UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA MECÂNICA
DEPARTAMENTO DE PROJETO MECÂNICO

SIMGRAF
UM AMBIENTE COMPUTACIONAL PARA A SIMULAÇÃO E VALIDAÇÃO DE SISTEMAS AUTOMATIZADOS DE PRODUÇÃO UTILIZANDO O GRAFCET

N 16/94

Por: **Robson Figueira Dal'Bó**
Orientador: **Prof. Dr. João Maurício Rosário**

Dissertação apresentada à Faculdade de Engenharia Mecânica - FEM - UNICAMP como parte dos requisitos para a obtenção do título de **MESTRE** em **ENGENHARIA MECÂNICA.**

Campinas - 1994

ESTE EXEMPLAR CORRESPONDE A REDAÇÃO FINAL DA TESE DEFENDIDA POR Robson Figueira Dal'Bó E APROVADA PELA COMISSÃO JULGADORA EM 24/05/94.

João Maurício Rosário
ORIENTADOR

UNIVERSIDADE ESTADUAL DE CAMPINAS

FACULDADE DE ENGENHARIA MECÂNICA

TESE DE MESTRADO

TÍTULO DA TESE: SIMGRAF, UM AMBIENTE COMPUTACIONAL PARA A SIMULAÇÃO E VALIDAÇÃO DE SISTEMAS AUTOMATIZADOS DE PRODUÇÃO UTILIZANDO O GRAFCET

AUTOR: Robson Figueira Dal'Bó

ORIENTADOR: João Maurício Rosário

Aprovado por

João Maurício Rosário

Prof. Dr. João Maurício Rosário, Presidente

Prof. Dr. Geraldo Nonato Telles

Prof. Dr. Márcio Rillo

Campinas, 24 de maio de 1994

ESTE TRABALHO É DEDICADO A MINHA ESPOSA, ANA, QUE COM SEU AMOR E ETERNA COMPREENSÃO FORNECEU FORÇAS PARA QUE FOSSE FINALIZADO.

AGRADECIMENTOS

Ao Sr. Walter e Dona Débora, meus pais, pelo seus incentivos para que este trabalho não só fosse iniciado, mas fornecendo os alicerces necessários durante toda a vida.

Ao Professor João Maurício Rosário, pela orientação e paciência que souberam administrar os atrasos apesar da sua incansável dedicação à pesquisa.

Ao Professor João Plaza, pelo apoio e oportunidade de realização deste mestrado.

Aos parentes, amigos e conhecidos que, cada um ao seu modo, incentivaram-me a encontrar forças para vencer as barreiras.

Ao Dr. Hallim Féres Jr. e Dr. Gláuco Filellini, os médicos que cuidaram de minha vida e apoiaram na recuperação de meus conhecimentos.

A Deus, por estar sempre ao meu lado.

RESUMO

Nesta dissertação é elaborado e implementado um sistema para a simulação e validação da lógica utilizada para a automatização dos sistemas de produção.

O sistema é baseado no GRAFCET, sendo a interface de entrada e saída de dados, além do processamento feitos segundo esta metodologia.

O sistema foi elaborado seguindo a programação orientada à objeto, permitindo modularização e futuras ampliações.

Ao final do trabalho, serão realizados testes para a validação do SIMGRAF.

ABSTRACT

This dissertation presents a simulation and validation system for the logics will be used on production automation systems.

The simulation system is based on GRAFCET, all input and output data interface, as the processing of data are done by this methodology.

The system was created using the object oriented programming, that could modulate and amplify it in the future.

At the end of working, some tests are implemented to validate the SIMGRAF.

PROLOGO

Atualmente, o sistema de produção mundial passa por estágios de melhorias constantes a fim de enfrentar a concorrência, que torna-se maior com a internacionalização dos mercados consumidores.

Estágios de melhoria muitas vezes baseados em melhorias de qualidades e diminuição de custos.

Para assegurar a melhoria de qualidade, numerosos esforços têm sido feitos: trabalhos que permitam obter certificados de ISO 9000, grupos de trabalhos com técnicas de TQC (Total Quality Control), técnicas para resolução de problemas de qualidade, enfim, numerosas "brain storms" a fim de ter-se melhores qualidades.

Maiores têm sido os esforços de diminuição de custos, sem a diminuição de qualidade, ou melhor, com a melhoria da mesma.

É neste contexto que se situa este trabalho.

Com o interesse de aumento de vendas, existe o interesse das fábricas em produzirem mercadorias cada vez mais "customizadas", isto é, de acordo com as especificações que o cliente (customer) necessita, não deixando em segundo plano os custos.

A "customerização" gera a necessidade das plantas estarem aptas a se adaptarem à variação de produtos fabricados. Esta adaptação deve existir desde a entrada de novas matérias-primas nos estoques até mesmo a expedição dos produtos já fabricados, passando pela produção.

Como ferramentas indispensáveis para a customerização tem-se a interligação entre base de dados e informações da planta, a qual deve ser completa e eficiente, bem como as máquinas possuirem um alto nível de adaptabilidade aos produtos desejados pelo cliente.

A adaptabilidade das máquinas implica diretamente em sistemas versáteis.

Por outro lado, a diminuição de custos também exige que as máquinas fiquem o mínimo tempo paradas, sem produzirem. Seja o tempo parado causado pela troca de processos, pela implantação de novas programações das máquinas, por erros de programação ou operação errada das máquinas.

Erros na programação ou operação errada estão relacionados com a confiabilidade dos sitemas implantados.

Sistemas mais confiáveis são aqueles que têm menor probabilidade de ocorrerem falhas ou paradas em um período de tempo determinado.

Para ter-se sistemas bastante confiáveis, uma solução está no intertravamento entre operações, diretamente ligado ao aumento de complexidade do sistema. Complexidade que consome muitas horas de engenharia, podendo ser ajudada por sistemas de simulação.

Sistemas de simulação permitem não só analisar o comportamento dos intertravamentos, mas também encontrar erros na lógica. Seja quais forem, os problemas devem ser encontrados e solucionados antes de serem implantados nas máquinas.

Para haver maior versatilidade de programação, existe a necessidade dos programas estarem bem documentados, possibilitando não só ao conceptor do programa a alteração rápida e segura, mas a documentação deve deixar aptas as pessoas que vierem substituir o conceptor.

Todo este processo consome tempo e, em conseqüência, eleva custos.

Tendo como objetivo ajudar a solucionar este problema de versatilidade aliada à confiabilidade, surgiu a idealização deste trabalho.

Com este objetivo, a dissertação foi subdividida nos seguintes capítulos:

Capítulo 1, onde são apresentandas as partes, operativa e comando, que compõem uma máquina ou equipamento de produção, bem como as etapas necessárias para automação dos mesmos;

Capítulo 2, onde é descrito o GRAFCET, padrão para representação da parte lógica de operação das máquinas;

Capítulo 3, onde são explanados o simulador de GRAFCET (SIMGRAF) construído, as classes, subclasses e métodos utilizadas para a sua construção em Linguagem Orientada a Objeto (LOO);

Capítulo 4, onde são idealizadas máquinas para as quais será feita a descrição da parte lógica, utilizando o GRAFCET, e seguindo-se a conversão do GRAFCET para a sintaxe do SIMGRAF e executadas as simulações;

Capítulo 5, onde são apresentadas as conclusões e considerações finais da dissertação;

ÍNDICE

LISTA DE FIGURAS

Lista de Tabelas

Capítulo 1

Concepção da Automação

1.1. Introdução

Em fase anterior à exposição do GRAFCET, de sua utilização na automação industrial e no simulador do GRAFCET (SIMGRAF) faz-se necessário a descrição das partes em que podem ser divididos os equipamentos de produção.

Também serão expostas as fases de um projeto de automação industrial em equipamentos novos ou já existentes.

1.2. Elementos dos Equipamentos de Produção

Os equipamentos de produção, sejam simples máquinas ou células integradas de produção ou mesmo linhas contínuas ou de processamento batch, podem ser divididos em duas partes, conforme mostrado na fig 1.1:

1a. parte de operação;
2a. parte de comando.

A parte operativa é responsável pela modificação dos objetos que serão por ela manuseados. Por exemplo, em uma termo-embaladora, pode-se considerar que a parte operativa é composta por esteiras, cortadoras, aquecedores, etc.

A parte de comando, como o próprio nome já especifica, é a responsável pela coordenação dos ações da parte operativa. No exemplo, pode-se considerar parte de comando os sensores de posição, temperatura, controlador lógico (CPU), displays e teclados (IHM), contatores, distribuidores (pré-acionadores) e outros.

Aprofundando-se ainda mais pode-se dividir a parte de comando em outras duas partes:

equipamentos;

lógica.

A parte de equipamentos é composta pelos meios físicos capazes de fornecer informações, processá-las segundo um algoritmo e em seguida atuar sobre o sistema a fim de controlá-lo. Nesta parte são inclusos os pré-acionadores, sensores, IHM, as unidades de comunicação entre sistemas e a unidade de processamento das informações.

O algoritmo segundo o qual as informações são processadas é a chamada parte lógica.

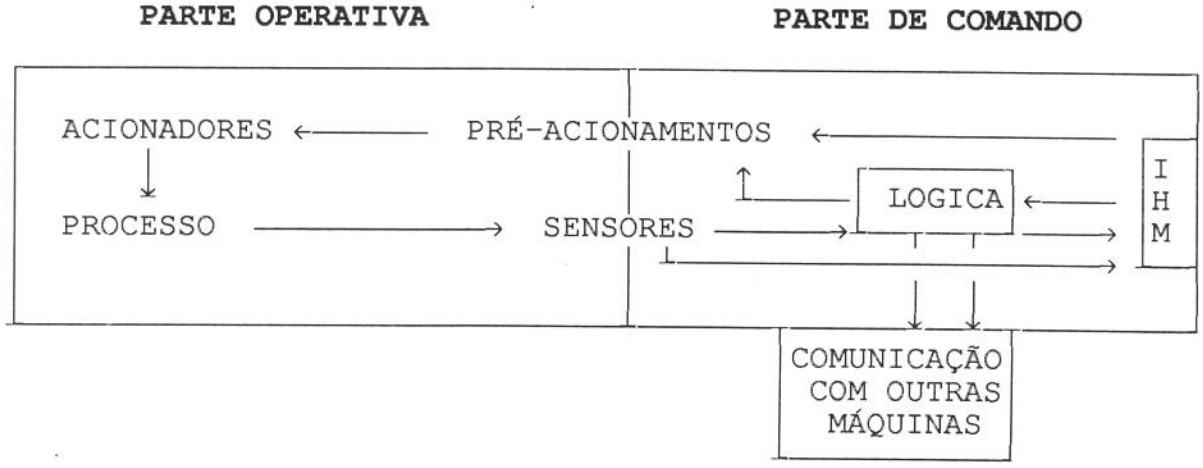

FIG 1.1 - Elementos dos equipamentos de produção

Concentrando-se ainda mais o objetivo, neste trabalho será enfocado a lógica dos sistemas.

A partir deste ponto será descrito o processo de criação de novos sistemas automáticos. Cabe ressaltar que, para o processo de automatização de sistemas já existentes, algumas tarefas não serão necessárias.

1.3. Automação dos Equipamentos de Produção

A criação da parte de comando de um sistema pode ser dividido em quatro fases. São elas:

1o. Concepção: estabelecimento detalhado das necessidades do funcionamento do sistema. Sendo elaborado um caderno de tarefas onde o detalhamento de necessidades será descrito;

2o. Realização: aprimoramento do caderno de tarefas, transformando o seu texto em esquemas lógicos que deverão existir no sistema;

3o. Preparação: estabelecimento dos meios físicos para que as especificações do caderno de tarefas possam ser implementadas, sejam elas hardware e/ou software. Quando trata-se de automatização de sistemas já existentes, esta fase pode, em muitos pontos, ser efetuada através de adaptação a equipamentos já existentes.

4o. Exploração: parte final da elaboração da parte de comando, onde a lógica será implementada e em seguida validada.

As três primeiras fases, concepção, realização e preparação, podem ser feitas de maneira razoavelmente rápida. Devendo ser bastante clara e muito bem documentada, a fim de que a manutenção e alteração da lógica seja possível, principalmente em sistema sofisticados.

É na fase de exploração que se encontram os trabalhos mais árduos de engenharia, uma vez que a lógica deverá ser traduzida em entradas e saídas, devendo ser testadas as combinações das mesmas e identificadas as redundâncias e repetições da lógica. Ao final deste trabalho o sistema pode ser considerado validado.

Esta árdua tarefa consome muito tempo e esforços nem sempre eliminando situações que podem ser danosas aos clientes (máquinas).

Para ajudar nesta função concebeu-se a criação do simulador SIMGRAF (Simulador de GRAFCET), onde a lógica pode ser testada e, no futuro, eliminação de redundâncias lógicas e melhorias na tradução do caderno de tarefas do equipamento para parte lógica poderão ser realizadas. Viabilizando programações menores, mais confiáveis e versáteis atendendo ao objetivo maior de produzir corretamente com menores custos.

Neste capítulo foram descritas as duas partes nas quais qualquer equipamento de produção pode ser dividido, a parte de comando e a parte operativa. A parte de comando é a responsável pelo gerenciamento das tarefas do equipamento, a lógica do equipamento.

Esta lógica, quando em fase de desenvolvimento de uma automação necessita ser testada, documentada e validada. Para este fim o GRAFCET pode fornecer o auxílio na descrição da lógica, como será descrito no capítulo seguinte.

Capítulo 2

O GRAFCET

2.1. Introdução

Como já foi descrito no capítulo 1, a parte lógica dos sistemas automatizados necessita ser descrita e documentada. Para isso existem algumas formas já conhecidas, por exemplo, representação por contatos (normalmente utilizadas em CLP) e a representação por Rede de Petri (apresentada no Anexo A).

Apresentar-se-á uma forma alternativa que é o GRAFCET.

O GRAFCET tem a função de exprimir graficamente a lógica dos sistemas.

Sugere-se que, caso seja desejado um maior aprofundamento no GRAFCET, sejam consultados [01], [02], [03], [10], [11] e [20].

2.2. GRAFCET

Para explicitar um sistema automatizado em uma forma gráfica, que permita a qualquer analisador compreender o funcionamento do sistema, é necessário que o "desenhista" crie uma tabela de códigos a fim de tornar seu "desenho" ou esquema compreensível às pessoas que forem utilizá-los.

Caso esse "desenhista" não só deseje explicitar, mas além disso deseje modificar a forma do automatismo, ou seja, manipular a lógica do sistema, a tabela de códigos deverá ser ainda mais complexa.

O "desenhista" poderia utilizar graficamente as redes de Petri e/ou outras soluções.

Tendo em vista a complexidade das Redes de Petri, possuindo uma possibilidade de aplicações muito superior às que estão restritos os comportamentos cíclicos das máquinas e sistemas automáticos. Aliando-se às possibilidades de conflitos (estruturais ou efetivos) que possam ocorrer quando essas ferrramentas são utilizadas para exprimir-se um sistema automático, criou-se o GRAFCET (Gráfico de Comando Etapa-Transição).

O GRAFCET é considerado uma particularização das Redes de Petri.

O GRAFCET foi criado pela AFCET (Association Française pour la Cybernètique, Economique et Technique) e posteriormente padronizado pela ADEPA (Agencie Nationale pour la Developpement de la Production Automatise).

O GRAFCET é baseado na descrição do sistema a ser automatizado (sistema) em um caderno de tarefas. O caderno de tarefas transmite as necessidades de um usuário (cliente) para o conceptor (fornecedor), que inicialmente chamávamos de "desenhista".

2.2.1. Caderno de Tarefas

O caderno de tarefas deve relatar ao conceptor as funções a serem desempenhadas pelo sistema, de forma clara, completa, precisa, não ambígua e não omissa.

Para garantir que o caderno de tarefas desempenhe corretamente suas funções, isto é, relatar ao conceptor o sistema, torna-se necessário que essa descrição seja realizada em dois níveis:

1o. Nível: São as especificações funcionais. Permitirão ao fornecedor compreender as ações do sistema, descrevendo passo-a-passo o comportamento da parte de comando. Não são definidos as maneiras que tais tarefas serão realizadas, isto é, os tipos de acionamentos, sensores, atuadores etc.

2o. Nível: São agrupadas as especificações técnicas e operacionais que caracterizaram as formas e restrições do sistema automatizado e o processo. São elas:

especificações tecnológicas: completam as especificações funcionais caracterizando as formas de ações entre o sistema e o processo, tais como, acionadores, sensores, interface homem-máquina e outros.

especificações operacionais: caracterizam o funcionamento do equipamento durante sua vida útil, por exemplo situações de emergência, possibilidades de manutenção, disponibilidade, confiabilidade, possibilidade de alterações do sistema, etc.

O fornecedor, a partir desses dois níveis, transforma a descrição do caderno de tarefas em uma descrição mais profunda do equipamento (dossiê técnico).

O dossiê técnico é a forma escolhida pelo fornecedor para representação do caderno de tarefas, por exemplo, diagrama de contatos, programa, etc.

Chegado a este nível o sistema poderá ser traduzido para o GRAFCET.

2.3. Definições

O GRAFCET se baseia em um grupo de definições fundamentais sobre as quais são estabelecidas as regras fundamentais.

Considera-se essencial a exposição destas definições e regras antes de ser apresentado o trabalho desta dissertação sobre o Simulador de Grafcet (SIMGRAF).

A seguir será iniciada a abordagem das definições fundamentais as quais, como todo o GRAFCET, será baseada na álgebra booleana de verdadeiro e falso.

O sistema objeto, isto é, o sistema ao qual deseja-se associar um GRAFCET pode ser dividido em:

1o.etapas;
2o.transições;
3o.ligações orientadas.

2.3.1. Etapas

Blocos aos quais estão associadas uma ou mais ações do objeto, ou seja, atuações sobre as variáveis de entrada ou saída.

No GRAFCET as ações podem ser representadas de maneira literal, descrevendo-se as atuações que ocorrerão sobre o sistema, ou de forma simbólica, onde a correlação entre símbolos e os meios físicos foi pré-definida.

Uma etapa pode estar ativa ou inativa em um determinado instante.

Ao conjunto de etapa ativas em um determinado momento dá-se o nome de situação do objeto em estudo ou situação do sistema.

2.3.2. Transições

Funções lógicas que coordenam a evolução entre as etapas de origem e de destino ligadas à transição.

A transição pode ser ou não válida em um instante.

É associada uma receptividade à cada transição, ou seja , as condições lógicas que são avaliadas possibilitando ou não a validação da receptividade. Quando a transição é valida, possibilita a ativação das etapas destino e a desativação das etapas de origem.

Da mesma forma que a etapa, a transição pode ser representada de forma literal ou simbólica.

A receptividade pode ser influenciada pelo tempo, isto é, um temporizador será inicializado pela ativação da etapa especificada, desde que a etapa anterior à esta receptividade esteja validada.

2.3.3. Ligações Orientadas

As LIGAÇÕES ORIENTADAS indicam o caminho de evolução entre uma etapa e uma transição ou entre uma transição e uma etapa. São representadas por linhas que possuem sentido único de ligação.

2.4. Exemplo de GRAFCET

Por exemplo considera-se o objeto em estudo uma carro que se desloca em um monotrilho de uma extremidade a outra. Inicialmente o carro está na extremidade esquerda A, quando acionado o botão de partida S o carro se deslocará em direção à extremidade B. Ao chegar à B então retorna a A, onde para, esperando pelo reinício do próximo ciclo.

Associando-se o objeto à um GRAFCET temos as seguintes etapas:

ETAPA 1: carro parado em A
ETAPA 2: carro se deslocando de A para B
ETAPA 3: carro se deslocando de B para A

Como transições tem-se:
TRANSIÇÃO 1: acinamento do botão de partida S
TRANSIÇÃO 2: o carrinho está na extremidade B
TRANSIÇÃO 3: o carrinho está na extremidade A

Como ligações orientadas tem-se:
LIGAÇÃO 1: da ETAPA 1 à TRANSIÇÃO 1
LIGAÇÃO 2: da TRANSIÇÃO 1 à ETAPA 2
LIGAÇÃO 3: da ETAPA 2 à TRANSIÇÃO 2
LIGAÇÃO 4: da TRANSIÇÃO 2 à ETAPA 3
LIGAÇÃO 5: da ETAPA 3 à TRANSIÇÃO 3
LIGAÇÃO 6: da TRANSIÇÃO 3 à ETAPA 1

Graficamente tem-se o seguinte GRAFCET:

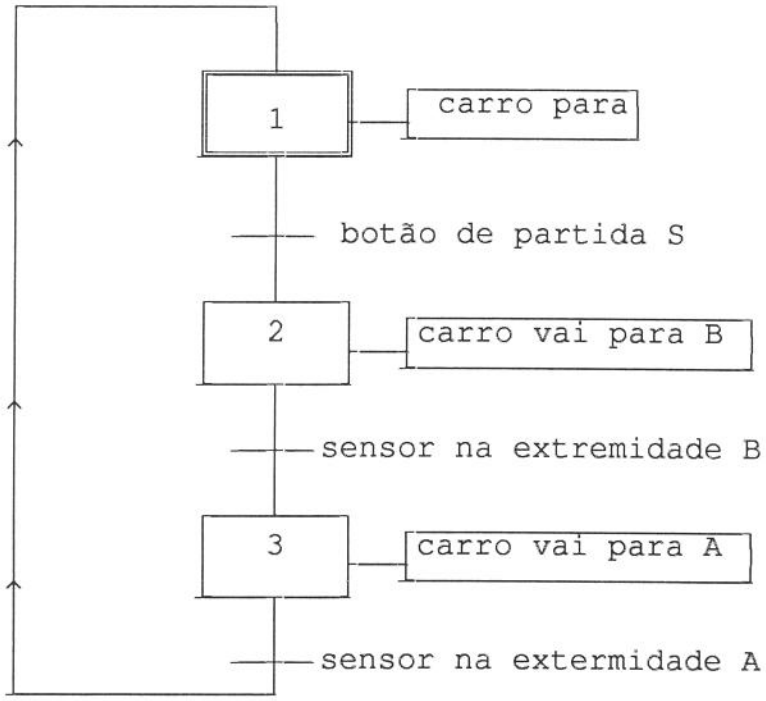

fig 2.1 - Exemplo de GRAFCET

2.5. Regras Imperativas e de Evolução

No exemplo acima observa-se o respeito à regra imperativa do GRAFCET que é a alternância entre etapa e transição com transição e etapa.

Cabe ser ressaltado que nenhuma ação é realizada ou nenhuma receptividade é avaliada em um espaço de tempo nulo, isto é, não são instantâneas.

Este ressalto é significativo devido a existência de etapas sem nenhuma ação associada, bem como da possibilidade de transições sem receptividade associada. Características importantes que possibilitam a melhor representação do objeto por parte do conceptor.

O comportamento do GRAFCET é baseado em cinco regras de evolução, que são:

1o. A inicialização refere-se as etapas que estão ativas no início do funcionamento do GRAFCET;

2o. O disparo de uma transição só acontece quando todas as suas etapas de entrada estão ativas e sua receptividade é verdadeira.

3o. A validação de uma transição ocasiona a ativação das etapas de saída e a desativação das etapas de entrada;

4o. Transições simultaneamente validáveis são simultaneamente validadas, ou seja, em uma determinada situação do objeto em estudo todas as transições que são validáveis se tornam válidas em um mesmo instante de tempo;

5o. Se uma etapa é desativada e ativada ao mesmo instante então permanecerá ativada.

2.6. Particularidades

Uma vez estabelecidas as definições fundamentais assim como as regras de evolução do GRAFCET podem ser expostos algumas particularidades de receptividades, ações e arcos de ligação. Particularidades que tornam possíveis a melhor descrição do sistema.

2.6.1. Receptividades

Como particularidades de receptividades tem-se:

AND As receptividades podem ser composta de mais de uma variável ligadas pela função lógica AND. As variáveis podem ser uma determinada etapa ou entrada ou saída;

OR As receptividades podem ser composta de mais de uma variável ligadas pela função lógica OR. As variáveis podem ser uma determinada etapa ou transição ou entrada ou saída;

NOT As receptividades podem ser composta de mais de uma variável ligadas pela função lógica NOT. As variáveis podem ser uma determinada etapa ou transição ou entrada ou saída;

TEMPORIZADOR Uma receptividade pode tornar-se verdadeira após um determinado tempo de ativação de uma determinada etapa;

MUNDACA DE ESTADO Uma receptividade pode avaliar o estado de uma variável ou a mundaça deste estado. As variáveis podem ser uma determinada etapa ou transição ou entrada ou saída;

2.6.2. Ações

Como particularidades das ações tem-se:

AÇÕES CONDICIONAIS A execução de uma ação pode ser submissa a uma condição lógica entre variável de entrada ou de outra etapa, isto é, mesmo o estado de uma etapa sendo ativo podem haver ações condicionais que não sejam executadas pois sua condição é momentaneamente falsa.

AÇÕES IMPULSIONAIS A ativação da ação ocorrerá durante um tempo determinado quando a etapa tornar-se ativa.

AÇÕES CONTÍNUAS A ação permanece sendo executada enquanto a etapa estiver ativa;

2.6.3. Ligações Orientadas

Como particularidades de ligações orientadas tem-se:

DISTRIBUIÇÃO AND Uma transição possui seu arco de ligação de saída conectado a duas ou mais etapas que tornar-se-ão ativas ao mesmo instante. Conforme exemplo a seguir:

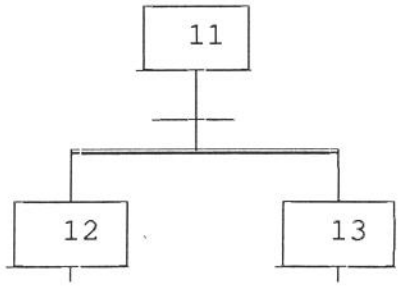

fig 2.2 - Distribuição AND

JUNÇÃO AND Uma transição possui seu arco de ligação de entrada conectado a duas ou mais etapas que deverão estarem ativas em um mesmo instante para que a receptividade seja avaliada. Conforme exemplo a seguir:

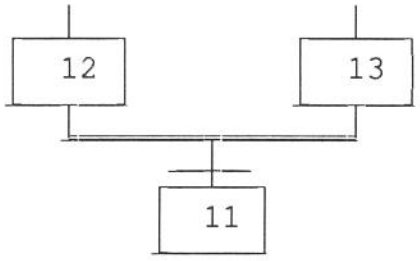

fig 2.3 - Junção AND

DISTRIBUIÇÃO OR Uma etapa pode estar conectado a duas ou mais transições que serão testadas em um mesmo instante podendo ou não serem ativadas, de acordo com a avaliação de suas receptividades. Conforme exemplo a seguir:

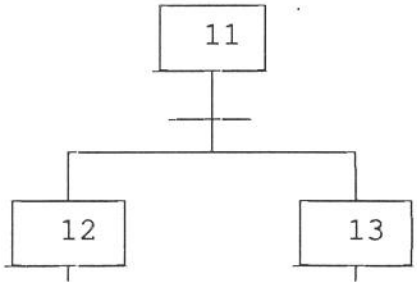

fig 2.4. - Distribuição OR

JUNÇÃO OR Uma ou mais transições estão com seus arcos de ligação de saída conectados à mesma etapa, a qual se tornará ativa assim que a primeira transição tornar-se verdadeira. Conforme exemplo a seguir:

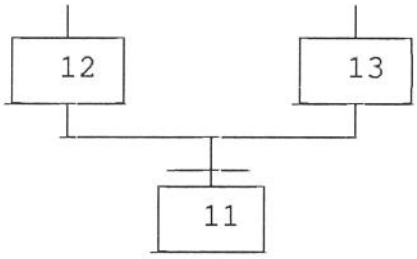

fig 2.5 - Junção OR

O GRAFCET permite que um caderno de tarefas seja traduzido em sub-programas desde que seja mantida a condição de não haver simultaneidade de chamadas a um mesmo sub-programa.

Neste capítulo observou-se a capacidade do GRAFCET em representar a parte lógica dos equipamentos. Foram descritas as regras de evolução e a regra imperativa do GRAFCET, bem como as particularidades das ações e das receptividades. No capítulo seguinte será descrito a aplicação do GRAFCET para a realização do SIMGRAF, ressaltando a necessidade da utilização da linguagem orientada a objeto (LOO) para implementação do SIMGRAF.

CAPÍTULO 3

O SIMULADOR DE GRAFCET

3.1. Introdução

O SIMGRAF pode ser dividido em duas partes:

.descrição do objeto a ser simulado;

.simulação do objeto.

A parte de descrição do objeto terá a função de formular o GRAFCET de acordo com o padrão estabelecido. Este padrão pode ser dividido em duas etapas:

.descrever os arcos de ligação entre etapas-transições e entre transições-etapas;

.descrever as funções lógicas das receptividades e as ações das etapas.

A parte de simulação do objeto terá a função de analisar a validade ou não das receptividades e, conseqüentemente, a ativação ou não das etapas, ou seja, a execução das ações a si relacionadas.

Neste capítulo será exposto como o sistema de Simulação de GRAFCET (SIMGRAF) foi elaborado. Serão abordados:

sintaxe utilizada para a descrição do objeto em simulação;

descrição do macro-fluxograma;

operação do SIMGRAF;

descrição do SIMGRAF através da LOO.

3.2. Sintaxe de Descrição do Objeto

Os sistemas industriais descritos segundo a metodologia GRAFCET, isto é, divididos em etapas, transições, ações e receptividades devem ser codificados segundo a sintaxe elaborada para a utilização do SIMGRAF.

As ligações existentes entre as M etapas e N transições serão representadas por uma matriz M1 de M linhas por N colunas. Onde o elemento $m1_{ij}$ é unitário se existir ligação entre a etapa i e a transição j. O elemento $m1_{ij}$ é nulo caso contrário.

As ligações existentes entre as N transições e M etapas serão representadas por uma matriz M2 de N linhas por M colunas. Onde o elemento $m2_{ji}$ é unitário se existir ligação entre a transição j e a etapa i. O elemento $m2_{ji}$ é nulo caso contrário.

As ações serão representadas através da matriz M3, que será tridimensional.

A ação de uma etapa será decomposta em sub-ações.

A sub-ação corresponde a setar ou resetar uma entrada ou saída.

A sub-ação pode estar condicionada ao estado de uma entrada ou saída ou etapa ser verdadeira ou falsa.

O número de colunas da matriz M3 será o número de etapas M, o número de linhas será 4 e a profundidade (eixo z) será o número máximo de sub-ações por etapa.

As receptividades serão representadas através da matriz M4, que será tridimensional.

As receptividades serão decompostas em subgrupos lógicos (AND, OR ou NOT). Os subgrupos de uma receptividade serão relacionados através das funções AND ou OR.

Os subgrupos podem ser "temporizadores" que disparam a contagem com a ativação de uma determinada etapa. O subgrupo tornar-se-á verdadeiro quando o número de teste realizados pelo SIMGRAF ultrapassar o valor pré-determinado.

A receptividade também pode ser fictícea, isto é, tornar-se-á verdadeira quando a sua etapa anterior tornar-se ativa.

O número de colunas da matriz M4 será o número de transições N, o número de linhas será o número máximo de variáveis em um mesmo subgrupo acrescidos de 4 (linhas utilizadas pelo sistema) e a profundidade (eixo z) será o número máximo de subgrupos por receptividade.

O estado das entradas (INPUT's) é armazenado no vetor V1, unidimensional, com comprimento igual ao número de entradas.

O estado das saídas (OUTPUT's) é armazenado no vetor V2, unidimensional, com comprimento igual ao número de saídas.

O estado das etapas é armazenado no vetor V3, unidimensional, com comprimento M.

O estado das transições é armazenado no vetor V4, unidimensional, com comprimento N.

O SIMGRAF possui um contador de teste que gerencia as temporizações e o número total de testes realizados na simulação.

O SIMGRAF permite que o objeto seja a ele descrito via teclado ou via arquivo de dados. Caso se deseje utilizar o arquivo de dados este deverá obedecer a sintaxe pré-estabelecida, a qual será descrita posteriormente.

No SIMGRAF existe a possibilidade de entradas e saídas serem fictíceas, via teclado ou via portas paralelas, neste caso serão permitidas até 16 entradas e 16 saídas.

3.3. Programação Orientada à Objeto (POO)

3.3.1. Conceito

Técnica que permite codificar o programa de forma a tornar transparente (invisível) ao usuário a forma que os dados são manipulados para obter-se a ação desejada. A esta forma de transparência dá-se o nome de encapsulamento.

A linguagem orientada a objeto é formada por classe que, através de mensagens (atributos) solicitam que um dos diferentes métodos (ações) que a compõem seja executado sobre o objeto.

Os métodos deverão ser executados sobre qualquer tipo de objeto solicitante, isto é chamado de polimorfismo.

Tendo-se diferentes objetos que necessitam de classes para manipulá-los de forma pouco diferentes, existe a possibilidade de herdar os métodos de uma classe superior. Desta forma é desnecessária a redefinição de toda a classe, não gerando códigos redundantes.

Como exemplo de programação orientada a objeto pode-se vizualizar as atividades para acionar um motor trifásico de alta potência, que será chamado de MOTOR 1.

Como objeto tem-se o MOTOR 1, o seu acionamento é transparente ao usuário que simplesmente aperta a botoeira para ligá-lo no painel de comando.

Quando a botoeira é pressionada é solicitada á classe acionamento, através da mensagem "partir em estrela" envia aos contatores do motor ações para a partida em modo estrela.

Neste instante é também enviada a mensagem de "inicializar temporizador de conversão" para que o método de contagem de X segundos antes de enviar nova mensagem de "partir em triângulo" a outro método desta classe.

O método de "partir em triângulo" fará a troca da ligação do motor de estrela para triângulo.

Caso houvesse o acionamento de um pequeno MOTOR 2 conectado à mesma botoeira, este necessitaria apenas da ligação em triângulo. A mesma classe seria utilizada com a diferença nas chamadas de métodos, isto é o polimorfismo.

Uma outra solução seria haver uma classe , com o acionamento de motores com ligações tipo triângulo, e uma subclasse que, além de herdar esse tipo de acionamento da classe, possuiria o método de temporizador e o de ligação em estrela.

Através da LOO é possível a criação de uma biblioteca de rotinas permitindo maior flexibilidade, confiabilidade, localização de erros e produtividade na elaboração de softwares. Estas facilidades são contrabalanceadas pela complexidade na programação/implementação, que exige melhores aplicativos para manipular o grande número de bibliotecas que são criadas, além do maior tempo para a análise e execução dos sistemas.

Maiores detalhes podem ser obtidos através de [05], [07], [08] e [06].

3.3.2. Aplicação ao SIMGRAF

O SIMGRAF foi baseado em POO pois:

1o. O GRAFCET é baseado em estruturas bastante modulares (objetos - etapas/ações, transições/receptividades e arcos de ligação);

2o. Os objetos necessitam ser trabalhados externamente, sendo comandados pelo usuário, e internamente, verificando a lógica do GRAFCET;

3o. Facilidade de programação e futura ampliação do trabalho.

3.4. Descrição do Macro-fluxograma do SIMGRAF

O macro-fluxograma do SIMGRAF é apresentado na figura abaixo.

Nesta figura pode-se notar a semelhança das funções de cada uma das células, sendo mais um fator que indicará o uso da linguagem orientada a objeto (LOO) para a implementação do SIMGRAF.

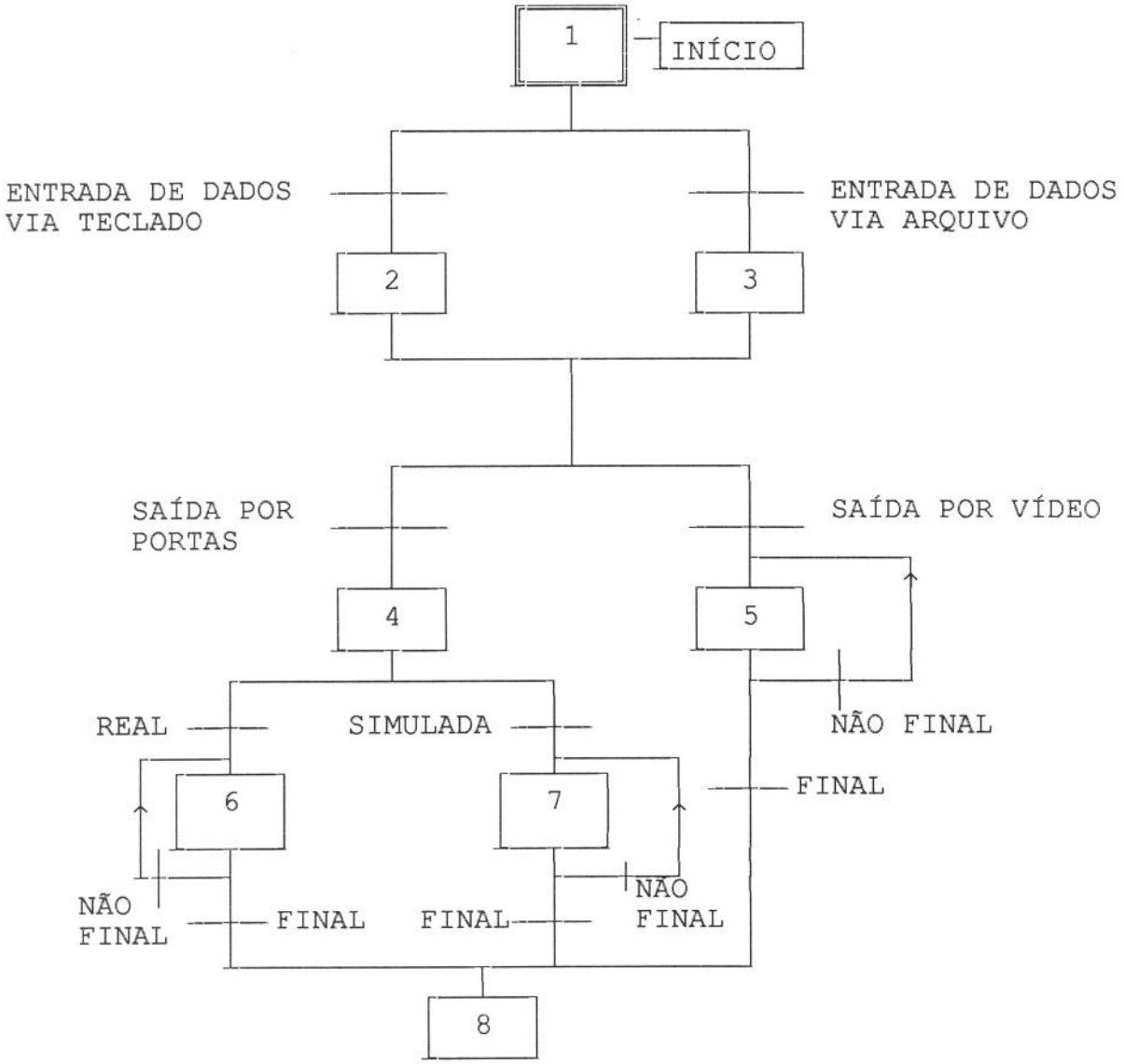

fig 3.1 - Macro-fluxograma do SIMGRAF

3.5. Operação do SIMGRAF

A seguir será descrito o modo de operação do SIMGRAF. Serão descritas, em blocos, as fases de operação.

3.5.1. Modo de Operação

O SIMGRAF foi idealizado para ser operado em um sistema composto por um microcomputador acoplado a um conjunto de entradas e saídas paralelas, conforme pode ser observado na figura.

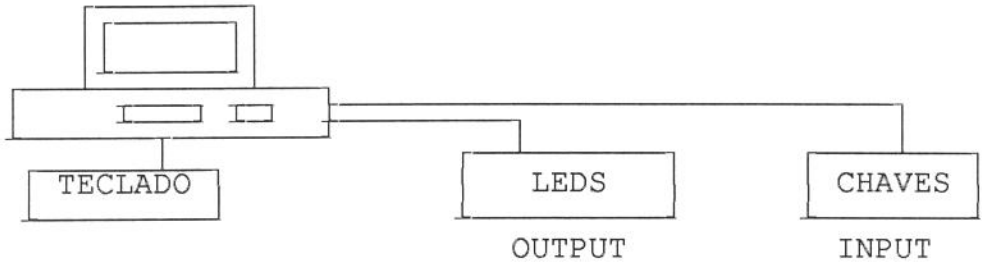

fig 3.2 - Base Operacional do SIMGRAF

Inicialmente é solicitado ao usuário o modo que serão realizadas as entradas e saídas de dados.

Primeiro o SIMGRAF questiona se serão utilizadas portas de I/O ou se as mesmas serão efetuadas via teclado.

Caso seja feita a opção por porta de I/O, o SIMGRAF questionará se as portas são reais ou serão simuladas, isto é, se será mostrado em vídeo o valor da palavra de dois bytes que seria enviado a porta paralela de saída.

Caso seja desejado a utilização de portas reais o SIMGRAF questionará o endereço inicial da porta, cujo default é 228H.

3.5.2. Descrição do Objeto

Após especificado qual o modo de operação do SIMGRAF este solicitará a forma em que o objeto lhe será descrito, via arquivo ou via teclado.

3.5.2.1. Via Arquivo

Caso a opção seja via arquivo, inicialmente será solicitado o nome do arquivo que contém o dimensionamento do objeto a ser simulado e em seguida será solicitado o nome do arquivo que descreve o obje-

to através dos vetores V1 (entradas ativas), V2 (saídas ativas), V3 (etapas ativas) e as matrizes M1 (ligações etapas-transições), M2 (ligações transições-etapas), M3 (ações das etapas) e M4 (receptividades das transições).

Estes arquivos deverão ter extensão .GFT.

Todos os dados necessários para a descrição do objeto poderão ser armazenados no mesmo arquivo, porém serão requisitados pelo SIMGRAF em duas fases separadas.

3.5.2.2 Via Teclado

Caso a opção de descrição do objeto seja via teclado, será inicialmente solicitado a descrição do dimensionamento do objeto.

Serão solicitados os seguintes dimensionamentos:

.número de etapas;
.número de transições;
.número de entradas (inputs);
.número de saídas (outputs);
.número máximo de variáveis por subgrupo de receptividade;
.número máximo de subgrupos por receptividade;
.número máximo de dezenas para tempo de simulação;
.número máximo de sub-ações por etapa.

Após o dimensionamento do objeto serão solicitados a descrição de:

.arcos de ligação entre etapas e transições;
.arcos de ligação entre transições e etapas
.entradas (inputs) inicialmente ativas;
.saídas (outputs) inicialmente ativas;
.ações das etapas;
.receptividades das transições;
.etapas inicialmente ativas;

A seguir considera-se necessário o detalhamento de cada fase de descrição do objeto.

3.5.3. Arcos de Ligação entre Etapas e Transições

O SIMGRAF questiona qual o número da etapa de origem do arco de ligação que deseja-se descrever e o número da transição de destino do arco.

Este questionamento é contínuo até que todas as ligações entre etapas e transições sejam descritas ao SIMGRAF, o que o operador sinalizará teclando <F> após o último arco.

3.5.4. Arcos de Ligação entre Transições e Etapas

O SIMGRAF questiona qual o número da transição de origem do arco de ligação que deseja-se descrever e o número da etapa de destino do arco.

Este questionamento é contínuo até que todas as ligações entre transições e etapas sejam descritas ao SIMGRAF, o que o operador sinalizará teclando <F> após o último arco.

3.5.5. Entradas Inicialmente Ativas

O SIMGRAF questiona quais são as entradas (inputs) que estarão ativas no início da simulação.

Este questionamento é contínuo até que todas as entradas inicialmente ativas tenham sido descritas ao SIMGRAF, o que o operador sinalizará teclando <F> após a última entrada.

3.5.6. Saídas Inicialmente Ativas

O SIMGRAF questiona quais são as saídas (outputs) que estarão ativas no início da simulação.

Este questionamento é contínuo até que todas as saídas inicialmente ativas tenham sido descritas ao SIMGRAF, o que o operador sinalizará teclando <F> após a última saída.

3.5.7. Ações das Etapas

O SIMGRAF solicitará a descrição das ações que deverão ser realizadas quando a etapa tornar-se ativa.

Serão solicitadas ao operador as seguintes perguntas:

Qual o número da etapa cujas ações serão descritas?

A ação se atuará sobre uma entrada ou sobre uma saída?

Qual o número do borne em que será realizada a ação?

A ação deverá setar ou resetar o borne escolhido?

É uma ação condicional?

Se for ação condicional, é condicional a uma etapa, saída, etapa ou transição?

Se for ação condicional, qual o número do borne que deverá ser testado para verificar a veracidade da questão?

Se for ação condicional, o borne em questão deverá estar ativo ou inativo para tornar a ação condicional verdadeira?

Existe mais ações a serem realizadas por esta etapa?

Para finalizar a descrição de ações de etapas o operador deverá apertar <F>. Para descrever ações de outra etapa deverá apertar <ENTER>.

3.5.8. Receptividade das Transições

O SIMGRAF solicitará a descrição das receptividades que deverão ser testadas nas transições.

Uma receptividade é dividida em subgrupos relacionados pelas funções lógicas AND ou OR. Internamente cada subgrupo pode conter múltiplas variáveis relacionadas pela mesma função lógica AND, OR. Existe a possibilidade do subgrupo possuir a função NOT, mas em apenas uma variável. As variáveis dos subgrupos podem ser entradas, saídas ou etapas.

Por exemplo, a receptividade de número 5 do objeto em questão será verdadeira quando a saída de número 2 E a entrada de número 4 forem ambas verdadeiras OU quando a etapa de número 3 E a etapa 7 forem verdadeiras OU a entrada de número 1 for falsa.

A receptividade do exemplo $((Sa_2.En_4)+(Et_3.Et_7)+\sim(En_1))$ deverá ser descrita ao SIMGRAF como sendo composta por três subgrupos.

O primeiro subgrupo é feito por uma lógica AND de duas variáveis, a saída de número 2 e a entrada de número 4.

Este subgrupo se relaciona com um segundo subgrupo através de uma lógica OR.

O segundo subgrupo é feito por uma lógica AND de duas variáveis, a etapa 3 e a etapa 7.

Este subgrupo se relaciona com um terceiro subgrupo através de uma lógica OR.

O terceiro subgrupo é feito por uma lógica NOT de uma variável, a entrada 1.

Este subgrupo é o último.

Serão solicitadas ao operador as seguintes perguntas:

Qual o número da transição cuja receptividade será descrita?

Qual a operação lógica do subgrupo?

Quantas variáveis existem neste subgrupo?

Existe mais subgrupos nesta receptividade?

Se existem mais subgrupos, qual a função lógica que relaciona estes dois subgrupos?

Para cada variável deste subgrupo questiona se é uma entrada, saída ou etapa? Qual o número de borne desta variável?

Existem duas funções que podem ser inclusas nas receptividades, o temporizador e a "nenhuma".

A função "nenhuma" só poderá ser inclusa no primeiro subgrupo, não existirão outros subgrupos relacionados a esta receptividade. Através desta função uma receptividade tornar-se-á verdadeira no teste seguinte à ativação da etapa de origem do arco de ligação de etapa à transição. Tornando verdadeira a etapa de destino do arco de ligação de transição à etapa.

A função temporizador em uma transição está relacionada ao teste de ativação de uma determinada etapa. Assim o temporizador iniciará a contagem quando uma etapa X especificada tornar-se ativa e o temporizador tornar-se-á ativo decorridos Y teste da ativação da etapa.

Para a função temporizador, serão solicitadas ao operador as seguintes questões:

Qual o número da etapa que inicializará a contagem dos testes?

Quantos testes deverão decorrer para a ativação do temporizador?

Para finalizar a descrição das receptividades associadas às transições o operador deverá teclar <F>. Para descrever uma nova transição deverá teclar <ENTER>.

3.5.9. Etapa Inicialmente Ativa

O SIMGRAF solicitará ao operador a descrição dos números das etapas que estarão ativas no instante inicial da simulação.

Para finalizar a descrição dos números das etapas, deverá teclar <F> ou teclar <ENTER> para descrever nova etapa ativa em t_0.

Uma vez finalizada as fases de descrição do objeto a ser simulado, o SIMGRAF questiona se o operador deseja que sejam exibidos os dados de descrição do objeto.Para isto deverá responder com <S> ou <N>.

Caso opção afirmativa, o SIMGRAF exibirá os dados na sintaxe que os dados são coletados do arquivo .GFT, ou seja, separados em vetores V1 (entradas ativas em t_0), V2 (saídas ativas em t_0), V3 (etapas ativas em t_0) e as matrizes M1 (ligações etapas-transições), M2 (ligações transições-etapas), M3 (ações das etapas) e M4 (receptividades das transições).

3.6. Simulação

Neste ponto o SIMGRAF já terá todos os dados para inicializar a simulação do objeto descrito. A simulação será realizada com entradas e saídas por vídeo, teclado ou portas paralelas conforme descrito no início do SIMGRAF.

Caso a opção tenha sido feita por entrada de dados via teclado e saída por vídeo, o operador deverá descrever todas as mudanças de estado das entradas que deseja que ocorram em um mesmo ciclo de testes, através da opção <E> e em seguida solicitar ao SIMGRAF que execute o teste através da opção <T>.

Poderá finalizar a simulação com a opção <F>.

Caso a opção tenha sido por entrada e saída de dados via porta paralela simulada, o operador deverá converter cada entrada como um bit de uma palavra de 16 bits para um número inteiro entre 0 e 65535, onde a entrada número 1 é o bit menos significativo. De maneira análoga a saída 1 será o bit menos significativo de uma palavra de 16 bits que será convertido para número inteiro.

Deverá também utilizar as teclas de <T> e <F> para executar o teste e finalizar a simulação, respectivamente.

Caso a opção tenha sido por entrada e saída de dados via porta paralela real, o operador deverá colocar as chaves de entrada nas posições desejadas e então teclar <T> para executar o teste do SIMGRAF. Observando as respostas nos leds de saída ou vídeo.

Para finalizar a simulação do objeto o operador deverá utilizar a tecla <F>.

3.7. Descrição do SIMGRAF através da Linguagem Orientada a Objeto (LOO)

3.7.1. Classes , Subclasses e Métodos

Como o SIMGRAF foi feito em Linguagem Orientada a Objeto (LOO). a sua estrutura de programação foi dividida em classes, subclasses e métodos.

Esta estrutura será exposta nas tabelas 3.1, 3.2, 3.3, 3.4 e 3.5 e seus pontos principais serão explicados em seguida.

O SIMGRAF apresenta as seguintes classes, subclasses e principais métodos:

classes	subclasses	métodos
PRINCIPAL		JANELA CONFERIR
	CONSTANTES	LE_CONSTANTES RANGE
	LER_ET_TRA	INICIALIZA LER_ET LER_TE LOOP
	AÇÃO	INICIALIZA LER_AÇÃO LOOP
	ESTADO	INICIALIZA ETAPA_INICIAL LOOP
	LER_ENT_SAI	INICIALIZA LER_ENTRADA LER_SAÍDA LOOP
	RECEPTIVIDADE	INICIALIZA LER_AÇÃO LOOP
	PROGRAMA	TESTA_ENTRADA TESTA_SAÍDA TESTA_ETAPA TESTA_TRANSIÇÃO E OU NÃO TEMPO TESTA_RECEPTIVIDADE TESTA_COND CONVERTE SETA_ENTRADA SETA_SAÍDA SETA_ETAPA SETA_TRANSIÇÃO RESETA_TEMPO LOOP
	TELA	ETAPA TRANSIÇÃO ENTRADA SAÍDA TEMPO
	EXIBIR	AGUARDA IMPRIME_V1 IMPRIME_V2 IMPRIME_V3 IMPRIME_M1 IMPRIME_M2 IMPRIME_M3 IMPRIME_M4 LOOP

tab 3.1 - Classe **PRINCIPAL** *e suas Subclasses e Métodos*

LEITURA		ABRE_ARQUIVO FECHA_ARQUIVO LE_BLOCO LER_CARACTER LER_BOOLEANO_V1 LER_BOOLEANO_V2 LER_BOOLEANO_V3 LER_STRING PROCURA_CARACTER MONTA_M1 MONTA_M2 LE_CONSTANTES_ARQUIVO LOOP

tab 3.2 - Classe **LEITURA** *e suas Subclasses e Métodos*

TEMPORIZADOR		INICIALIZA INCREMENTA

tab 3.3 - Classe **TEMPORIZADOR** *e suas Subclasses e Métodos*

OUTPUT		TRANSFORMA_SAIDA

tab 3.4 - Classe **OUTPUT** *e suas Subclasses e Métodos*

INPUT		TRANSFORMA_ENTRADA

tab 3.5 - Classe **INPUT** *e suas Subclasses e Métodos*

3.7.2. Descrição das Classes, Subclasses e Métodos

Neste tópico serão explanadas as funções das classes, subclasses e principais métodos.

3.7.2.1. Classe Principal e Subclasses

Classe que contém métodos que serão utilizados nas suas subclasses. Os parâmetros utilizados por estes métodos devem ser do mesmo tipo (type) em todas as suas chamadas. Não podendo ,por exemplo, em um momento ser chamado com um parâmetro do tipo integer e em outro instante ser chamado com um parâmetro do tipo real.

Os métodos que compõem estas classe são: janela e conferir.

JANELA : Método que define a cor de fundo e do texto que serão exibidos a partir de sua chamada.

CONFERIR : Método que confere se o valor digitado está ou não dentro de um intervalo entre um e o parâmetro passado na chamada do método.

3.7.2.1.1. Constantes

Subclasse da classe PRINCIPAL, é responsável pela coleta do dimensionamento do objeto a ser simulado quando.

Os métodos que compõem esta subclasse são: le_constantes e range.

LE CONSTANTES : Método responsável pela chamada do objeto RANGE para cada uma das oito dimensões (etapa, transição, entrada, saída, máximo número de variáveis por subgrupo de receptividade, máximo número de subgrupos por receptividade, número máximo de testes de simulação e máximo número de ações por etapa) que permitirão o dimensionamento do objeto a ser simulado pelo SIMGRAF.

RANGE : Método responsável pela leitura do dados e verificação se o valor digitado está entre o mínimo e o máximo aceitos pelo SIMGRAF.

3.7.2.1.2. Ler_et_tra

Subclasse da classe PRINCIPAL, é responsável pela coleta dos arcos de ligação entre etapas e transições (matriz M1) e entre as transições e etapas (matriz M2). Permitindo ao SIMGRAF reconhecer o objeto a ser simulado.

Os métodos que compõem esta subclasse são: inicializa, ler_et, ler_te e loop.

INICIALIZA : Método que inicializa as matrizes M1 e M2 com todos os seus elementos false.

LER_ET : Método que faz a leitura da i-ésima etapa de partida do arco de ligação existente, bem como da j-ésima transição de chegada do respectivo arco. Setando o elemento $M1_{ij}$.

LER_TE : Método que faz a leitura da j-ésima transição de partida do arco de ligação existente, bem como da i-ésima etapa de chegada do respectivo arco. Setando o elemento $M2_{ji}$.

LOOP : Método que contém o loop principal desta subclasse que primeiramente faz a leitura dos dados de M1, chamando LER_TE, até que seja finalizada a descrição dos arcos de ligação entre etapas e transições digitando-se <F>. Em seguida faz a leitura dos dados de M2 finalizando a descrição dos arcos de ligação entre transições e etapas, chamando LER_TE, até que seja digitando-se <F>.

3.7.2.1.3. Ação

Subclasse da classe PRINCIPAL, é responsável pela coleta das ações das etapas, quando estas se tornarem ativas.

Os métodos que compõem esta subclasse são: inicializa, etapa_acao e loop.

INICIALIZA : Método que inicializa a matriz M3, matriz de ações das etapas com todos os elementos nulos;

LER_ACAO : Método que faz a leitura das ações de uma etapa, conforme descrito em SINTAXE DA DESCRIÇÃO DO OBJETO a ser simulado pelo SIMGRAF.

LOOP : Método que contém o loop principal desta subclasse que fará a chamada do método LER_AÇÃO até que seja digitado <F>, quando todas as etapas tiverem sido entradas.

3.7.2.1.4. Estado

Subclasse da classe PRINCIPAL, é responsável pela coleta das etapas que estarão ativas no instante inicial da simulação.

Os métodos que compõem esta subclasse são: inicializa etapa_inicial e loop.

INICIALIZA : Método que inicializa os vetores de etapas ativas (V3) e transições ativas (V4) com todos os elementos false;

ETAPA_INICIAL : Método que faz a leitura da etapa ativa no instante inicial da simulação, setando o elemento do vetor V3.

LOOP : Método que contém o loop principal desta subclasse que fará a chamada do método ETAPA_INICIAL até que seja digitado <F>, quando todas as etapas inicialmente ativas tiverem sido entradas.

3.7.2.1.5. Ler_ent_sai

Subclasse da classe PRINCIPAL, é responsável pela coleta das entradas e saídas que estarão ativas no início da simulação.

Os objetos que compõem esta subclasse são: inicializa, ler_entrada, ler_saída e loop.

INICIALIZA : Método que inicializa os vetores das entradas ativas (V1) e de saídas ativas (V2) com todos os seus elementos false;

LER_ENTRADA : Método que faz a leitura das entradas quer estarão ativas no início de simulação. Armazenando os dados no vetor V1.

LER_SAIDA : Método que faz a leitura das saídas que estarão ativas no início da simulação. Armazenando os dados no vetor V2.

LOOP : Método que contém o loop principal desta subclasse que primeiramente faz a leitura dos dados de V1, executando LER_ENTRADA, até que seja digitado <F>. Em seguida faz a leitura dos dados de V2, executando LER_SAÍDA, até que seja digitado <F>.

3.7.2.1.6. Receptividade

Subclasse da classe PRINCIPAL, é responsável pela coleta de dados das receptividades das transições do objeto a ser simulado pelo SIMGRAF.

Os métodos que compõem esta subclasse são: inicializa, ler_ação e loop.

INICIALIZA : Método que inicializa os elementos da matriz de receptividade (M4) com todos os elementos nulos;

LER_ACAO : Método que faz a leitura da matriz dos dados da receptividade de uma transição, conforme descrito na "SINTAXE DA DESCRIÇÃO DO OBJETO".

LOOP : Método que contém o loop principal desta subclasse que fará a chamada do método LER_AÇÃO até que seja digitado <F>, quando todas as receptividades tiverem sido digitadas.

3.7.2.1.7. Programa

Subclasse da classe PRINCIPAL, é a responsável pela avalia ção das receptividades das transições e execução das ações das etapas.

Os métodos que compõem esta subclasse são: testa_entrada, testa_saída, testa_etapa, testa_transição, e, ou, não, tempo, testa_receptividade, testa_cond, converte, seta_entrada, seta_saída, seta_etapa, seta_transição, reseta_tempo, altera_entrada e loop.

TESTA_ENTRADA : Método que testa se uma entrada está ativa ou não;

TESTA_SAIDA : Método que testa se uma saída está ativa ou não;

TESTA_ETAPA : Método que testa se uma etapa está ativa ou não;

TESTA_TRANSICAO : Método que testa se uma transição está ativa ou não;

E : Método que avalia a função lógica AND é verdadeira ou falsa para os dois parâmetros apresentados;

OU : Método que avalia se a função lógica OR é verdadeira ou falsa para os dois parâmetros apresentados;

NAO : Método que avalia se a função lógica NOT é verdadeira ou falsa para o parâmetro apresentado;

TEMPO : Método que controla o contador de testes das receptividades temporizadas.

TESTA RECEPTIVIDADE : Método que avalia se uma receptividade é verdadeira ou falsa;

TESTA COND : Método que testa se uma ação condicional é verdadeira no momento em que a etapa torna-se verdadeira.

CONVERTE : Método que avalia se um variável da matriz de receptividade é momentaneamente verdadeira ou falsa, seja entrada, saída, etapa ou transição.

SETA ENTRADA : Método que modifica o valor lógico de uma entrada;

SETA SAIDA : Método que modifica o valor lógico de uma saída;

SETA ETAPA : Método que modifica o valor lógico de uma etapa;

SETA TRANSICAO : Método que modifica o valor lógico de uma transição;

RESETA TEMPO : Método que zera o contador das receptividades temporizadas.

LOOP : Método que contém o loop principal desta subclasse, onde são executadas as leituras das entradas, avaliação das receptividades e modificação das saídas segundo as ações das etapas. Este loop pode ser interrompido pela quantidade de testes realizados na simulação (definido no dimensionamento do objeto) ou por solicitação do usuário.

ALTERA ENTRADA : Método que aguarda a entrada de dados que permitam a mudança do status de uma entrada. Este método é utilizado na simulação feita utilizando-se entrada e saída de dados via teclado e não por porta paralelas.

3.7.2.1.8. Tela

Classe responsável pela exibição do status (ativo ou não) da simulação do SIMGRAF. Controlando a exibição do status das entradas, saídas, etapas, transições e contador de testes do sistema.

Os métodos que compõem esta classe são: etapa, transição, entrada, saída e tempo.

ETAPA : Método que exibe na tela o status das etapas do objeto que está sendo simulado.

TRANSICAO : Método que exibe na tela o status das transições do objeto que está sendo simulado após o último teste do SIMGRAF.

ENTRADA : Método que exibe na tela o status das entradas do objeto que está sendo simulado.

SAIDA : Método que exibe na tela o status das saídas do objeto que está sendo simulado.

TEMPO : Método que exibe na tela o número de testes do objeto que está sendo simulado.

3.7.2.1.9. Exibir

Classe responsável pela exibição da descrição do objeto ao usuário, caso assim seja desejado.

Esta descrição é feita por partes (telas).

Os métodos que compõem esta classe são: aguarda, imprime_v1, imprime_v2, imprime_v3, imprime_m1, imprime_m2,imprime_m3, imprime_m4 e loop.

AGUARDA : Método que aguarda até que a tecla <ENTER> seja pressionada.

IMPRIME V1 : Método que exibe o status do vetor de entrada (V1).

IMPRIME V2 : Método que exibe o status do vetor de saída (V2).

IMPRIME V3 : Método que exibe o status do vetor de etapas ativas (V3).

IMPRIME M1 : Método que exibe a matriz com os arcos de ligação entre etapas e transições (M1).

IMPRIME M2 : Método que exibe a matriz com os arcos de ligação entre transições e etapas (M2).

IMPRIME M3 : Método que exibe as ações de cada etapa separadamente.

IMPRIME M4 : Método que exibe as receptividades de cada transição separadamente.

LOOP : Método que faz a exibição da descrição do objeto a ser simulado através da chamada dos outros métodos desta classe.

3.7.2.2. Classe Temporizador

Classe responsável pela manipulação do relógio interno do SIMGRAF.

Os métodos que compõem esta classe são: inicializa e incrementar.

INICIALIZA : Método que inicializa o contador de teste do SIMGRAF, setando-o em zero.

INCREMENTA : Método que incrementa em um contador de testes do SIMGRAF.

3.7.2.3. Classe Leitura

Classe responsável pela leitura do objeto a ser simulado via arquivo de dados. Efetuando tanto a leitura de dados do arquivo que contém os dados de dimensionamento do objeto quanto do arquivo de dados que descreve o objeto propriamente dito.

É interressante lembrar que ambos os dados poderão ser descritos no mesmo arquivo.

Os métodos que compõem esta classe são: abre_arquivo, fecha_arquivo, le_bloco, ler_caracter, ler_booleano_v1, ler_booleano_v2, ler_booleano_v3, ler_string, procura_caracter, monta_m1, monta_m2, le_constantes_arquivo e loop.

ABRE ARQUIVO : Método que abre o arquivo desejado permitindo ao SIMGRAF a leitura de seus dados.

FECHA ARQUIVO : Método que fecha o arquivo utilizado

para leitura de dados.

LE BLOCO : Método que lê no arquivo aberto o próximo caracter numérico.

LER CARACTER : Método que procura no arquivo aberto a próxima ocorrência do caracter desejado.

LER BOOLEANO V1 : Método que lê o vetor inicial de entradas (V1).

LER BOOLEANO V2 : Método que lê o vetor inicial de saídas (V2).

LER BOOLEANO V3 : Método que lê o vetor inicial de etapas ativas (V3).

LER STRING : Método que procura no arquivo aberto a próxima ocorrência da string desejada.

PROCURA CARACTER : Método que procura no arquivo aberto um determinado número de vezes a ocorrência de um caracter.

MONTA M1 : Método que lê a matriz dos arcos de ligação entre etapas e transições.

MONTA M2 : Método que lê a matriz dos arcos de ligação entre transições e etapas.

LE CONSTANTES ARQUIVO : Método que lê as dimensões do sistema a ser simulado no SIMGRAF.

LOOP : Método que faz a leitura da descrição do objeto a ser simulado através da chamada dos outros objetos desta classe.

3.7.2.4. Classe Output

Classe responsável pelo controle das portas de saída paralelas, quando conectadas ao SIMGRAF.

O método que compoe esta classe é: transforma_saida.

TRANSFORMA SAÍDA : Método que faz a conversão do vetor de saídas em sinais de tensão nas portas de saída paralelas.

3.7.2.5. Classe Input

Classe responsável pelo controle das portas de entrada paralelas, quando conectadas ao SIMGRAF.

O método que compoe esta classe é: transforma_entrada.

TRANSFORMA ENTRADA : Método que faz a conversão de sinais de tensão de entrada nas portas paralelas para o vetor de entradas.

3.8. Descrição da Interface de I/O

Para comunicação com o meio externo foi criada uma opção do uso de interface de entrada/saída (I/O) digital conectada ao barramento do microcomputador compatível com o IBM-PC.

A interface utilizada é a SDC2817 [17] que possui cinco endereços (32 linhas) de entradas e saídas. As 8 primeiras linhas são destinadas ao registrador de controle da interface (endereço inicial) definido pelo usuário pela alocação física no barramento do computador, devendo ser informado ao SIMGRAF.

No endereço inicial, a interface é configurada quanto ao direcionamento dos outros quatro registros que a compõem. Os quatro registros, cujos endereços são consecutivos ao endereço inicial, podem ser configuradas como entrada ou saída.

O SIMGRAF está utilizando o endereço default como o 228H onde é escrito o byte 0CH, configurando as duas primeiras portas, 229H e 230H, são utilizadas para a entrada e as duas últimas são utilizadas para saída, 231H e 232H.

Neste capítulo foi descrito o Simulador de GRAFCET (SIMGRAF), através da explanação das suas classes, subclasses e métodos. Observou-se também a forma em que o objeto a ser simulado deve ser descrito ao SIMGRAF utilizando-se a entrada de dados via teclado. No capítulo seguinte serão descritos os exemplos de simulações realizados com o SIMGRAF.

Capítulo 4

Testes de Simulação com o SIMGRAF

4.1. Introdução

Neste capítulo serão descritos alguns exemplos em que o SIMGRAF foi utilizado, objetivando testar a correção do GRAFCET criado com o caderno de tarefas proposto.

Não existe o objetivo de trabalhar com situações reais, mas sim de testar as capacidades do SIMGRAF.

Serão executadas três simulações sendo elas:

1o. Um carro transportador em um monotrilho;

2o. Uma estação de trabalho com mesa giratória para três diferentes tarefas consecutivas;

3o. Uma câmara de TV ao centro de um pentágono, onde cada vértice possui uma pessoa a ser focada.

4.2. Simulação 1

A simulação será feita passando-se pelas fases desde as necessidades do consumidor até a simulaçao do GRAFCET proposto pelo executor.

4.2.1 Caderno de Tarefas

A seguir serão apresentadas as descrições dos cadernos de tarefas associadas aos experimentos. Conforme foi descrito no capítulo 3 deste trabalho, esta descrição pode ser dividida em dois níveis.

4.2.1.1. 1o. Nível

Um carrinho está sobre um trilho e possui a posição inicial no ponto médio do trilho. Deste ponto pode se deslocar para qualquer dos lados quando pressionada uma das duas botoeiras disponíveis.

Quando está no ponto médio existem duas sinalizadoras indicando a posição. A sinalizadora correspondente ao lado para o qual o carrinho se desloca será apagada no momento em que o carrinho deixar o ponto médio.

Ao chegar no ponto final do percurso o carrinho retornará ao ponto inicial aguardando novo comando. Nas extremidades do trilho existem duas sinalizadoras, uma de cada lado, indicando que o carrinho atingiu o ponto. A sinalizadora da extremidade somente será desligada quando o carrinho atingir o ponto médio.

Caso as duas botoeiras sejam pressionadas no mesmo instante será aceso um alarme temporizado, o mesmo acontecendo se o botão ordenando movimentação em direção oposta estiver pressionado quando o carrinho atingir o final de um dos percursos.

4.2.1.2. 2o. Nível

Um carrinho está parado em um monotrilho no ponto médio (PM). Quando o botão BE é pressionado o carrinho se movimenta para a esquerda, sendo apagada a luz verde (VD) que estava acesa. Ao chegar a extremidade esquerda do monotrilho o carrinho aciona um sensor de final de curso (P1), fazendo com que seja acesa a luz amarela (AM) e o carrinho retorne ao ponto inicial. Ao chegar ao ponto inicial novamente o carrinho pára e a luz verde volta a acender apagando a luz amarela. Se ao invés de pressionar o botão BE o botão BD seja pressionado o carrinho se movimentará para a direita apagando a luz branca (BR). Ao chegar a extremidade direita do monotrilho o carrinho acionará o sensor (P2) fazendo com que a luz azul (AZ) seja acesa e o carrinho retorne ao ponto inicial. Ao chegar ao ponto inicial o carrinho pára, a luz branca é acesa e a luz azul apagada. Neste ponto o sistema fica esperando o próximo comando.

Caso sejam pressionados simultaneamente BD e BE o sistema ficará numa situação de alarme por 3 ciclos, acendendo a luz vermelha (VM), e depois retornará ao estado de espera de comando. Caso enquanto o carrinho esteja indo para uma das extremidades for pressionado um botão será gerado um alarme que será desligado no próximo ciclo.

4.2.2. Definição dos Pontos

Descrição	Ponto	Tipo	BORNE
Motor para a esquerda	ME	Saída	1
Motor para a direita	MD	Saída	2
Luz verde - centro esquerdo	VD	Saída	3
Luz vermelha - alarme	VM	Saída	4
Luz Azul - extremo direito	AZ	Saída	5
Luz amarela - extremo esquerdo	AM	Saída	6
Luz branca - centro direito	BR	Saída	7
Sensor de posição - ponto médio	PM	Entrada	1
Sensor de posição - ponto esquerdo	P1	Entrada	2
Sensor de posição - ponto direito	P2	Entrada	3
Botoeira direita	BD	Entrada	4
Botoeira esquerda	BE	Entrada	5

4.2.3. Esquema Físico

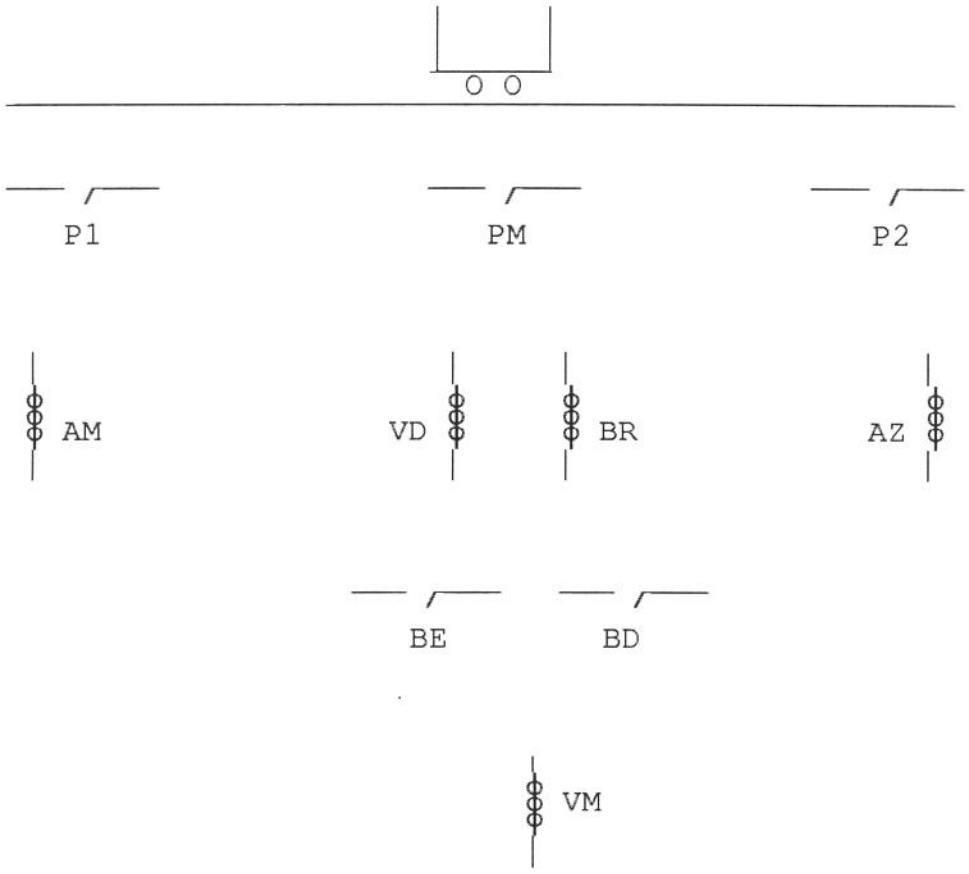

fig 4.1 - Simulação 1 - Esquema Físico

4.2.4. GRAFCET

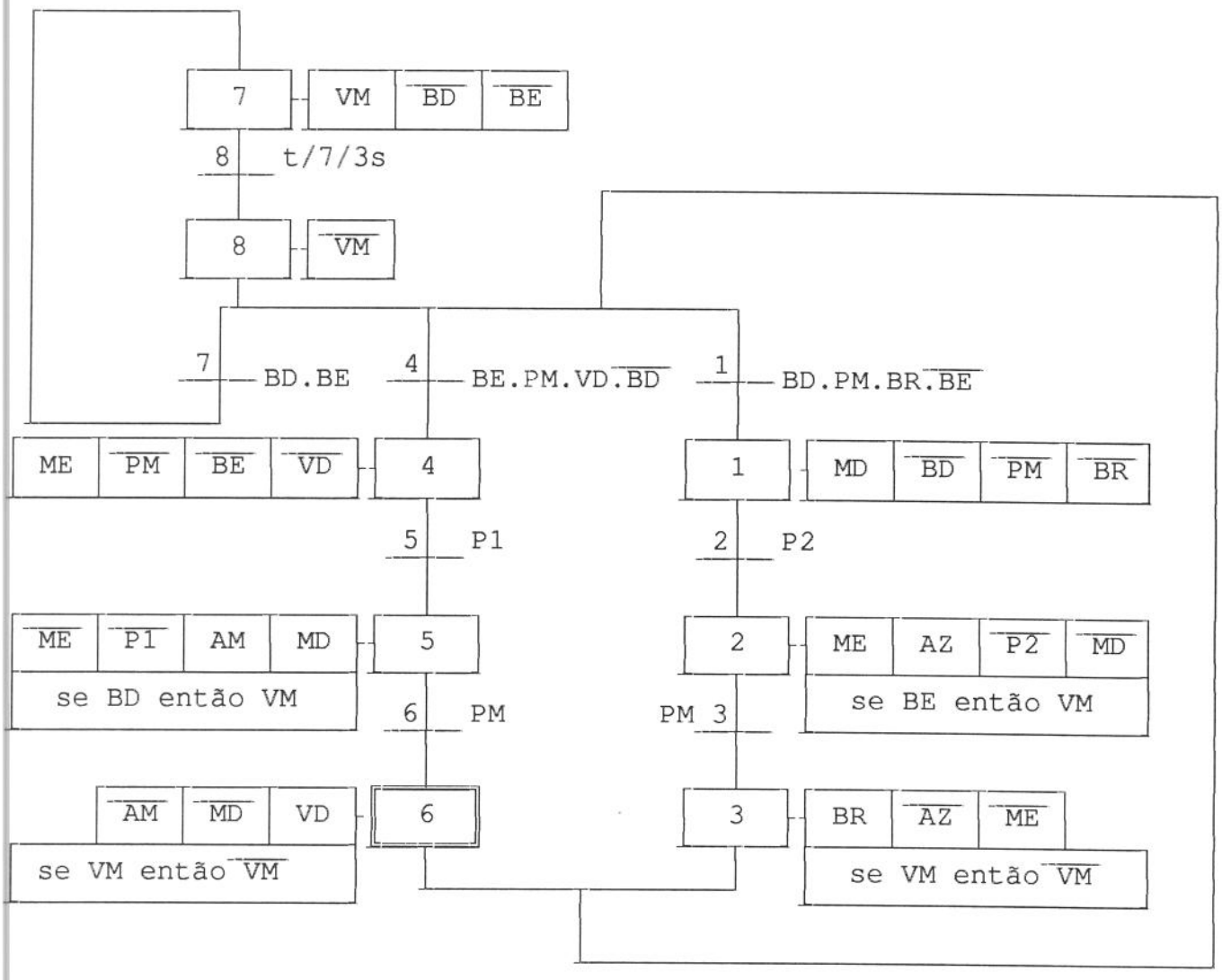

fig 4.2 - Simulação 1 - GRAFCET

4.2.5. Descrição da Simulação para o SIMGRAF

A seguir serão definidas as variáveis e parâmetros da SIMULAÇÃO 1 a serem utilizados pelo SIMGRAF. As variáveis serão denominadas de acordo com as definições fornecidas no capítulo de descrição do sistema SIMGRAF.

Descrição do experimento - Parâmetros do SIMGRAF

Número de Etapas	8
Número de Transições	8
Número de Entradas	5
Número de Saídas	7
Número máximo de variáveis por subgrupo	3
Número máximo de subgrupo por transição	2
Número máximo de bornera	8
Dezena máxima do número de testes	5
Número máximo de ações por etapa	4

A seguir é descrito o experimento, seguindo a sintaxe criada para a descrição do objeto a ser simulado pelo SIMGRAF. Esta sintaxe é descrita no Anexo B.

Vetor de parametrização do SIMGRAF
E1
A8 8 5 7 4 2 8 2 5 A

Vetor de entradas inicialmente ativas
V1
A1 0 0 0 0 A

Vetor de saídas inicialmente ativas
V2
A0 0 1 0 0 0 1 A

```
Vetor de etapas inicialmente válidas
V3
A0 0 0 0 0 1 0 0 A


Matriz de ligações etapas ⇒ transições

 1 2 3 4 5 6 7 8
M1
A0 1 0 0 0 0 0 0 A        1
A0 0 1 0 0 0 0 0 A        2
A1 0 0 1 0 0 1 0 A        3
A0 0 0 0 1 0 0 0 A        4
A0 0 0 0 0 1 0 0 A        5
A1 0 0 1 0 0 1 0 A        6
A0 0 0 0 0 0 0 1 A        7
A1 0 0 1 0 0 1 0 A        8


Matriz de ligações transições ⇒ etapas
 1 2 3 4 5 6 7 8
M2
A1 0 0 0 0 0 0 0 A        1
A0 1 0 0 0 0 0 0 A        2
A0 0 1 0 0 0 0 0 A        3
A0 0 0 1 0 0 0 0 A        4
A0 0 0 0 1 0 0 0 A        5
A0 0 0 0 0 1 0 0 A        6
A0 0 0 0 0 0 1 0 A        7
A0 0 0 0 0 0 0 1 A        8
```

```
Matriz tridimensional de ações
M3
A4 C102 1 0 0  C
   C4 2 0 0  C
   C1 2 0 0  C
   C107 2 0 0  C
A5 C101 1 0 0  C
   C105 1 0 0  C
   C3 2 0 0  C
   C102 2 0 0  C
   C104 1 1 5  C
A4 C107 1 0 0  C
   C105 2 0 0  C
   C101 2 0 0  C
   C104 2 1 104  C
A4 C101 1 0 0  C
   C1 2 0 0  C
   C5 2 0 0  C
   C103 2 0 0  C
A5 C101 2 0 0  C
   C2 2 0 0  C
   C106 1 0 0  C
   C102 1 0 0  C
   C104 1 1 4  C
A4 C106 2 0 0  C
   C102 2 0 0  C
   C103 1 0 0  C
   C104 2 1 104  C
A3 C104 1 0 0  C
   C4 2 0 0  C
   C5 2 0 0  C
A1 C104 2 0 0  C
```

```
Matriz tridimensional de receptividades
M4
A2 C1 3 1 1 4 1 107  C
   C3 1 2 0 5  C
A1 C1 1 2 0 3  C
A1 C1 1 2 0 1  C
A2 C1 3 1 1 5 1 103  C
   C3 1 2 0 4  C
A1 C1 1 2 0 2  C
A1 C1 1 2 0 1  C
A1 C1 2 2 0 4 5  C
A1 C5 1 2 0 7 3 0 0  C
```

4.3. Simulação 2

4.3.1. Caderno de Tarefas

4.3.1.1 1o. Nível

O sistema é composto por uma mesa giratória de três posições, de 120° em 120°, por uma correia de alimentação de peças, por uma furadeira e por uma rosqueadeira.

As peças chegam pela correia de alimentação de onde são transportadas através de um pistão com ventosa que faz o carregamento e o descarregamento das peças que estão na mesa giratória.

As peças são giradas da estação de carregamento para a estação de trabalho 1, onde serão realizados dois furos com a mesma profundidade um em cada uma das posições pré-determinadas. A furadeira será movimentada de um furo para o outro através de um pistão pneumático.

As peças são giradas para a estação de trabalho 2, onde o furo A sofrerá o processo de rosqueamento interno até a profundidade determinada.

4.3.1.2. 2o. Nível

O sistema é composto por uma mesa giratória, que através de sensor indutivo detecta as três marcações existentes formando entre si 120^0. A correia de alimentação de peças a serem usinadas possui um detector ótico para verificar a existência ou não de peças na posição que permite a transferência de peças para a mesa.

O transporte de peças é feito de um pistão com sensores de final de curso e ainda equipado com pistão para movimentação da ventosa, permitindo que as peças sejam retiradas da mesa giratória. O pistão de movimentação da ventosa possui também sensores de final de curso.

Na estação 1, a furadeira possui detectores de posição na vertical (posição de repouso, posição de aproximação da peça a ser perfurada e posição de profundidade do orifício) e ainda horizontal (posição do furo 1 e posição do furo 2). A furadeira também possui a velocidade vertical de perfuração e a velocidade vertical de aproximação.

Na estação 2, a rosqueadeira irá fazer rosca no furo 1 a mesma possui velocidade vertical de aproximação e velocidade vertical de operação. Existem também sensores de posição (posição de repouso, posição de aproximação e posição de profundidade da rosca).

4.3.2. Descrição do Experimento - Definição dos Pontos

Descrição	Ponto	Tipo	Borne
Pistão de transferência da peça da correia para a mesa giratória.Posição desligada é retraído.	T_1	Saída	1
Pistão da ventosa para vácuo. Posição desligada é distendida	T_2	Saída	2
Correia de transporte de peças até o pistão T_1	C_1	Saída	3
Furadeira da estação 1	F_1	Saída	4
Velocidade rápida de descida da furadeira F_1	V_1	Saída	5
Velocidade lenta de descida da furadeira	V_2	Saída	6
Velocidade lenta de subida da furadeira	V_3	Saída	7
Pistão de movimentação horizontal de furadeira F_1 Posição desligada é retraído.	T_3	Saída	8
Motor da rosqueadeira	M_1	Saída	9
Velocidade rápida de descida da rosqueadeira	V_4	Saída	10
Velocidade lenta de descida da rosqueadeira	V_5	Saída	11
Velocidade lenta da subida da rosqueadeira	V_6	Saída	12
Rotação do sentido horário da mesa de trabalho	G_1	Saída	13
Sensor de posição de T_1 retraído	P_{11}	Entrada	1
Sensor de posição de T_1 distendido	P_{12}	Entrada	2
Sensor de posição de T_2 retraído	P_{13}	Entrada	3
Sensor de posição de T_2 distendido	P_{14}	Entrada	4
Sensor de detecção de peça na correia	O_1	Entrada	5
Sensor superior de movimentação vertical de F_1	S_1	Entrada	6
Sensor médio de movimentação vertical de F_1	S_2	Entrada	7
Sensor inferior de movimentação vertical de F_1	S_3	Entrada	8
Sensor de posição de T_3 distendido	P_2	Entrada	9
Sensor de posição de T_3 retraído	P_3	Entrada	10
Sensor superior de movimentação vertical de M_1	S_4	Entrada	11
Sensor médio de movimentação vertical de M_1	S_5	Entrada	12
Sensor inferior de movimentação vertical de M_1	S_6	Entrada	13
Sensor de rotação de 120° da mesa	R_1	Entrada	14
Botão de inicialização/continuação	B_1	Entrada	15

4.3.3. Esquema Físico

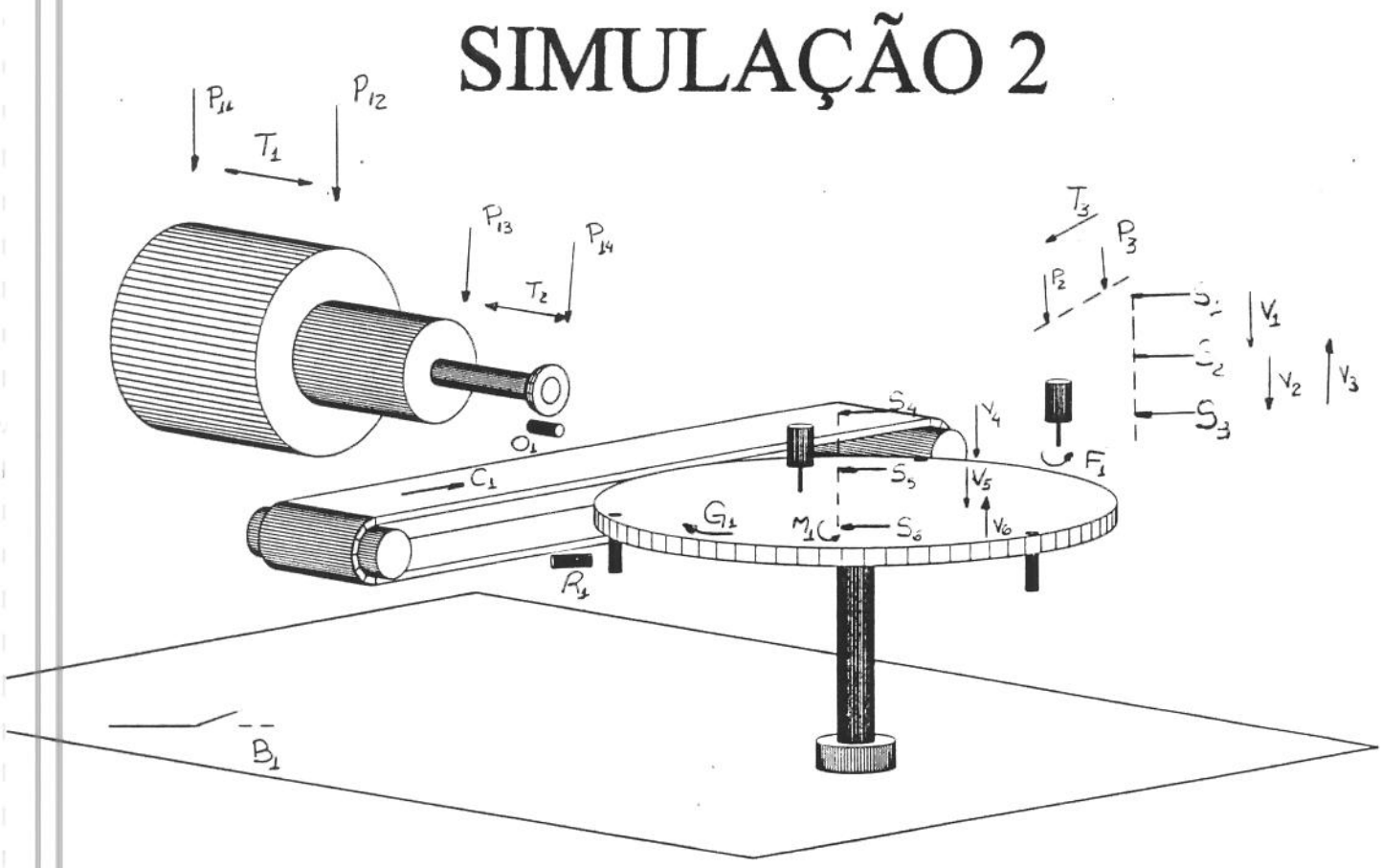

fig 4.3 - Simulação 2 - Esquema Físico

4.3.4. GRAFCET

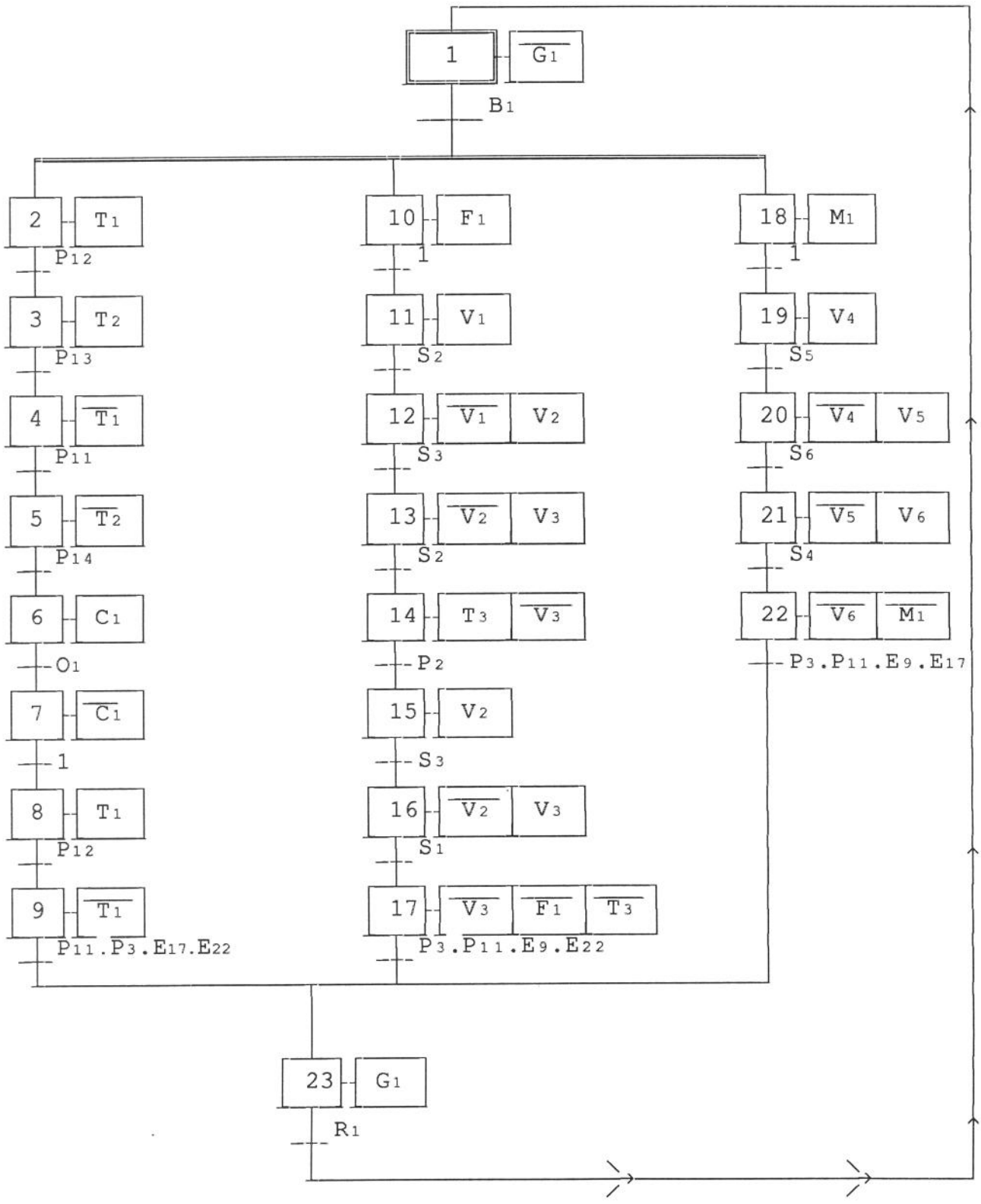

fig 4.4 - Simulação 2 - GRAFCET

4.3.5. Descrição da Simulação para o SIMGRAF

Número de Etapas	23
Número de Transições	23
Número de Entradas	15
Número de Saídas	13
Número máximo de variáveis por subgrupo	4
Número máximo de subgrupo por transição	1
Número máximo de bornera	23
Dezena máxima do número de testes	5
Número máximo de ações por etapa	3

A seguir é descrito o experimento, seguindo a sintaxe criada para a descrição do objeto a ser simulado pelo SIMGRAF. Esta sintaxe é descrita no Anexo B.

```
Vetor de parametrização do SIMGRAF
E1
A23 23 15 13 4 1 23 5 3 A

Vetor de entradas inicialmente ativas
V1
A0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 A

Vetor de saídas inicialmente ativas
V2
A0 0 0 0 0 0 0 0 0 0 0 0 0 0 A

Vetor de etapas inicialmente ativas
V3
A1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 A
```

Matriz de ligações etapas ⇒ transições

```
 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3
M1
A1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 A        1
A0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 A        2
A0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 A        3
A0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 A        4
A0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 A        5
A0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 A        6
A0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 A        7
A0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 A        8
A0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 A        9
A0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 A       10
A0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 A       11
A0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 A       12
A0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 A       13
A0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 A       14
A0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 A       15
A0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 A       16
A0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 A       17
A0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 A       18
A0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 A       19
A0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 A       20
A0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 A       21
A0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 A       22
A0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 A       23
```

Matriz de ligações transições ⇒ etapas

	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3		
M2																									
A0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	A		1
A0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	A		2
A0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	A		3
A0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	A		4
A0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	A		5
A0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	A		6
A0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	A		7
A0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	A		8
A0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	A		9
A0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	A		10
A0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	A		11
A0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	A		12
A0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	A		13
A0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	A		14
A0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	A		15
A0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	A		16
A0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	A		17
A0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	A		18
A0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	A		19
A0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	A		20
A0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	A		21
A0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	A		22
A1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	A		23

```
Matriz tridimensional de ações
M3
A1 C113 2 0 0  C
A1 C101 1 0 0  C
A1 C102 1 0 0  C
A1 C101 2 0 0  C
A1 C102 2 0 0  C
A1 C103 1 0 0  C
A1 C103 2 0 0  C
A1 C101 1 0 0  C
A1 C101 2 0 0  C
A1 C104 1 0 0  C
A1 C105 1 0 0  C
A2 C106 1 0 0  C
   C105 2 0 0  C
A2 C107 1 0 0  C
   C106 2 0 0  C
A2 C108 1 0 0  C
   C107 2 0 0  C
A1 C106 1 0 0  C
A2 C107 1 0 0  C
   C106 2 0 0  C
A3 C107 2 0 0  C
   C108 2 0 0  C
   C104 2 0 0  C
A1 C109 1 0 0  C
A1 C110 1 0 0  C
A2 C110 2 0 0  C
   C111 1 0 0  C
A2 C111 2 0 0  C
   C112 1 0 0  C
A2 C109 2 0 0  C
   C112 2 0 0  C
A1 C113 1 0 0  C
```

```
Matriz tridimensional de receptividade
M4
A1 C1 1 2 0 15  C
A1 C1 1 2 0 2  C
A1 C1 1 2 0 3  C
A1 C1 1 2 0 1  C
A1 C1 1 2 0 4  C
A1 C1 1 2 0 5  C
A1 C4  C
A1 C1 1 2 0 2  C
A1 C1 4 2 0 10 1 217 222  C
A1 C4  C
A1 C1 1 2 0 7  C
A1 C1 1 2 0 8  C
A1 C1 1 2 0 7  C
A1 C1 1 2 0 9  C
A1 C1 1 2 0 8  C
A1 C1 1 2 0 6  C
A1 C1 4 2 0 10 1 209 222  C
A1 C4  C
A1 C1 1 2 0 12  C
A1 C1 1 2 0 13  C
A1 C1 1 2 0 11  C
A1 C1 4 2 0 10 1 209 217  C
A1 C1 1 2 0 14  C
```

4.4. Simulação 3

4.4.1. Caderno da Tarefas

4.4.1.1 1o. Nível

O sistema é composto por uma câmera de vídeo localizada no centro de um pentágono. Em cada um dos vértices do pentágono existe um entrevistado sentado.

O entrevistador ficará em uma posição externa e não será focalizado pela câmera. O operador de câmera a comandará através de um sistema composto por cinco botões correspondentes às posições que deseja focalizar e os demais comandos de focalização.

Ao pressionar qualquer um dos botões a câmera se deslocará para o entrevistado pelo menor trajeto possível, aguardando a nova solicitação de movimentação.

4.4.1.2. 2o. Nível

A câmera está fixa em um suporte giratório com a lente orientada para a mesma direção em que existe um indicador metálico de direção no suporte.

Existem cinco sensores indutivos fixos ao solo posicionados na direção dos vértices.

O operador de câmera optará por um dos cinco botões de posicionamento da câmera para direcioná-la para o vértice desejado. O sistema optará pelo sistema de rotação, horário ou anti-horário, para visualizar o entrevistado em menor tempo.

Os botões permanecem retidos pelo sistema até serem soltos automaticamente.

4.4.2. Descrição do Experimento - Definição dos Pontos

Descrição	Ponto	Tipo	BORNE
Girar no sentido horário	GH	Saída	1
Girar no sentido anti-horário	GA	Saída	2
Destino vértice 1	V1	Entrada	1
Destino vértice 2	V2	Entrada	2
Destino vértice 3	V3	Entrada	3
Destino vértice 4	V4	Entrada	4
Destino vértice 5	V5	Entrada	5
Sensor de posição do vértice 1	C1	Entrada	6
Sensor de posição do vértice 2	C2	Entrada	7
Sensor de posição do vértice 3	C3	Entrada	8
Sensor de posição do vértice 4	C4	Entrada	9
Sensor de posição do vértice 5	C5	Entrada	10

4.4.3. Esquema Físico

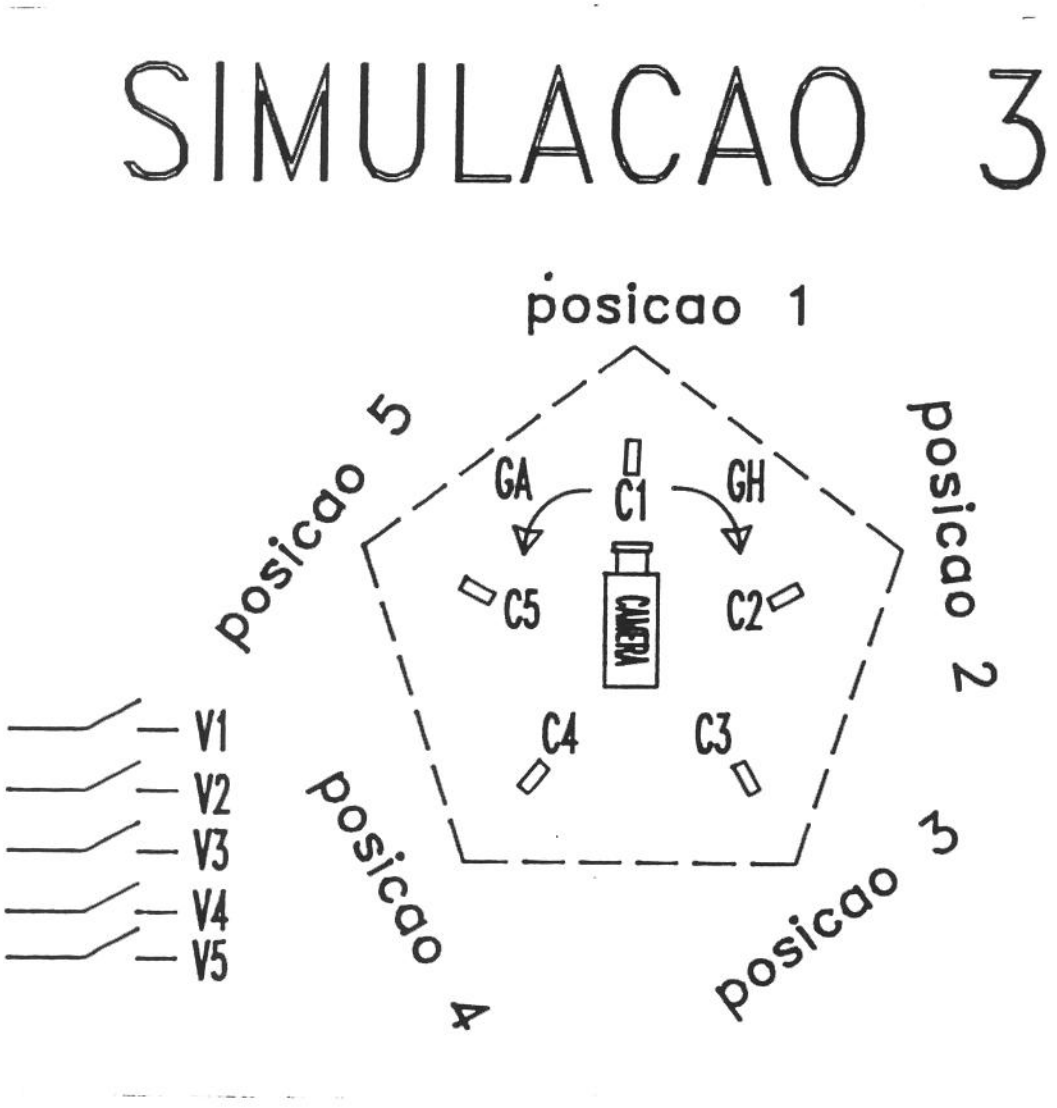

fig 4.5 - Simulação 3 - Esquema Físico

4.4.4. GRAFCET

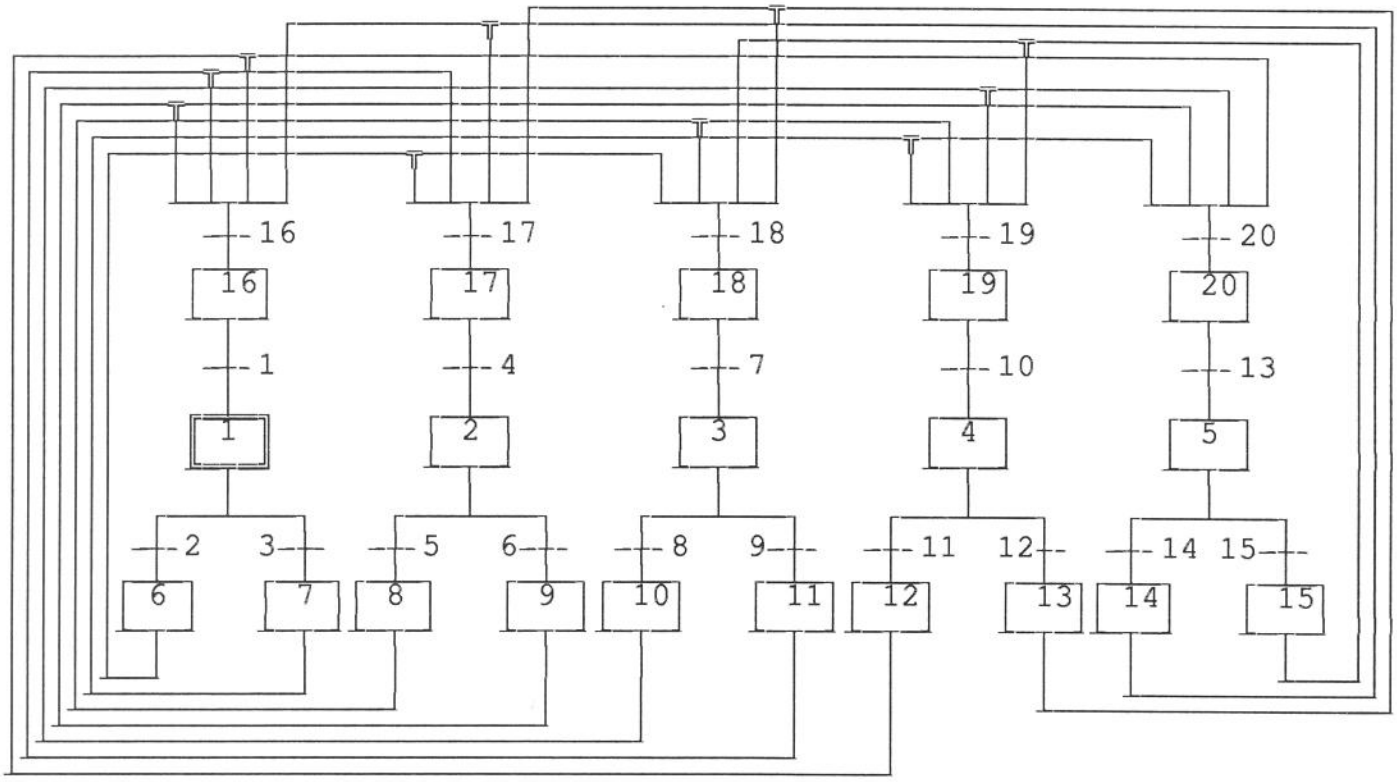

fig 4.6 - Simulação 3 - GRAFCET

TRANSIÇÃO	RECEPTIVIDADE	ETAPA	AÇÃO
1	C1	1	$\overline{GH}$ $\overline{GA}$
2	V2 + V3	2	$\overline{GH}$ $\overline{GA}$
3	V4 + V5	3	$\overline{GH}$ $\overline{GA}$
4	C2	4	$\overline{GH}$ $\overline{GA}$
5	V3 + V4	5	$\overline{GH}$ $\overline{GA}$
6	V1 + V5	6	GH
7	C3	7	GA
8	V4 + V5	8	GH
9	V1 + V2	9	GA
10	C4	10	GH
11	V5 + V1	11	GA
12	V2 + V3	12	GH
13	C5	13	GA
14	V1 + V2	14	GH
15	V3 + V4	15	GA
16	V1	16	$\overline{V1}$
17	V2	17	$\overline{V2}$
18	V3	18	$\overline{V3}$
19	V4	19	$\overline{V4}$
20	V5	20	$\overline{V5}$

4.4.5. Descrição do Simulação para o SIMGRAF

Número de Etapas	20
Número de Transições	20
Número de Entradas	10
Número de Saídas	2
Número máximo de variáveis por subgrupo	2
Número máximo de subgrupo por transição	1
Número máximo de bornera	20
Dezena máxima do número de testes	5
Número máximo de ações por etapa	2

A seguir é descrito o experimento, seguindo a sintaxe criada para a descrição do objeto a ser simulado pelo SIMGRAF. Esta sintaxe é descrita no Anexo B.

```
Vetor de parametrização do SIMGRAF
E1
A20 20 10 2 2 1 20 5 2 A

Vetor de entradas inicialmete ativas
V1
A1 0 0 0 0 0 0 0 0 0 A

Vetor de saídas inicialmente ativas
V2
A0 0 A

Vetor de etapas inicialmente ativas
V3
A1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 A
```

Matriz de ligações entre etapas ⇒ transições

```
      1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0

M1
     A0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 A   1
     A0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 A   2
     A0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 0 0 0 A   3
     A0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 A   4
     A0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 A   5
     A0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 A   6
     A0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 A   7
     A0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 A   8
     A0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 A   9
     A0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 A   10
     A0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 A   11
     A0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 A   12
     A0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 A   13
     A0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 A   14
     A0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 A   15
     A1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 A   16
     A0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 A   17
     A0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 A   18
     A0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 A   19
     A0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 A   20
```

Matriz de ligações entre transições ⇒ etapas

	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0		
M2																						
A1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	A	1
A0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	A	2
A0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	A	3
A0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	A	4
A0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	A	5
A0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	A	6
A0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	A	7
A0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	A	8
A0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	A	9
A0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	A	10
A0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	A	11
A0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	A	12
A0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	A	13
A0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	A	14
A0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	A	15
A0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	A	16
A0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	A	17
A0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	A	18
A0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	A	19
A0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	A	20

```
Matriz tridimensional de ações
     M3
     A2 C101 2 0 0  C  1
        C102 2 0 0  C
     A2 C101 2 0 0  C  2
        C102 2 0 0  C
     A2 C101 2 0 0  C  3
        C102 2 0 0  C
     A2 C101 2 0 0  C  4
        C102 2 0 0  C
     A2 C101 2 0 0  C  5
        C102 2 0 0  C
     A1 C101 1 0 0  C  6
     A1 C102 1 0 0  C  7
     A1 C101 1 0 0  C  8
     A1 C102 1 0 0  C  9
     A1 C101 1 0 0  C  10
     A1 C102 1 0 0  C  11
     A1 C101 1 0 0  C  12
     A1 C102 1 0 0  C  13
     A1 C101 1 0 0  C  14
     A1 C102 1 0 0  C  15
     A1 C1  2 0 0   C     16
     A1 C2  2 0 0   C     17
     A1 C3  2 0 0   C     18
     A1 C4  2 0 0   C     19
     A1 C5  2 0 0   C     20
```

Matriz tridimensional de receptividades

M4

A1 C1 1 2 0 6 C	1
A1 C2 2 2 0 2 3 C	2
A1 C2 2 2 0 4 5 C	3
A1 C1 1 2 0 7 C	4
A1 C2 2 2 0 3 4 C	5
A1 C2 2 2 0 1 5 C	6
A1 C1 1 2 0 8 C	7
A1 C2 2 2 0 4 5 C	8
A1 C2 2 2 0 1 2 C	9
A1 C1 1 2 0 9 C	10
A1 C2 2 2 0 5 1 C	11
A1 C2 2 2 0 2 3 C	12
A1 C1 1 2 0 10 C	13
A1 C2 2 2 0 1 2 C	14
A1 C2 2 2 0 3 4 C	15
A1 C1 1 2 0 1 C	16
A1 C1 1 2 0 2 C	17
A1 C1 1 2 0 3 C	18
A1 C1 1 2 0 4 C	19
A1 C1 1 2 0 5 C	20

Capítulo 5

Conclusões e Perspectivas

O objetivo principal deste trabalho, criar um sistema capaz de validar GRAFCET's, foi alcançado. O sistema está apto a ser utilizado em outros projetos de automação, tal como o projeto de uma plataforma para monitoramento, simulação e controle de sistemas automatizados de produção que está em início de desenvolvimento [18].

A partir do trabalho desenvolvido pode ser dado andamento ao objetivo de ser criado o sistema menos custoso de controladores lógicos programáveis com hardware mais simples, visto que a plataforma utilizada para o trabalho foi um IBM-PC e compatível. Pode também ser prosseguido o ensino e aprendizado da metodologia do GRAFCET, a qual está em fase de inicialização nas indústrias brasileiras.

Algumas melhorias podem ser realizadas sobre o trabalho desenvolvido, tais como:

1o. implementação da descrição do objeto a ser simulado e validado seja realizada utilizando um editor gráfico, dispensando a tarefa de descrever o GRAFCET através da sintaxe descrita no Anexo B;

2o. criação de um analisador de sintaxe;

3o. implementação de classes que possam verificar re-

dundâncias desnecessárias tanto a nível de etapas e transições como de ações e receptividades;

4o. implementação de classes capazes de identificar etapas e transições iguais porém estando localizadas em posições diferentes do GRAFCET, tornando viável a criação de sub-programas conforme ressaltado no capítulo 2;

5o. criação de bibliotecas de GRAFCET's simulados.

ANEXO A

REDES DE PETRI (RDP)

1. Definições

A RdP é definida pela 4-upla:

$$R = \langle P, T; Pré, Pós \rangle$$

onde:

P: é o conjunto de etapas (lugares), sendo p_i a etapa de índice i e m a ordem de P;

T: é o conjunto de transições, sendo t_j a transição de índice j e n a ordem de T;

Pré: é a aplicação de P x T, denominada Incidência Anterior, onde cada elemento será denotado por $pré_{ij}$;

Pós: é a aplicação de P x T, denominada Incidência Posterior, onde cada elemento será denotado $pós_{ij}$.

Cada etapa pode conter um número qualquer de marcações (senhas).

2. Notações

Podemos representar uma RdP de duas formas diferentes, a gráfica e a matricial.

2.1. MATRICIAL

A RdP pode ser representada de forma matricial por 2 matrizes (m x n). Uma matriz será utilizada para representar a aplicação Pré e a segunda para a aplicação Pós, onde as linhas estão associadas as etapas e as colunas associadas as transições.

Os elementos serão inteiros positivos e representarão o número de marcações necessárias (Pré) e resultantes (Pós).

O estado atual da RdP será representado pelo vetor M (nx1), onde cada elemento m_i representa o número de marcações da etapa i.

Pode-se representar as entradas das etapas através da notação $.p_i$ e das transações por $.t_j$. As representações das saídas serão $p_i.$ e $t_j.$ para etapa e transição, respectivamente:

$.p_i$ serão os elementos não nulos da linha p_i da matriz Pós;
$p_i.$ serão os elementos não nulos da linha p_i da matriz Pré;
$.t_j$ serão os elementos não nulos da coluna t_j da matriz Pré;
$t_j.$ serão os elementos não nulos da coluna t_j da matriz Pós.

Esta forma não permite uma representação completa de uma RdP já que ela não indica o número de marcações necessárias para cada entrada ou saída das transições ou das etapas.

2.2. GRÁFICA

Pode-se representar uma RdP de forma gráfica onde as etapas serão indicadas por círculos e as transições serão indicadas por retângulos.

Todos elementos, não nulos, da matriz Pré serão representados por arcos que unirão os elementos $p_i t_j$ .Os elementos da matriz Pós representarão os arcos que unirão os elementos $t_j p_i$.

Ao lado de cada transição será indicado o número de marcações necessárias, para cada um dos arcos, para a validação da mesma bem como o número de marcações que esta validação proporcionará para cada etapa a ela ligada.

3. DINÂMICA

Uma transição só será válida quando todas as suas etapas de entrada possuírem um número maior ou igual ao número de marcações necessárias à validação do arco que as une, isto é, a pré-condição da transição.

Usando-se $Pré(t_i)$ para indicar o vetor coluna i da matriz Pré, uma transição só será válida se, dado um conjunto de marcas **M** :

$$M \geq Pré(t_i)$$

isto é,

$$m_i \geq pré_{ij}$$

Uma vez satisfeitas as pré-condições, a transição será válida e as marcações necessárias a sua validação serão retiradas das etapas de entrada e serão acrescentadas às etapas de saída o número de marcações correspondentes à cada arco de saída, isto é, à pós-condição da transição.

Em conseqüência das pós-condições e pré-condições, o número de marcação em cada transição pode ser não conservativo.

Portanto, sendo M' o novo estado da Rede de Petri, teremos:

$$M' = M - Pré(t_j) + Pós(t_j)$$

Desta forma define-se a MATRIZ de INCIDÊNCIA (C), como sendo uma aplicação de P x T dada por:

$$C(p,t) = Pós(p,t) - Pré(p,t) = M' - M$$

Como as RdP's podem possuir uma etapa com várias transições de saída e uma transição com várias etapas de entrada, podem ocorrer dois tipos de conflitos em decorrência do número de marcações necessárias e disponíveis para a validação de transição:

conflito estrutural: quando duas (t1 e t2) transições possuem ao menos uma etapa de entrada em comum, assim

$$.t1 \cap .t2 \neq 0 \quad \text{ou} \quad \langle (Pré(t1))', Pré(t2) \rangle \neq 0$$

onde : < , > é produto escalar;

conflito efetivo: quando as marcações disponíveis em uma etapa num instante t, não permitem que as transições a esta etapa ligadas (t1 e t2) sejam simultaneamente validadas:

$$M < Pré(t1) + Pré(t2)$$

Denominaremos Rede de Petri Pura como sendo a RdP onde não existe nenhum arco ligando a transição (t_j) à etapa (p_i) se houver o arco ligando a etapa (p_i) à transição (t_j).

As Rdp puras possuem as seguintes propriedades que permitem simplificações:

a) independência entre as entradas e as saídas de marcações;

b) a matriz de incidência (C) torna-se suficiente para descrever a rede, pois:

$$Pré(p,t) = \max(0,-C(p,t))$$

$$Pós(p,t) = \max(0,C(p,t))$$

c) uma transição é validável se :

$$M + C(t) \geq 0$$

e sua validação acarreta :

$$M' = M + C(t)$$

Desta forma, pode-se mostrar que todas as RdP impuras (não puras) podem ser transformadas em RdP puras desdobrando se todas as transições e agrupando se as etapas suplementares de maneira a assegurar a seqüencialidade de validação de falsas transições assim introduzidas.

4. SEQÜÊNCIA DE VALIDAÇÕES

Dada uma seqüência de validações de transições, t1,t2,...,tp pode-se, por comutatividade, representar o novo vetor de marcações M_{p+1} por :

$$M_{p+1} = M1 + C(t1) + C(t2) + \ldots + C(tp)$$

onde S é a seqüência de transições válidas assim

$$S=\{t1,t2,...,tp\}$$

Pode-se denotar a seqüência S como o vetor (n x 1) onde s_j é o número de vezes que a transição j será validada, sendo denominado vetor característico.

Tem-se portanto que a Equação Fundamental das RdP's :

$$M' = M + C.S$$

que torna possível a análise de RdP por técnicas de álgebra linear, porém a relação

$$M + C.S \geq 0$$

não garante que seqüência S seja validável tendo-se a marcação inicial M.

Para tornar isto verdadeiro é necessária a verificação da seqüência passo a passo.

5. ANÁLISE DE RdP'S

Uma vez feita a modelagem do sistema através de uma RdP será necessária a análise a fim de evitar funcionamentos incorretos em determinadas circunstâncias. Isto poderá ser bastante trabalhoso dada a explosão combinatorial de estados que as possíveis validações de transições conduzem.

Desta forma existem certas propriedades que permitirão garantir alguns aspectos de funcionamento incorreto. São elas:

1o. Ausência de bloqueio infinito;

2o. Quantidade máxima de marcas que podem haver em uma etapa, visto que elas traduzem freqüentemente situações físicas reais;

3o. Comportamento sobre evoluções infinitas, já que os sistemas devem funcionar permanentemente.

6. ESTUDO SOBRE RdP

Até o momento introduziu-se a teoria necessária [09], [04] para construir e analisar as RdP's, porém torna-se necessário agora estabelecer métodos para estudos sobre a RdP, por exemplo, simular o seu comportamento.

Para isto encontram-se duas dificuldades básicas:

Definição da marcação inicial, e em conseqüência, do "plano de simulação";

Explosão combinatorial, já citada anteriormente.

Assim, para tornar mais simples os estudos, pode-se utilizar métodos de verificações de propriedades que se subdividem em três grandes grupos:

a. Utilização de álgebra linear, operando-se com matrizes e vetores;

b. Redução da rede;

c. Teoria de Grafos.

ANEXO B

DESCRIÇÃO DA SINTAXE DO ARQUIVO DESCREVENDO UM GRAFCET

Caso a opção de descrição do objeto a ser simulado pelo SIMGRAF via arquivo seja escolhida é necessário que o arquivo que conterá os dados descrevendo o GRAFCET tenha a sua forma obedecendo às 16 regras de sintaxe.

As simulações realizadas no capítulo 4 para a validação do SIMGRAF obedecem às regras de sintaxe descrita abaixo, podendo ser utilizadas como exemplo para apresentar a sintaxe.

Na sintaxe serão utilizados os valores 0 ou 1, para a descrição da existência ou não do referido sinal ou arco de ligação.

Os caracteres indicados neste texto referem-se aos valores da tabela ASCII, decimal.

As regras de sintaxe são as seguintes:

Sintaxe 1. Será considerado início da descrição do objeto a ser simulado, isto é, dimensionamento do objeto quando for encontrado os caracteres 69 e 49, consecutivamente;

Sintaxe 2. Ao ser encontrado o caracter 65, deverão ser fornecidos os números de etapas, transições, entradas, saídas, máximo número de variáveis por subgrupo, máximo número de subgrupos por receptividade, máximo valor entre etapas, transições, entradas e saídas, máximo número de dezenas de testes que a simulação pode atingir e finalizando com o número máximo de ações por etapa.

Os números deverão ser separados por espaço, devem estar na mesma

linha, não deve haver espaço entre o caracter 65 e o número de etapas e finalmente deve haver espaço entre o máximo número de ações por etapa e o caracter 65 que finalizará o vetor de dimensionamento E1.

Sintaxe 3. Será considerado início dos dados de descrição do objeto ao ser encontrado os caracteres 86 e 49, consecutivamente;

Sintaxe 4. Ao ser encontrado o caracter 65, será inicializada a leitura das entradas inicialmente ativas. O vetor será finalizado pelo caracter 65 e deverão haver tantos 0 ou 1 quanto forem o número de entradas descrito pelo vetor de dimensionamento do objeto.

O valor 0 indicará que a entrada está inicialmente desligada e o valor 1 indicará o oposto.

Os valores 0 ou 1 deverão ser separados por um espaço.

Após a última entrada ser descrita deverá haver um espaço e o caracter 65.

Sintaxe 5. Será considerado início do vetor de saídas inicialmente ativas ao ser encontrado o caracter 86 e 50, consecutivamente;

Sintaxe 6. Ao ser encontrado o caracter 65, será inicializada a leitura das saídas inicialmente ativas. O vetor será finalizado pelo caracter 65 e deverão haver tantos 0 ou 1 quanto forem o número de saídas descrito pelo vetor de dimensionamento do objeto.

O valor 0 indicará que a saída está inicialmente desligada e o valor 1 indicará o oposto.

Os valores 0 ou 1 deverão ser separados por um espaço.

Após a última saída a ser descrita deverá haver um espaço e o caracter 65.

Sintaxe 7. Será considerado início do vetor de etapas inicialmente ativas ao ser encontrado o caracter 86 e 51, consecutivamente;

Sintaxe 8. Ao ser encontrado o caracter 65, será inicializada a leitura das etapas inicialmente ativas. O vetor será finalizado pelo caracter 65 e deverão haver tantos 0 ou 1 quanto forem o número de etapas descrito pelo vetor de dimensionamento do objeto.

O valor 0 indicará que a etapa está inicialmente ativa e o valor 1 indicará o oposto.

Os valores 0 ou 1 deverão ser separados por um espaço.

Após a última etapa ser descrita deverá haver um espaço e o caracter 65.

Sintaxe 9. Será considerado início da matriz dos arcos de ligação entre as etapas (linhas da matriz) e as transições (colunas das ações) ao ser encontrado o caracter 77 e 49, consecutivamente;

Sintaxe 10. Ao ser encontrado o caracter 65, será inicializada a leitura dos arcos de ligação entre a etapa, representada pela linha, e a transição, coluna. O vetor será finalizado pelo caracter 65.

O valor 0 indicará a inexistência do arco de ligação e o valor 1 indicará o oposto.

Os valores 0 ou 1 deverão ser separados por um espaço.

Após a última transição da referida etapa ser descrita deverá haver um espaço e o caracter 65.

Sintaxe 11. Será considerado início da matriz dos arcos de ligação entre as transições (linhas da matriz) e as etapas (colunas das ações) ao ser encontrado o caracter 77 e 50, consecutivamente;

Sintaxe 12. A representação de existência ou não dos arcos de ligação será semelhante ao utilizado para a matriz de ligações entre etapas e transições;

Sintaxe 13. Será considerado início da matriz tridimensional de ações das etapas ao ser encontrado o caracter 77 e 51, consecutivamente;

Sintaxe 14.A matriz $M3_{ixjxk}$ representará as k ações das i etapas que atuarão sobre o borne descrito em j = 1 e ativarão ou desativarão conforme descrito em j = 2

Ao ser encontrado o caracter 65, será considerado o início da descrição das ações da etapa. O próximo caracter a ser encontrado indicará o número de ações desta etapa.

Após o número de ações deverá existir um caracter 65 e então iniciará a leitura dos seguintes valores:

a. Número do borne sobre o qual será executada a ação;
b. Se a ação será de ativação ou desativação;
c. Se a ação for condicional (valor 0) ou não (valor 1 para condicional afirmativa ou valor 2 para condiconal negativa);
d. Se a ação for condicional a uma entrada (valores entre 1 e 99, de acordo com o número da entrada) ou a uma saída (valores entre 101 à 199) ou de etapa (valores entre 200 e 299);

Após o quarto valor desta linha do arquivo de dados deverão existir dois caracteres espaço e o caracter 67 finalizando esta linha de dados. A próxima linha será inicializada pelo caracter 67 indicando a próxima ação desta etapa até que sejam lidas todas ações desta etapa, passando à próxima etapa.

Sintaxe 15. Será considerado início da matriz tridimensional de receptividades das transições ao ser encontrado o caracter 77 e 52, consecutivamente;

Sintaxe 16. A matriz $M4_{ixjxk}$ representará os k subgrupos das i transições que conterão as j informações seguintes:

j = 1 → qual é o tipo de operação lógica do subgrupo (1 é a operação AND, 2 é a OR, 3 é a NOT, 4 representará uma receptividade sempre válida, 5 é um subgrupo condicional;

j = 2 → quantas variáveis existem neste subgrupo (valor entre 1 e 5);

j = 3 → se este é o último subgrupo desta receptividade;

j = 4 → qual é a operação lógica que une a receptividade ao próximo subgrupo (1 representa a operação AND e 2 a operação OR);

j = 5 até 9 indicam os bornes das variáveis, valores entre 1 e 299.

Ao ser encontrado o caracter 65, será considerado o início da descrição das receptividades da transição. O próximo caracter a ser encontrado indicará o número de subgrupos desta transição.

Após o número de ações deverá existir um caracter 67 e então iniciará a leitura dos valores representando os valores de j de 1 até 9, caso existam.

Após o último caracter do subgrupo desta linha do arquivo de dados deverão existir dois caracteres espaço e o caracter 67 finalizando a linha. A próxima linha será inicializada pelo caracter 67 e inicializado próximo subgrupo desta receptividade até que sejam lidos todos subgrupos desta transição, passando a próxima transição.

BIBLIOGRAFIA

1. Referências Bibliográficas

[01] Blanchard, M.; "Comprendre Maitriser et Appliquer Le GRAFCET"

[02] Bossy, J. C.; Brard, P.; Faugère, P.;Merlaud, C.; "Le GRAFCET sa Pratique et ses Applications", Educalivre, Paris, 1979

[03] Soriano, T.; "Definition des Representations Textuelle et Graphique dún Modele Oriente Object de la Commande en Genie Automatique", Tese de Doutorado no Institut Superieur des Materiaux et de la Construction Mecanique (ISMCM), 1987.

[04] Frachet, J. P.; "Introduction aux Reseaux de Petri", Laboratoire de Genie Automatique, ISMCM, Paris

[05] Cox, B. J.; "Object Oriented Programming - An Evolutionary Approach", Addison Wesley, 1987

[06] Shammas, N.; "Programação Orientada para Objeto com Turbo Pascal 5.5", McGraw-Hill, 1991

[07] Pascoe, G. A.; "Elements of Object Oriented Programming", Byte, August 1986

[08] Mayer, R. C.; "Software na Linha de Montagem", Exame, Junho 1992

[09] Brams, G. W.; "Reseaux de Petri: Theorie et Pratique", Masson, 1983

[10] Bertrand, P.; Bouteille, D.; Collot, R.; Garnier, J. C.; Hénau, J. F.; "Les Automatismes Électropneumatiques et Pneumatiques", Lúsine, Paris, 1985

[11] Bouteille, D.; Bouteille, N.; Chantreuil, S.; Collot, R; Frachet, J. P.; Le Gras, H.; Merlaud, C.; Selosse, J.; Sfar, A.; "Les Automatismes Programmables", Toulosse, 1987

[17] SDC Engenharia, Sistemas e Software; "SDC2817 - Interface de Entrada e Saída Digital"

[18] Rosário, J. M.; projeto de pesquisa "Implementação de Plataforma para Monitoramento, Simulação e Controle de Sistemas Automatizados de Produção", Laboratório de Automação e Robótica - FEM - UNICAMP, 1994

[20] ADEPA/AFCET; "Le GRAFCET", Cepadues édition, 1992

2. Bibliografia Geral

[12] Briard, M.; Brendle, P.; "Production Automatisée", ISMCM, 1985

[13] Pun, L.; documentação de suporte ao curso de "Systemes Industriels D'Intelligence Artificielle", ISMCM, Paris

[14] Malucelli, V. V.; Hautberge, B.; Fonseca, K. V. O.; Neves Jr., F.; "Ambiente de Suporte a Sistemas de Controle de Processos Industriais", Centro Federal de Educação Tecnológica do Paraná

[15] Filgueiras, L. V. L.; "Integração Homem-Máquina em Processos Industriais", Escola Politécnica, USP

[16] Wood, S.; "Turbo Pascal - Guia do Usuário", McGraw Hill, 1987

[19] Rosário, J. M.; Khair, M. El.; "Generation de la commande des systèmes mécaniques polyarticulés", ISMCM, Paris, 1989

[21] Bittar, R. C. S. M.; "A utilização do GRAFCET como ferramenta na Automação Industrial", UNICAMP, 1993