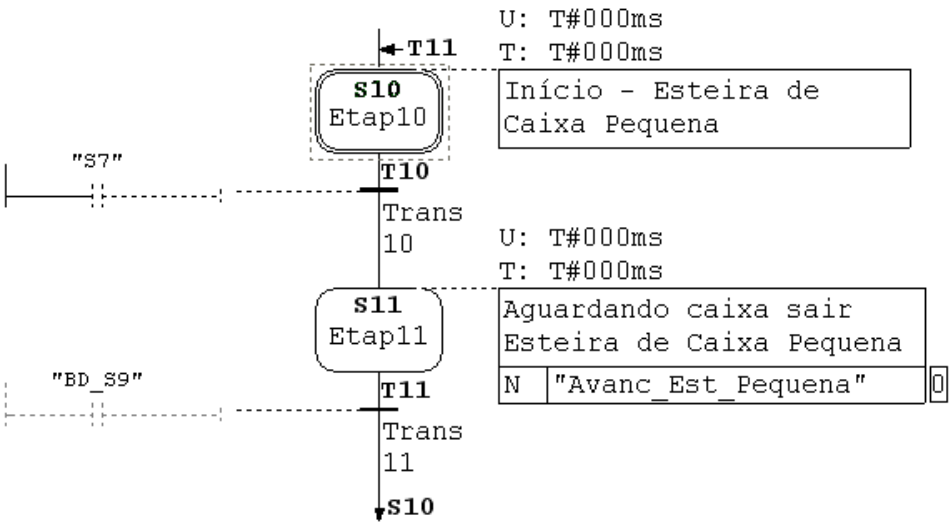


Figura 25 – **SFC de controle da *Esteira de Caixa Pequena*.**
Fonte: Autoria própria.

Classificador. Para caixas grandes os roletes deverão avançar até que a caixa esteja fora do bloco (↓S8 = 1). O Grafcet para o módulo de classificação é mostrado na Figura 26.

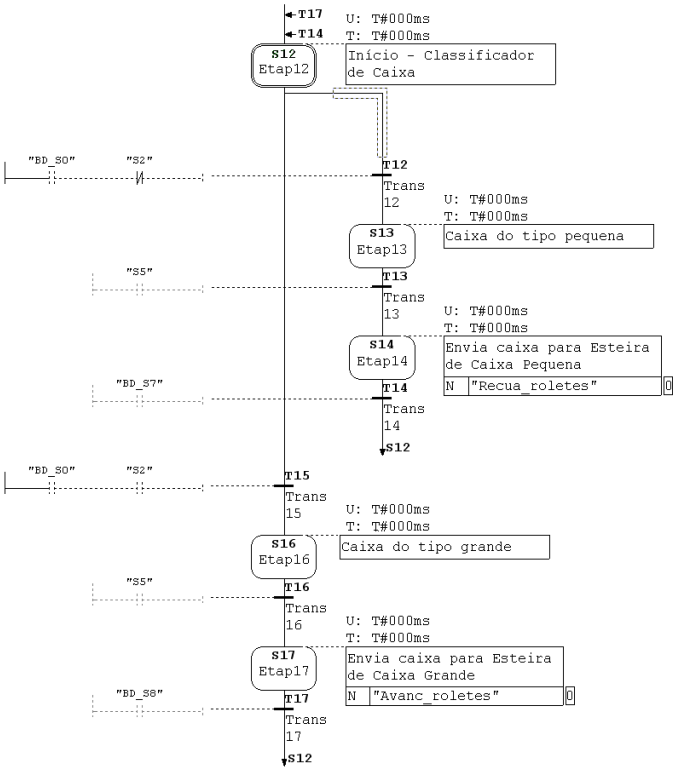


Figura 26 – **SFC de controle do *Módulo de Classificação*.**
Fonte: Autoria própria.

Nesta etapa de conversão é possível inserir ou remover elementos de apoio, como os contadores e detectores de borda de subida, para garantir a perfeita implementação do

controle. A abstração de algumas etapas do sistema também é possível, permitindo a criação de um modelo mais legível.

8ª Etapa: com base nos Grafcets gerados e na RdPI que representa o processo é necessário marcar as etapas iniciais dos Grafcets. Essa marcação é realizada observando-se os elementos modelados na primeira etapa da conversão, sendo necessária para que o sistema controlado inicie em estado conhecido pelo projetista.

9ª Etapa: utilizando todos os Grafcets gerados nas etapas anteriores realiza-se a implementação em um CLP, que realizará o controle do processo. Desta forma, todos os Grafcets serão executados simultaneamente, proporcionando o controle individual de cada elemento do sistema, conforme mostrado na Figura 27.

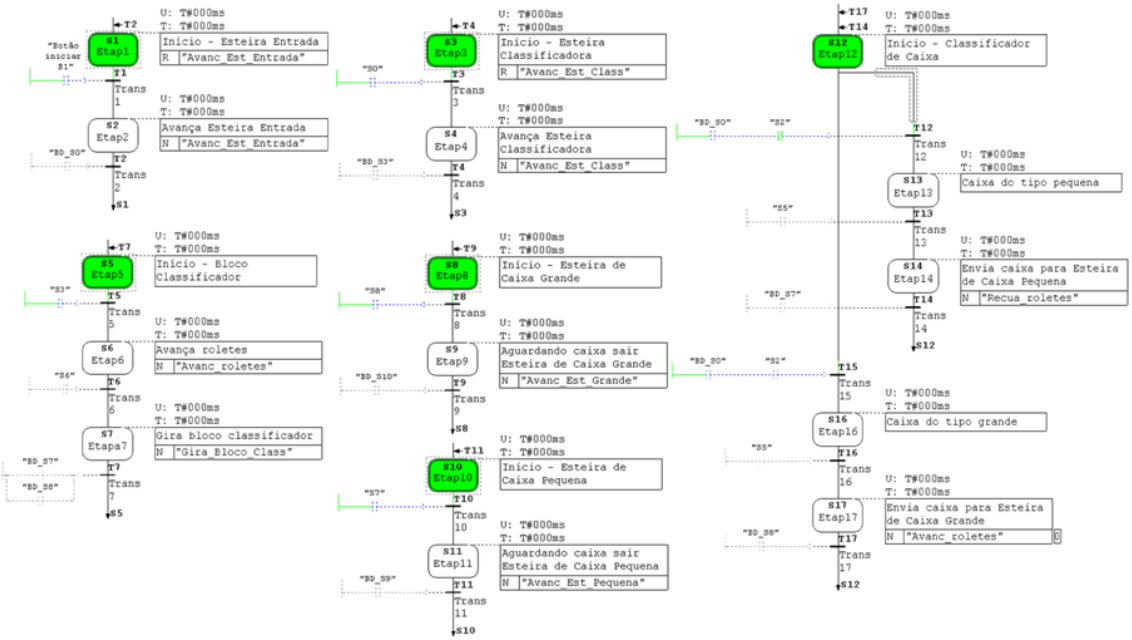


Figura 27 – SFC de controle do processo de Triagem de caixas.
Fonte: Autoria própria.

Com base na Figura 27 é possível observar as etapas destacadas na cor verde simbolizando o ponto inicial de cada SFC de controle.

6.4 Aplicação da metodologia ao 2ª caso prático

Buscando validar a aplicação da metodologia para outras especificações de funcionamento realiza-se o controle do sistema de Triagem de Caixas modificando a quantidade

de caixas processadas simultaneamente pelo sistema, restringindo o número de caixas processadas em dois. Desta forma, o segundo caso prático possui uma complexidade maior, conforme apresentado na Figura 28.

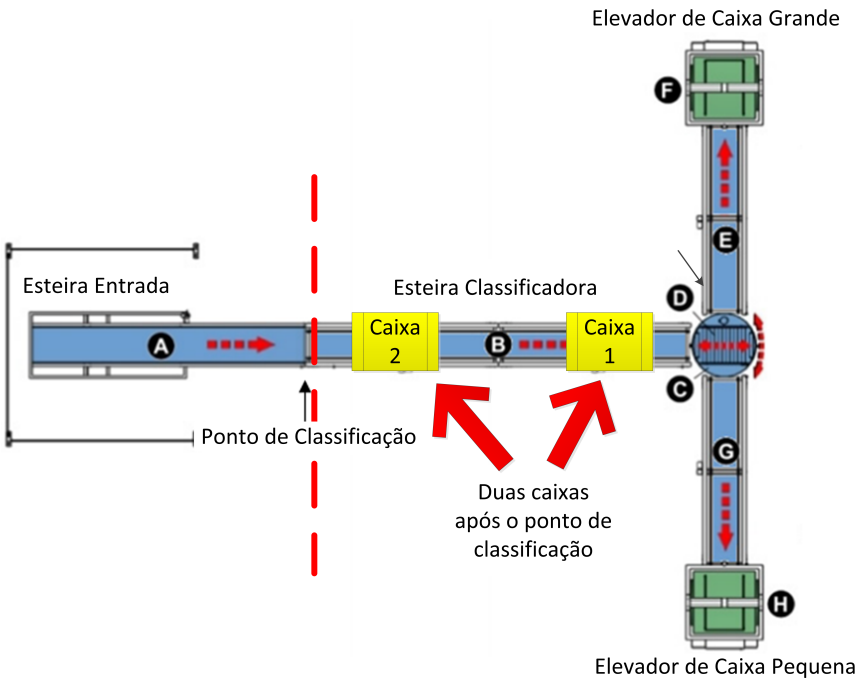


Figura 28 – Esquemático de representação do 2º caso prático.
Fonte: Autoria própria.

As etapas propostas na metodologia foram executadas novamente com o objetivo de gerar um novo modelo descrito através das Redes de Petri Interpretadas representando o sistema e as novas especificações de funcionamento. Realizando a 1ª e 2ª etapas obtemos as RdPI geral mostrada na Figura 38.

Na 3ª,4ª e 5ª etapas a RdPI geral é analisada buscando a correção de possíveis problemas lógicos, sendo que a tabela de dados é fundamental para o acompanhamento da dinâmica do sistema. A análise estrutural da rede é importante para a identificação dos invariantes de transição e lugar.

Para a 6ª e 7ª etapas utiliza-se a RdPI geral e a tabela de dados, dando origem aos módulos de controle para cada elemento do sistema. Este módulos são descritos em Grafcets e as condições associadas às transições são retiradas da tabela de dados. Ao final desta etapas obtem-se 7 módulos, sendo dois módulos responsáveis por armazenar as informações referentes ao tipo da caixa classificada.

Após a implementação dos SFCs CLP é possível visualiza-se o adequado funcionamento do processo de Triagem de Caixas, admitindo o processamento de duas caixas simultaneamente sem problemas na lógica de controle.

6.5 Plataforma utilizando FPGA

Outra tecnologia utilizada para a validação da metodologia é o FPGA, devido a limitação do CLP quando aplicado ao controle de processos concorrentes. Tal estratégia possui a vantagem de executar o controle do processo de forma simultânea, pois todos os eventos de um SED evoluem paralelamente seguindo a dinâmica do processo em questão. Em alguns casos, devido ao alto grau de paralelismo presente no processo, torna-se inviavél a utilização do CLP como equipamento de controle.

Buscando contornar esta limitação do CLP, alguns trabalhos apresentam técnicas para conversão das sequências de controle implementadas em CLP para conjuntos de circuitos lógicos ou diretamente para a linguagem de especificação VHDL (WEGRZYN; ADAMSKI; MONTEIRO, 1998; ECONOMAKOS; ECONOMAKOS, 2008).

Outros trabalhos propõem a conversão baseando-se no modelo representativo do processo (ICHIKAWA et al., 2006; OLIVEIRA, 2008), que possui um poder de abstração maior quando comparado com as linguagens de programação de CLPs.

Neste trabalho, os módulos de controle gerados para os dois casos práticos são convertidos para Máquinas de Estados Finitos, sendo implementadas no equipamento de controle.

O algoritmo gerado é gravado diretamente no FPGA Altera Cyclone II, apresentado na Figura 29.

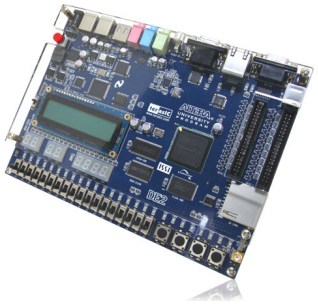


Figura 29 – Placa de desenvolvimento Altera DE2 Cyclone II.
Fonte: Altera ().

O FPGA é responsável por realizar o controle do processo, sendo que os módulos de controle são convertidos em Máquinas de Estados Finitos descritas sobre a linguagem de especificação VHDL e executados. Um esquemático mostrado na Figura 30 ilustra as conexão entre os equipamentos utilizados.

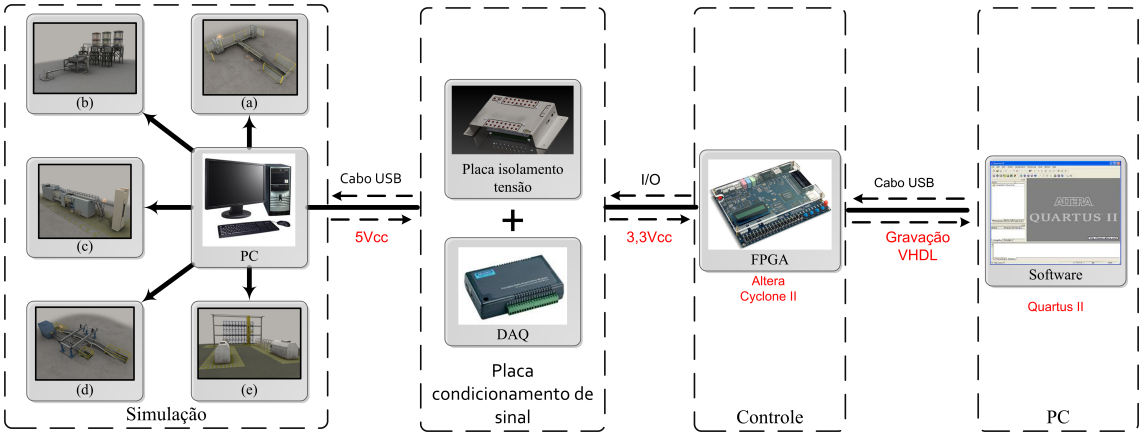


Figura 30 – Esquemático do experimento utilizando o FPGA.
Fonte: Autoria própria.

Utilizando a linguagem de especificação de *hardware* VHDL realiza-se a conversão dos módulos para Máquinas de Estados Finitos que realizam o controle do processo de Triagem de Caixas. A Máquina de Estado Finito que realiza o controle da *Esteira de Entrada* é descrita no Quadro 2, expresso sobre a linguagem de especificação VHDL:

```
1              -- INÍCIO MÓDULO 1 - ESTEIRA DE ENTRADA
2              CASE MODULO1 IS
3                  WHEN ETAPA1 = > -- ETAPA 1
4                      IF BOTAO_INICIAR = '1' THEN
5                          MODULO1 < = ETAPA2;
6                          AVANCA_EST_ENT < = '1';
7                          LIGA_LED_G_0   < = '1';
8                      END IF;
9                  WHEN ETAPA2 = > -- ETAPA 2
10                     IF SAIDA_CX_EST_ENT_FALL = '1' THEN
11                         MODULO1         < = ETAPA1;
12                         AVANCA_EST_ENT < = '0';
13                         LIGA_LED_G_0   < = '0';
14                     END IF;
15                 WHEN OTHERS = >
16                     MODULO1 < = ETAPA1;
```

Quadro 2 – Algoritmo exemplo.
Fonte: Autoria própria.

Para o módulo de controle da *Esteira de Entrada* obtemos a Máquina de Estados Finitos apresentada na Figura 31.

Com base na Figura 31 pode-se identificar o arco que interliga a $ETAPA1$ a $ETAPA2$ que possui a condição associada de $(B1 = 1; est\_ent = 1)$ e o arco posterior possui a condição ($\downarrow$S0 = 1 $\wedge$ $\downarrow$S2 = 0).

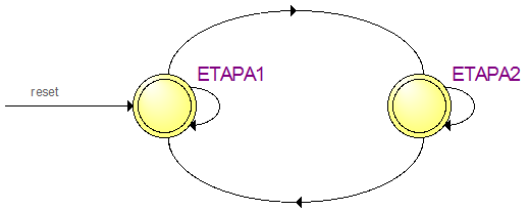


Figura 31 – **Máquina de estados finitos do módulo de controle da *Esteira de Entrada*.**
Fonte: Autoria própria.

Para o módulo de controle da *Esteira Classificadora* temos a Máquina de Estado Finito apresentada na Figura 32.

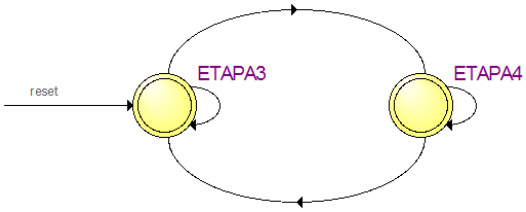


Figura 32 – **Máquina de estados finitos do módulo de controle da *Esteira Classificadora*.**
Fonte: Autoria própria.

Com base na Figura 32 o arco que interliga a $ETAPA3$ a $ETAPA4$ possui a condição ($\downarrow$S0 = 1 $\wedge$ $\downarrow$S2 = 1) e o outro arco a condição (S3 = 1 $\wedge$ Cx_G = 1).

Para o módulo de controle do *Bloco Classificador* temos a Máquina de Estado Finito apresentado na Figura 33.

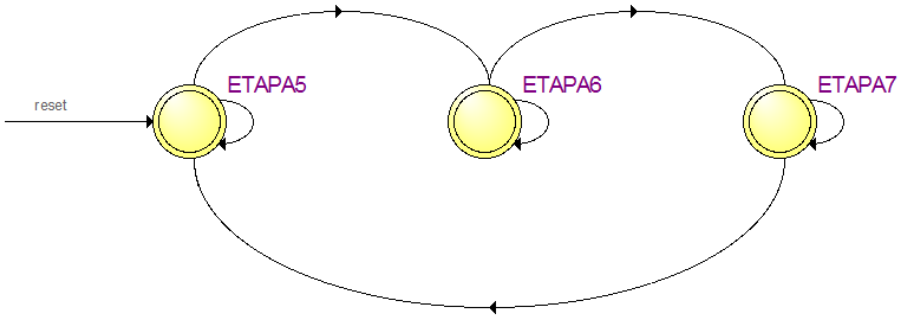


Figura 33 – **Máquina de estados finitos do módulo de controle *Bloco Classificador*.**
Fonte: Autoria própria.

Com base na Figura 33 o arco que interliga a $ETAPA5$ a $ETAPA6$ possui a condição (S3 = 1) e o arco da $ETAPA7$ para a $ETAPA5$ temos (S7 = 1$wedge$S8 = 1).

A Máquina de Estados Finitos apresentada na Figura 34 representa o módulo de controle da *Esteira de Caixa Grande*.

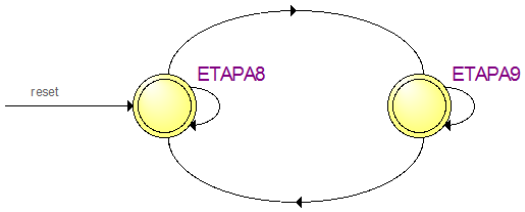


Figura 34 – Máquina de estados finitos do módulo de controle da *Esteira Caixa Grande*.
Fonte: Autoria própria.

Com base na Figura 34 o arco que liga a $ETAPA8$ a $ETAPA9$ possui a condição (S8 = 1) e o outro arco a condição (↓S10 = 1).

A Máquina de Estados Finitos apresentada na Figura 35 representa o módulo de controle da *Esteira de Caixa Pequena*.

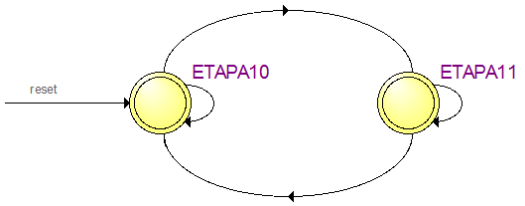


Figura 35 – Máquina de estados finitos do módulo de controle *Esteira Caixa Pequena*.
Fonte: Autoria própria.

Com base na Figura 35 o arco que liga a $ETAPA10$ a $ETAPA11$ possui a condição (S7 = 1) e o outro arco a condição (↓S9 = 1).

O módulo classificador de caixa que armazena o tipo da caixa classificada é convertido na Máquina de Estados Finitos apresentada na Figura 36. Os arcos utilizados para ligar as $ETAPA12$ a $ETAPA16$ associam as condições que representam o tipo da caixa classificada.

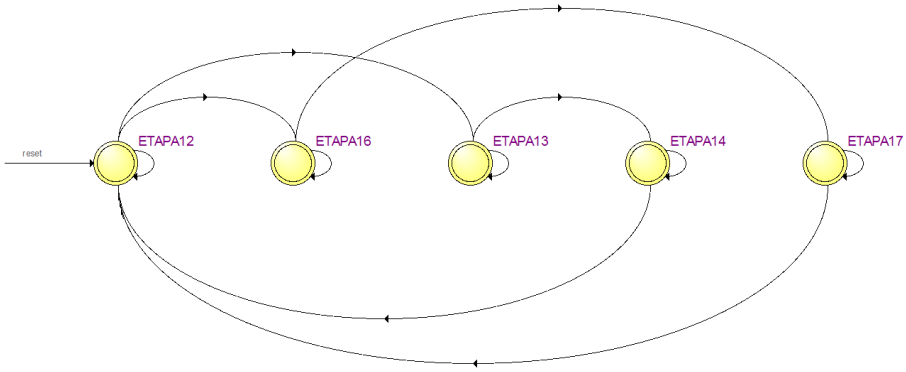


Figura 36 – Máquina de estados finitos do primeiro *Módulo Classificador*.
Fonte: Autoria própria.

Para o módulo de classificação do segundo caso prático obtém-se a Máquina de Estados Finito apresentada na Figura 37.

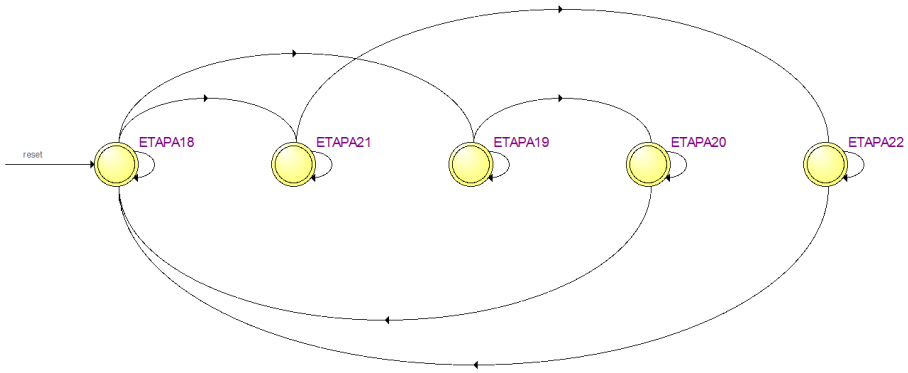


Figura 37 – Máquina de estados finitos do segundo *Módulo Classificador*.
Fonte: Autoria própria.

Após a implementação das Máquinas de Estados Finito no FPGA o processo de Triagem de Caixas foi executado de acordo com esquemático apresentado na Figura 30, onde as boas propriedades analisadas no modelo estão mantidas e as especificações de funcionamento executadas.

6.6 Análise dos Resultados

Para possibilitar a comparação dos resultados obtidos para os dois casos práticos apresentados anteriormente, organizou-se as informações relevantes ao projeto sendo apresentadas na Tabela 5.

Observa-se na Tabela 5 um aumento na quantidade de lugares e transições descritos na RdPI geral resultando em uma diminuição na legibilidade da rede. Este efeito é proporcional a aumento da complexidade das especificações de funcionamento, sendo que o sistema poderá assumir novos estados e, consequente, o modelo representativo deverá descreve-los.

Tabela 5 – Detalhes dos projetos

Dados	1ª **caso prático**	2ª **caso prático**
É passível de simulação	sim	sim
Possibilita correções em tempo de projeto	sim	sim
Atende as especificações de funcionamento	sim	sim
Número de lugares gerados na RdPI	13	20
Número de transições geradas na RdPI	9	15
Algoritmo gerado é legível	sim	sim
Algoritmo gerado é livre de *deadlocks*, *livelocks* e *conflitos*	sim	sim
Módulos de controle construídos	6	7
Complexidade atribuída a especificação de funcionamento	baixa	média

Fonte: Autoria própria.

Para os dois casos práticos apresentados a metodologia proposta obteve resultados satisfatórios, garantindo a geração de algoritmos livres de *deadlocks* e conflitos e mantendo a sua legibilidade e reutilazação.

6.7 Conclusão

Neste capítulo foram apresentados os detalhes da metodologia proposta para síntese de projetos de automação industrial, utilizando o controle de um processo de Triagem de Caixas para validar tal proposta. A metodologia foi aplicada para dois casos práticos, o primeiro caso prático foi utilizado para exemplificar a utilização da metodologia, o segundo caso prático possui especificações de funcionamento mais complexas, gerando uma RdPI geral com um número maior de lugares e transições quando comparado com o primeiro caso. Este aumento de complexidade também é visto no algoritmo de controle, mas a corretude lógica é mantida em ambos os casos.

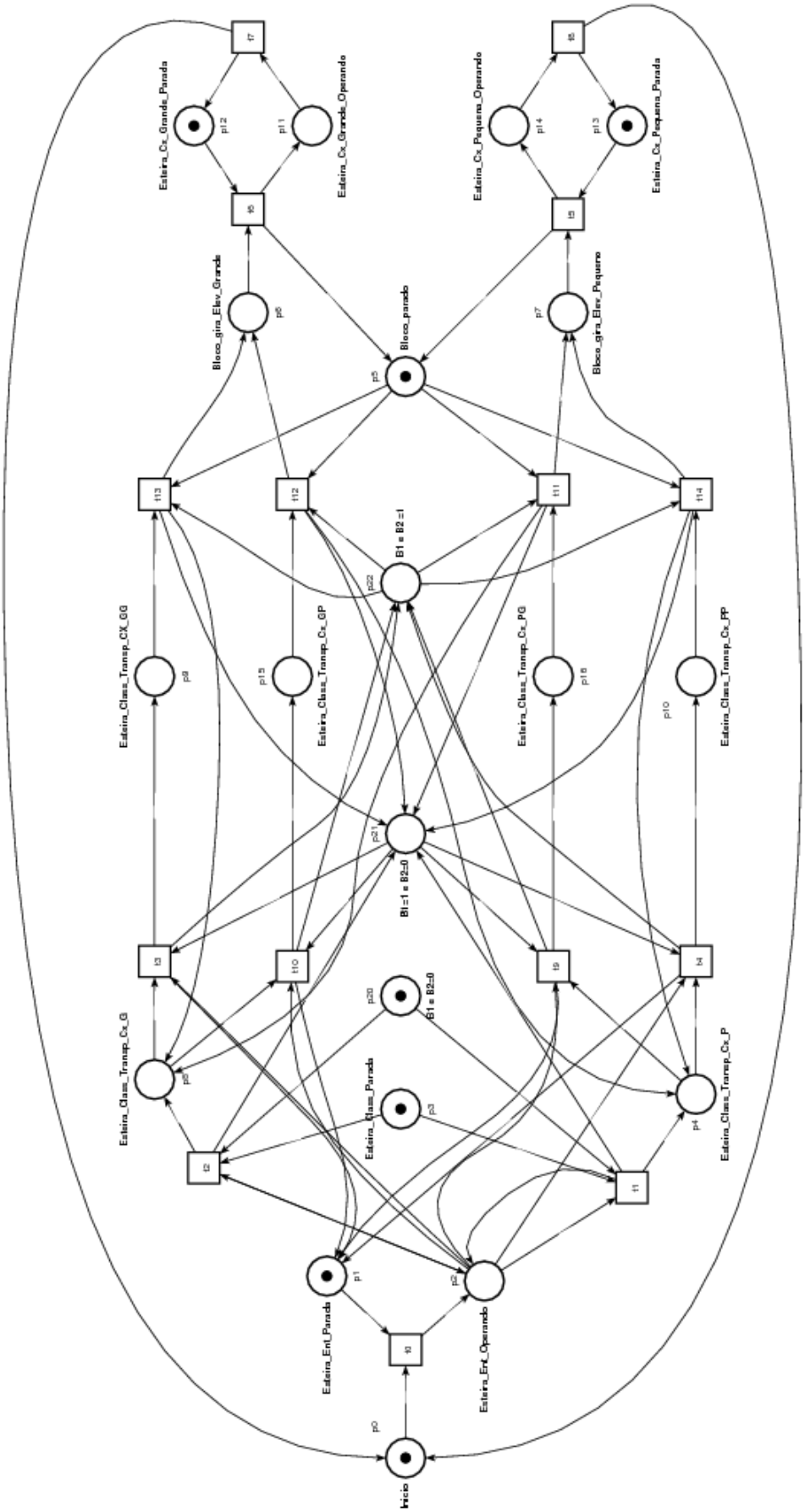


Figura 38 – RdPl do sistema de Triagem de Caixas para 2º caso prático
Fonte: Autoria própria.

7 CONCLUSÃO

Este trabalho apresentou uma metodologia que visa contribuir para as técnicas de desenvolvimento de projetos de automação industrial, propondo etapas bem definidas para a construção de algoritmos de controle legíveis e reutilizáveis.

Tal metodologia concilia as vantagens da utilização dos métodos formais para representação da dinâmica de um sistema a eventos discretos com as necessidades de ferramentas para simulação e análise por parte do projetista, garantido que em tempo de projeto possam ser realizados correções e testes na lógica de controle, procedimentos que garantem a construção de algoritmos de controle livres de *deadlocks*, *livelocks* e conflitos.

Para validar a metodologia foram apresentados os detalhes da síntese do algoritmo de controle de dois casos práticos, onde o primeiro caso possui uma especificação de funcionamento mais simples visto que o objetivo para este caso prático foi exemplificar a utilização da metodologia. Para o segundo caso prático as especificações de funcionamento são mais complexas, evidenciando a generalidade da metodologia quando aplicada a projetos de automação industrial com um maior nível de complexidade, mantendo a corretude lógica do algoritmo de controle sintetizado.

Outro aspecto importante é que a metodologia proposta não é dependente da tecnologia utilizada para realizar o controle do processo, sendo comprovado pela implementação dos algoritmos gerados em duas tecnologias diferentes, o CLP e o FPGA. Com o CLP o processo foi controlado de forma adequada, não violando as especificações de funcionamento e de segurança. Para o FPGA, onde o objetivo é explorar as características paralelas, implementou-se o algoritmo descrito em VHDL realizando o controle do processo de Triagem de Caixas de forma satisfatória. Vale ressaltar que a utilização da tecnologia reconfigurável para o controle de processos de automação industrial é uma apontada como uma tendência de mercado, devido a sua capacidade de contornar as dificuldades encontradas pelos processadores sequências quando aplicados ao controle de processo com alto grau de paralelismo, justificando a inserção deste assunto neste trabalho.

Como recomendação para trabalhos futuros, sugere-se a construção de um ferramenta computacional para automatizar o processo de síntese do algoritmo de controle, onde através de uma representação da RdPI em um ambiente gráficos seja possível a geração do algoritmo, através da execução da etapas apresentadas neste trabalho. A

inclusão de elementos temporais a esta ferramenta computacional potencializaria a sua utilização para projetos de automação mais complexos e extensos, possibilitando a utilização da mesma para identificação do tempo de produção e, consequentemente, a sua otimização.

REFERÊNCIAS

AGUIRRE-SALAS, L.; SANTOYO-SANCHEZ, A. Sequence-detectability analysis of interpreted Petri Nets under partial state observations. in: IEEE CONFERENCE ON EMERGING TECHNOLOGIES FACTORY AUTOMATION, 14, 2009, Mallorca. **Proceedings of**, p. 1–7, Mallorca: IEEE, 2009.

ALTERA. **DE2 Development and Education Board**. Disponível em: <http://www.altera.com/education/univ/materials/boards/de2/unv-de2-board.html>: Acessado em: 26 jan. 2013.

BOBDA, C. **Introduction to Reconfigurable Computing**: architectures, algorithms and applications. London: Springer, 2007.

CARDOSO, J.; VALETTE, R. **Redes de Petri**. Florianópolis: Editora da UFSC, 1997.

CASSANDRAS, C. G.; LAFORTUNE, S. **Introduction to Discrete Event Systems**. 2^{nd} ed. New York: Springer, 2008.

DAVID, R. Grafcet: A powerful tool for specification of logic controllers. in: IEEE TRANSACTIONS ON CONTROL SYSTEMS TECHNOLOGY. **Proceedings of**, v. 3, n. 3, p. 253–268, sept. 1995.

DAVID, R.; ALLA, H. **Petri Nets and Grafcet**: tools for modelling discrete event systems. New York: Prentice Hall, 1992.

DAVID, R.; ALLA, H. **Discrete, continuous and hybrid Petri Nets**. 2^{nd} ed. Berlin: Springer, 2010.

DESEL, J.; REISIG, W. Place/Transition Petri Nets. **Lectures on Petri Nets I: Basic Models**, Springer, Berlin, v. 1491, p. 122–173, 1998.

ECONOMAKOS, C.; ECONOMAKOS, G. Optimized FPGA implementations of demanding PLC programs based on hardware high-level synthesis. in: IEEE INTERNATIONAL CONFERENCE ON EMERGING TECHNOLOGIES AND FACTORY AUTOMATION. **Proceedings of**, Hamburg, p. 1002–1009, sept. 2008.

FERNANDES, J.; ADAMSKI, M.; PROENCA, A. VHDL generation from hierarchical Petri Net specifications of parallel controllers. in: IEEE PROCEEDINGS COMPUTERS AND DIGITAL TECHNIQUES. **Proceedings of**, v. 144, n. 2, p. 127–137, mar. 1997.

GOMES, L.; BARROS, J.; COSTA, A.; NUNES, R. The Input-Output Place-Transition Petri Net class and associated tools. in: IEEE INTERNATIONAL CONFERENCE ON INDUSTRIAL INFORMATICS. **Proceedings of**, Vienna, v. 1, p. 509–514, june 2007.

GUSTIN, G. D. B. **Aplicação de Rede de Petri Interpretadas na Modelagem de Sistemas de Elevadores em Edifícios Inteligentes**. Dissertação (Mestrado) — Escola Politécnica da Universidade de São Paulo, 1999.

HAUCK, S.; DEHON, A. **Reconfigurable Computing**: The theory and practice of FPGA-Based computation. Burlington: Morgan Kaufmann, 2008.

ICHIKAWA, S.; AKINAKA, M.; IKEDA, R.; YAMAMOTO, H. Converting PLC instruction sequence into logic circuit: A preliminary study. in: IEEE INTERNATIONAL SYMPOSIUM ON INDUSTRIAL ELECTRONICS. **Proceedings of**, Montreal, v. 4, p. 2930–2935, july 2006.

INTERNATIONAL ELECTROTECHNICAL COMMISSION. IEC 61131-3, 2^{nd} Edition: Programmable Controllers - Programming Languages.

IORDACHE, M.; ANTSAKLIS, P. Petri Nets and programming: A survey. in: AMERICAN CONTROL CONFERENCE. **Proceedings of**, Missouri, v. 1, p. 4994–4999, jun. 2009.

MAGALHÃES, A. P. **Prática de Automação Industrial**. 1. ed. Cidade do Porto: Real Games Lda, 2009.

MAGALHÃES, A. P.; VIGÁRIO, B. T. Serios lecture from funny computer games. in: CONFERÊNCIA IBÉRICA DE SISTEMAS E TECNOLOGIAS DE INFORMAÇÃO. **Proceedings of**, v. 1, p. 497–502, 2009.

MURATA, T. Petri nets: Properties, analysis and applications. **Proceedings of The IEEE**, v. 77, p. 541–580, 1989.

OLIVEIRA, T. de. **Desenvolvimento de uma Arquitetura Multiprocessada e Reconfigurável para a Síntese de Redes de Petri em Hardware**. Tese (Doutorado em Engenharia Elétrica) — Universidade Estadual Paulista - UNESP, Ilha Solteira, 2008.

PAIS, R.; BARROS, S.; GOMES, L. A tool for tailored code generation from Petri Net models. in: IEEE CONFERENCE ON EMERGING TECHNOLOGIES AND FACTORY AUTOMATION. **Proceedings of**, Catania, v. 1, p. 856–864, sept. 2005.

PENG, S. S.; ZHOU, M. C. Ladder diagram and Petri-Net-based discrete-event control design methods. in: IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS, PART C: APPLICATIONS AND REVIEWS. **Proceedings of**, v. 34, n. 4, p. 523–531, nov. 2004.

PETERSON, J. L. **Petri Net Theory and the Modeling of Systems**. Upper Saddle River, NJ, USA: Prentice Hall PTR, 1981.

RAMIREZ-PRADO, G.; SANTOYO, A.; RAMIREZ-TREVINO, A.; BEGOVICH, O. Regulation problem in discrete event systems using interpreted Petri Nets. in: IEEE INTERNATIONAL CONFERENCE ON SYSTEMS, MAN, AND CYBERNETICS. **Proceedings of**, v. 3, p. 2174–2179, 2000.

RAMIREZ-TREVINO, A.; RIVERA-RANGEL, I.; LOPEZ-MELLADO, E. Observability of discrete event systems modeled by Interpreted Petri Nets. in: IEEE TRANSACTIONS ON ROBOTICS AND AUTOMATION. **Proceedings of**, v. 19, n. 4, p. 557–565, aug. 2003.

RIERA, B.; MARANGE, P.; GELLOT, F.; NOCENT, O.; MAGALHÃES, A. P.; VIGÁRIO, B. Complementary usage of real and virtual manufacturing systems for safe PLC training. in: IFAC SYMPOSIUM ON ADVANCES IN CONTROL EDUCATION. **Proceedings of**, v. 8, p. 89–94, 2009.

ROSE, J.; GAMAL, A. E.; SANGIOVANNI-VINCENTELLI, A. Architecture of field-programmable gate arrays. **Proceedings of the IEEE**, v. 81, n. 7, p. 1013–1029, jul 1993.

SANTOYO-SANCHEZ, A.; RUIZ-BELTRAN, E.; AGUIRRE-SALAS L.I.AND ORTIZ-MURO, V. Fault diagnosis of electrical systems using interpreted Petri Nets. in: IEEE INTERNATIONAL CONFERENCE ON EMERGING TECHNOLOGIES AND FACTORY AUTOMATION. **Proceedings of**, Hamburg, v. 1, p. 538–546, sept. 2008.

SILVA, C.; QUINTÁNS, C.; COLMENAR, A.; CASTRO, M. A.; MANDADO, E. A method based on Petri Nets and a matrix model to implement reconfigurable logic controllers. in: IEEE TRANSACTIONS ON INDUSTRIAL ELECTRONICS. **Proceedings of**, v. 57, n. 10, p. 3544–3556, oct. 2010.

SILVA, C.; QUINTÁNS, C.; MANDADO, E.; CASTRO, M. A. Methodology to implement logic controllers with both reconfigurable and programmable hardware. in: IEEE INTERNATIO-

NAL SYMPOSIUM ON INDUSTRIAL ELECTRONICS. **Proceedings of**, p. 324–328, june 2007.

SILVA, H. V. D.; VALLIM, M. B. R. Metodologia para conversão de Rede de Petri Interpretada para Grafcet visando projetos de automação industrial. in: IEEE/IAS INTERNACIONAL CONFERENCE ON INDUSTRY APPLICATIONS - INDUSCON. **Proceedings of**, v. 10, p. 1–8, nov. 2012.

VIGÁRIO, B. T.; MAGALHÃES, A. P.; FREITAS, F. T. Modern computer games technology in systems and control education. in: PORTUGUESE CONFERENCE ON AUTOMATIC CONTROL. **Proceedings of**, v. 1, p. 8, 2006.

WEGRZYN, M.; ADAMSKI, M. A.; MONTEIRO, J. L. The application of reconfigurable logic to controller design. **Control Engineering Practice**, v. 6, n. 7, p. 879–887, 1998.

ZHOU, M.; VENKATESH, K. **Modeling, Simulation, and Control of Flexible Manufacturing Systems**: A Petri Net approach. Danvers: World Scientific, 1999. (Series in Intelligent Control and Intelligent Automation, v. 6).