ESTE EXEMPLAR CORRESPONDE A REDAÇÃO FINAL DA TESE DEFENDIDA POR Cintia Kimie Aihara E APROVADA PELA COMISSÃO JULGADORA EM 06/04/2000

João Maurício Rosário
ORIENTADOR

UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA MECÂNICA

Projeto e Implantação de Plataforma Didática Aplicada ao Ensino e Pesquisa em Automação

Autor: **Cintia Kimie Aihara**

Orientador: **João Maurício Rosário**

36/00

UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA MECÂNICA
DEPARTAMENTO DE PROJETOS MECÂNICOS

Projeto e Implantação de Plataforma Didática Aplicada ao Ensino e Pesquisa em Automação

Autor: **Cintia Kimie Aihara**
Orientador: **João Maurício Rosário**

Curso: Engenharia Mecânica
Área de Concentração: Projeto Mecânico

Dissertação de mestrado apresentada à comissão de Pós Graduação da Faculdade de Engenharia Mecânica, como requisito para a obtenção do título de Mestre em Engenharia Mecânica.

Campinas, 2000
S.P. - Brasil

FICHA CATALOGRÁFICA ELABORADA PELA
BIBLIOTECA DA ÁREA DE ENGENHARIA - BAE - UNICAMP

Ai37p

Aihara, Cintia Kimie
Projeto e implantação de plataforma didática para ensino e pesquisa de sistemas automatizados integrados / Cintia Kimie Aihara.--Campinas, SP: [s.n.], 2000.

Orientador: João Maurício Rosário.
Dissertação (mestrado) - Universidade Estadual de Campinas, Faculdade de Engenharia Mecânica.

1. Automação. 2. Controladores programáveis. 3. Robótica. I. Rosário, João Maurício. II. Universidade Estadual de Campinas. Faculdade de Engenharia Mecânica. III. Título.

UNIVERSIDADE ESTADUAL DE CAMPINAS
FACULDADE DE ENGENHARIA MECÂNICA
DEPARTAMENTO DE PROJETOS MECÂNICOS

DISERTAÇÃO DE MESTRADO

Projeto e Implantação de Plataforma Didática Aplicada ao Ensino e Pesquisa em Automação

Autor: **Cintia Kimie Aihara**
Orientador: **João Maurício Rosário**

Prof. Dr. João Maurício Rosário, Orientador
FEM / UNICAMP

Prof. Dr. João Carlos Mendes Carvalho
Universidade Federal de Uberlândia

Prof. Dr. Marcius Fabius Henriques de Carvalho
FEM / UNICAMP

Campinas, 06 de abril de 2000

Dedicatória:

Dedico este trabalho aos meus pais e irmãos.

Agradecimentos

Este trabalho não poderia ser terminado sem a ajuda de diversas pessoas às quais presto minha homenagem:

Aos meus pais e irmãos, pelo incentivo em todos os momentos da minha vida.

Ao meu orientador, prof. Dr. João Maurício Rosário, pela forma como conduziu este trabalho e principalmente pela confiança que depositou em mim.

Aos amigos Almiro, Adelson, Maurício, Gastão, Hamilton, Rafael, Marcos e a todos os demais que ajudaram de forma direta e indireta na conclusão deste trabalho.

" Educai as crianças e não será preciso punir os homens "

Pitágoras

Resumo

AIHARA, Cintia Kimie, *Projeto e Implantação de Plataforma Didática para Ensino e Pesquisa de Sistemas Integrados Automatizados*, Campinas, Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, 2000. 104 p. Dissertação (Mestrado)

Os avanços tecnológicos dos últimos tempos implicou em um aumento da competitividade industrial e a integração de sistemas automatizados de produção, o que passou a ser o sinônimo de flexibilidade para a obtenção de produtos competitivos e com qualidade. Dentro deste contexto, este trabalho aborda os principais elementos que descrevem a Parte Operativa e a Parte Comando de um Sistema Automatizado Industrial, através do desenvolvimento de metodologias aplicadas ao ensino e formação em automação. Esta metodologia foi validada e implementada no Laboratório de Automação Integrada e Robótica, da UNICAMP, através de uma plataforma didática constituída de diferentes elementos automatizados (robôs, mesa rotativa, CLP's, atuadores e sensores). Nesta plataforma, foram implementados diferentes módulos de validação e aprendizagem.

Palavras Chaves

- Automação, Controlador Programável, Robótica

Abstract

AIHARA, Cintia Kimie, *Design and Implantation of Didactic Platform for Education and Research of Automated Integrated Systems*, Campinas,: Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, 2000,104 p. Dissertação (Mestrado)

Recent technological advances has been the responsible for an increase of industrial competitiveness and the integration of automatized systems of production, which became synonymous of flexibility for achieving competitive and high quality products. In this context, this work approach the main elements that describe the Operative Part and the Command Part of an Automated Industrial System, through the development of methodologies applied to education and formation in industrial automation. This methodology was validated and implemented in the Laboratory of Integrated Automation and Robotic, at the UNICAMP campus, through a consisting didactic platform with different automated elements (robots, rotating table, PLC's, actuators and sensors). In this platform, different modules of validation and learning have been implemented.

Key Words

Automation, Programmable Controller, Robotic

Índice

Lista de Figuras

Lista de Tabelas

Nomenclatura

CAD	Projeto Assistido por Computador
CAM	Manufatura Auxiliada por Computador
CC	Corrente Contínua
CIM	Manufatura Integrada por Computador
CLP	Controlador Lógico Programável
CN	Comando Numérico
CNC	Comando Numérico Computadorizado
CP	Controlador Programável
EAD	Ensino à Distância
FMS	Sistema de Manufatura Flexível
IEC	International Eletrotechnical Committe
PC	Parte Comando
PC	Microcomputador
PO	Parte Operativa
SAP	Sistema automatizado de Produção
SFC	Sequential Flow Chart

Capítulo 1

Introdução

Desde os tempos mais remotos, o homem vem tentando fazer com que utensílios / ferramentas, substituam-no no trabalho, sendo o seu maior sonho criar um autômato que realize todas as suas funções operárias.

No fim da Idade Média, que é um período em que vemos grandes avanços nas teorias Mecânica, Física e Química, como a determinação das leis de Newton e Lavosier, é também a época em que se deu início ao pensamento da máquina para substituir o homem.

O homem, por sua vez, sabendo das suas limitações em suas capacidades, tem criado artifícios que lhes permitam amplificar os seus poderes. Exemplos tais como: a criação de vestuário e habitações para sua proteção e a invenção de máquinas que permitam ampliar o seu poder de atuação sobre a natureza, uma vez que sua força muscular é escassa.

Nos últimos anos, com a globalização, as indústrias passaram por grandes modificações, com o intuito de se tornarem mais competitivas. Foi necessária a modernização de seus parques industriais, visando a competitividade de seus produtos, através do aumento da qualidade, redução de custos e preços acessíveis.

Com o intuito de alcançar tal objetivo, foram criados um conjunto de técnicas, dando-se o nome de AUTOMAÇÃO.

Este trabalho está inserido neste contexto, tendo como objetivo desenvolver o projeto e a implantação de uma plataforma didática, visando procedimentos metodológicos para a formação e a pesquisa na área de Integração de Sistemas Automatizados Industrial.

Neste capítulo será apresentada a metodologia desenvolvida neste trabalho e um breve histórico da Automação Industrial.

1.1 Automação Industrial – Histórico

A introdução das primeiras formas de automação se deu nas industrias de processo, através do desenvolvimento de equipamentos de controle e de medição elétrica e pneumática. Porém, a palavra automação ganhou relevância com o surgimento da máquina de comando numérico em 1949/50. Criada com a capacidade para realizar certas operações, previamente programadas, sem a intervenção direta de um operador, esta máquina abriu perspectivas para mudanças profundas na produção industrial.

As primeiras máquinas automáticas eram constituídas por sistemas de comando formados por circuitos com válvulas eletrônicas a vácuo e outros componentes, ligados por fios elétricos. A evolução tecnológica de materiais e componentes agilizou o avanço das máquinas automáticas de controle numérico. Estes componentes e válvulas foram substituídos pelo transistor e os fios por placas de circuitos integrados. Entretanto, a ligação do sistema de comando continuava sendo feita de forma rígida, através de fiação com a máquina. O próximo passo foi a substituição de todo esse sistema pelo computador, chegando-se ao CNC, versátil, sofisticado e revolucionário nas suas aplicações.

O Comando Numérico Computadorizado (CNC) pode ser definido como o uso do computador para comandar o caminho da ferramenta cortante de uma máquina operatriz, tendo com isto uma alta precisão no produto final e alta repetibilidade com um mesmo programa.

Podendo ainda associar o comando CNC diretamente com o CAD – Projeto Assistido por Computador – permitindo realizar o produto diretamente a partir do projeto.

Estas máquinas, não foram recebidas com entusiasmo, devido principalmente ao alto custo, à fragilidade das primeiras máquinas que exigiam permanente e custosa manutenção, ao desempenho das máquinas universais considerado satisfatório para a pequena e média empresa.

Esta visão e comportamento não durou, tendo em vista a evolução das máquinas CNC, que assumiram características próprias. O seu desempenho incluía possibilidades de mudanças de operações conforme o programa, troca automática de ferramenta e outros acessórios, capacidade de executar tarefas recebidas através de linhas de transmissão e armazenar as informações. A flexibilidade das máquinas e a comunicação estabelecida entre elas criaram um sistema de produção altamente integrado. Embora essas máquinas tenham as mesmas finalidades das máquinas universais, os procedimentos de trabalho da máquina CNC propiciaram ganho de produtividade, por conta da redução de tempo e melhoria da qualidade, suprimindo ou reduzindo trabalhos anteriormente necessários para a preparação e posicionamento da ferramenta e da peça, bem como paradas intermediárias para medições ou comparações.

1.2 Automação Industrial - Conceito

O conceito de automação é constantemente confundido com automatização. O conceito de automatização está ligado a realização de movimentos automáticos, repetitivos e mecânicos, sendo portanto sinônimo de mecanização. E, mecanismo implica em ação cega, sem correção. Já, a Automação possui um conceito de conjunto de técnicas, através das quais se constróem sistemas ativos capazes de atuar com uma eficiência ótima pelo uso de informações recebidas do meio sobre o qual atuam. Com base nas informações recebidas, o sistema calcula a ação corretiva mais apropriada. Ou seja, um sistema de automação comporta-se como o operador humano, utilizando as informações sensoriais. Este pensa e executa a ação mais apropriada. Na automação existe uma auto-adaptação às condições diferentes, de modo que as ações do sistema de maquinismos conduzam a resultados ótimos. A automação está ligada à utilização de sistemas automáticos.

Pode-se definir ainda a automação como sendo um sistema que tende a aumentar a eficiência de um determinado processo.

A Automação Industrial, na maioria das vezes processa-se da seguinte maneira: um computador recebe os sinais provenientes dos vários instrumentos de medidas da fábrica, compara tais medidas com os valores ideais e realiza operações matemáticas com a finalidade de gerar sinais de correção. Tais sinais instruirão os dispositivos de controle acerca da alteração mais apropriada para cada instante, com o intuito de conduzir a uma produção ótima sob um determinado ponto de vista, seja ele qualitativo ou quantitativo.

Com isto, a Automação Industrial pode ser dividida em três classes: a rígida, a flexível e a programável, (Vendrameto, 1994).

- A automação fixa é utilizada quando o volume de produção é elevada, desta forma a linha de produção é composta de diversas máquinas de Comando Numérico CN, chamadas estações de trabalho. Nestas estações são realizadas operações e à medida que são terminadas, as peças são transferidas a outras estações. Desta forma a produção possui uma linha de produção fixa, voltada apenas para a concepção de um determinado tipo de produto.

- A automação flexível é utilizada para um volume de produção médio e é decorrência da junção da mecânica, com o tratamento da informação pela informática e as tecnologias eletrônicas. Nesta forma, a automação aliada à flexibilidade possibilita que sejam fabricados diversos produtos ao mesmo tempo, utilizando-se o mesmo sistema de produção.

- A automação programável é utilizada para um volume de produção relativamente baixo e diversificado. Ou seja, a produção é efetuada em pequenos lotes. Nesta forma de produção, os equipamentos devem ser reprogramados a cada novo lote.

Com tudo isto, a modernização das fábricas, totalmente ou parcialmente automatizadas surgem no mercado, com o intuito de adaptar-se às exigências do mercado tornando-se mais competitivas. Portanto, podemos dizer que a automação no processo produtivo tem a finalidade básica de facilitar estes processos. Isto acarretará na realização de sistemas otimizados capazes de produzir bens com menor custo, maior quantidade, menor tempo e maior qualidade.

Visando essa realidade de mercado, foi implementado no Laboratório de Automação Integrada e Robótica da UNICAMP, um Sistema Automatizado de Produção – SAP – constituído de uma Parte Operativa didática contendo dois manipuladores robóticos, uma mesa rotativa acionado através de um motor C.C., atuadores diversos (bombas e motores elétricos) e sensores de posicionamento diferentes, e uma Parte Comando constituída de um CLP industrial , módulos de controles dos manipuladores robóticos.. A integração dos diferentes elementos, permitiu a mistura automatizada de cores. Neste projeto de pesquisa são fornecidos ferramentas para descrição e implementação desses diferentes elementos.

Com o objetivo de sistematizarmos a estrutura de comando do SAP apresentado, neste trabalho de pesquisa, serão abordados cada elemento separadamente e a integração entre os mesmos, formando com isto ao final, um sistema integrado de produção.

1.1 Objetivos

Este trabalho tem como objetivo principal o projeto e a implantação de uma plataforma didática para aplicação em ensino e formação em automação industrial e robótica, através da concepção de um sistema automatizado de produção, didático e de baixo custo, que contenha elementos que possibilitem uma visão completa de sistemas automatizados industriais.

Para que tal objetivo seja atingido, são apresentados, de forma didática e estruturada, ferramentas e elementos de automação utilizados, sendo apresentado ainda um exemplo de aplicação, utilizando-se a integração desses elementos.

1.2 Delineamento do Trabalho

A dissertação proposta, foi subdividida em cinco capítulos descritos a seguir.

No capítulo 1 é apresentada uma revisão bibliográfica abordando aspectos e conceitos concernentes a Automação Industrial, e as diferentes etapas e objetivos a serem delineados para o desenvolvimento desse trabalho.

No capítulo 2 são descritas as diferentes ferramentas para a descrição e modelagem de sistemas seqüênciais.

No capítulo 3 são apresentados os elementos utilizados em um SAP através de uma plataforma didática para mistura de cores, contendo as ferramentas descritas no capítulo 2.

No capítulo 4 é apresentada a implementação dos diferentes elementos de um Sistema Automatizado de Produção e a integração entre eles, com ênfase na programação desses elementos do SAP desenvolvido, e sua implementação utilizando os conceitos abordados anteriormente.

No capítulo 5 são apresentados as considerações finais e sugestões para trabalhos futuros.

Capítulo 2

Descrição de um Sistema Automatizado de Produção

O conhecimento de sistemas automatizados de produção é de uma diversidade de variáveis independentes, entrelaçadas a múltiplas áreas, com intensidade de influências alteradas conforme o momento, podendo-se citar a economia, a administração geral e da produção, a organização do trabalho; aspectos sociais e psicológicos como o emprego, a organização sindical, o sentimento de perda de conhecimento do trabalhador, as doenças profissionais e o lado tecnológico da automação com equipamentos "inteligentes" como: CAD – Projeto Auxiliado por Computador, CAM – Manufatura Auxiliada por Computador, CN – Comando Numérico, CNC – Comando Numérico Computadorizado, CLP – Controlador Lógico Programável, Robô, FMS – Sistemas de Manufatura Flexível e CIM - Manufatura Integrada por Computador. Estes temas, tratando-se de temas recentes, encontram-se em evolução e participam da elaboração de qualquer projeto para automação em graus de relevância diferentes. Com isto, um sistema industrial moderno, competitivo, que visa produtividade e qualidade precisa ser planejado, para que seja utilizada a tecnologia apropriada, com a integração das partes de forma a proporcionar uma flexibilidade.

A complexidade destes sistemas implica grande dificuldade na definição clara de suas especificações funcionais. Por ser uma área multidisciplinar , a automação industrial envolve uma grande variedade de assuntos de fundamental importância, incluindo o desenvolvimento de redes de comunicação, softwares dedicados, integração e flexibilidade de sistemas.

Verificamos isto no momento da aquisição de equipamentos por pequenas e médias empresas que, com expectativa de retorno a curto prazo e sem a devida adequação do ambiente fabril, têm experiências negativas, adiando e prejudicando a entrada de sistemas automatizados na linha de produção. Porém, o processo de automatização da manufatura ainda é um movimento em implantação e em evolução, exigindo a formalização para os modelos de Sistemas Automatizados.

Segundo a definição do Vocabulário Eletrotécnico Internacional, capítulo 351: Controle Automático – IEC 50(351), um Sistema de Controle pode ser dividido em duas partes interdependentes:

- *Sistema Controlado* – sistema que executa a operação física e;
- *Sistema de Controle* – equipamento que recebe as informações provenientes do operador, do processo a ser controlado, etc., e emite ordens ao Sistema Controlado.

Neste trabalho, estaremos referindo ao Sistema Controlado como Parte Operativa (PO) e o Sistema de Controle como Parte Comando (PC).

2.1 Caderno de Tarefas

O caderno de tarefas de um sistema automatizado de produção é a descrição de seu comportamento em função da evolução das etapas que descrevem o processo a ser automatizado. O mesmo deve apresentar uma descrição clara, precisa e sem ambigüidades, nem omissões, do papel das etapas constituintes do processo a ser automatizado. Com isto, a descrição do sistema deve ser dividida em dois níveis sucessivos e complementares: nível 1 e nível 2.

O **nível 1 – ESPECIFICAÇÕES FUNCIONAIS** - deve descrever o comportamento da parte comando PC em relação a parte operativa PO. As especificações funcionais permitem compreender quais as funções que o automatismo deve realizar, face às diferentes situações que podem surgir. Neste nível pouco importa qual a forma que realizará um determinado movimento, mas é importante conhecer em que circunstância o deslocamento deve ser realizado. É neste nível que aspectos de segurança previstos para o funcionamento devem ser incorporados nas

especificações funcionais, na proporção em que eles não dependam diretamente da tecnologia empregada.

O **nível 2 – ESPECIFICAÇÕES TECNOLÓGICAS E OPERACIONAIS** - deve acrescentar às exigências funcionais um detalhamento das condições de funcionamento dos constituintes do processo a ser automatizado, através de especificações tecnológicas e operacionais. Neste nível deve haver indicações sobre a exata natureza dos transdutores e atuadores, do modo como estes elementos deverão ser inseridos fisicamente no processo que compõe o sistema automatizado e informações acerca do seu meio ambiente de atuação.

Estas considerações são muito importantes para a exploração do processo a automatizar, considerando-se suas repercussões sobre o aspecto econômico, que são freqüentemente esquecidas durante a elaboração do caderno de tarefas, pois são difíceis de serem expressas de maneira quantitativa.

2.2 Linguagens para Controle de Sistemas Discretos

Com a complexidade crescente dos sistema automatizados de produção, o usuário passou a encontrar grandes dificuldades, em uma maneira clara, concisa e não ambígua, nas especificações funcionais associadas a esses sistemas. Esta complexidade tende a aumentar ainda mais, com a utilização de um número elevado de informações de entradas e saídas. Desta forma é necessário descrevermos o sistema através de ferramentas de descrição adequadas.

Para o desenvolvimento de um sistema automatizado de produção, devemos inicialmente descrevê-lo claramente, independentemente de sua estrutura. Esta descrição pode ser global ou específica. Esta linguagem deve ser do ponto de vista do homem, uma linguagem natural que expresse de modo simples a especificação do sistema, e do ponto de vista do dispositivo de controle, uma descrição simples que seja fácil de ser interpretada e executada. Esta é a maior dificuldade desta etapa, pois é necessário que se forneça ao projetista as informações de forma detalhada, independente do nível da descrição.

Uma vez que a linguagem verbal possibilita a interpretações diversas e até ambíguas a representação gráfica passa a ser a forma de mais fácil compreensão. A dificuldade consiste em se encontrar uma representação aceita por todas as partes envolvidas em um projeto, e que não seja diretamente relacionada à tecnologia ou implementação específica. Além disso, algumas vezes pode ser difícil encontrar um síbolo gráfico para cada função representada. Para atender essa necessidade a IEC (International Eletrotechnical Committe), apresentou a norma IEC 1131-3, que consiste na padronização de 5 linguagens de programação para CLP's, definindo os procedimentos de controle e intertravamento de sistemas seqüenciais.

No IEC os padrões para as linguagens para controle de sistemas discretos são definidos de modo flexível com a finalidade de estabelecer as especificações mínimas a serem representadas e as regras para as futuras expansões.

As linguagens padronizadas pela norma IEC 1131 são:

- SFC (Sequential Flow Chart): também conhecido pelo nome GRAFCET, divide o processo em um número definido de passos, separados por transições.

- Diagrama de blocos: linguagem gráfica que permite ao usuário construir procedimentos combinacionais complexos utilizando-se blocos padrões como, AND, OR, NOT, etc.

- Ladder: conhecida linguagem de contatos, bastante popular para representar funções de intertravamento elétrico e para processamento de alarmes

- Texto estruturado: Linguagem de alto nível similar ao Pascal, porém incorporando uma série de conceitos intuitivos ao engenheiro de automação. Seu uso é bastante interessante na implementação de procedimentos complexos, que são difíceis de expressar com linguagens gráficas tais como linguagem de algoritmos de otimização de processo e inteligência artificial.

- Lista de instruções: linguagem de baixo nível baseada em comandos "load", "store", "move", "add", que apresentam alta eficiência em pequenas aplicações, tais como sensores e atuadores inteligentes, ou na otimização de partes de uma aplicação.

2.3 Diagrama Funcional Seqüencial

O IEC definiu através da Norma IEC 60848 – "Preparação de Diagramas Funcionais para sistemas de Controle" - um método de representação que combina símbolos gráficos e declarações textuais, que permite a descrição tanto global como específica, através da relação precisa entre as entradas e saídas, independentemente das propriedades tecnológicas dos componentes usados ou de implementações futuras.

Esta norma teve como origem o GRAFCET, criado a partir de um grupo de pesquisadores e gerentes industriais franceses, envolvidos em complexos sistemas de controle discreto, que em 1975, se reuniram para comparar e avaliar os modelos e métodos para construção de sistemas de controle seqüencial. Através da coleta de suas experiências, em que utilizavam dezessete técnicas diferentes, desde questionários empíricos a modelos teóricos puros derivados da Redes de Petri. Decidiram então construir um modelo customizado, mais fácil que os até então utilizados, e mais adequado aos sistemas complexos e particularmente aos sistemas de manufatura. Após dois anos de vários encontros de estudos e trabalho, propuseram um modelo chamado GRAFCET. O nome derivou-se de *"Graf"*, devido ao fundamento gráfico e, AFCET, *Association Française de Cybernétique Économique et Technique*, a associação científica que suportou o trabalho. E em 1982 foi aceita como uma norma pela associação francesa AFNOR, *Association Française de Normatization.*

Os conceitos do GRAFCET se baseia em um conjunto de definições sobre as quais são estabelecidas suas regras fundamentais em álgebra booleana (verdadeiro ou falso). É um método de descrição, formado basicamente por um conjunto de etapas, transições e a condição associada à transição e ligações orientadas. Estes conceitos permanecem até hoje, sendo que a etapa representa o estado parcial do sistema, uma vez que esta pode estar ativa ou inativa, no qual a ação é realizada. A ação associada é realizada somente se a etapa estiver ativa. A transição indica

a possibilidade de evolução entre as etapas e a cada transição existe uma condição conhecida também por receptividade, representando uma decisão para a mudança de estado do sistema. Isto significa que uma transição seja efetuada é necessário que a etapa anterior à transição esteja ativa e que a condição lógica (booleana) associada à transição seja verdadeira. A ligação orientada, indica o caminho de evolução do GRAFCET.

Com a combinação destes elementos teremos a representação estática do Sistema Automatizado. Aplicando-se as Regras de Evolução, obtemos a visão dinâmica do mesmo. Isto pode ser verificado, no exemplo da figura 2.1, em que um motor inicia desligado. Quando acionado um botão o motor liga e somente após a desativação da botoeira o motor é desligado e assim sucessivamente.

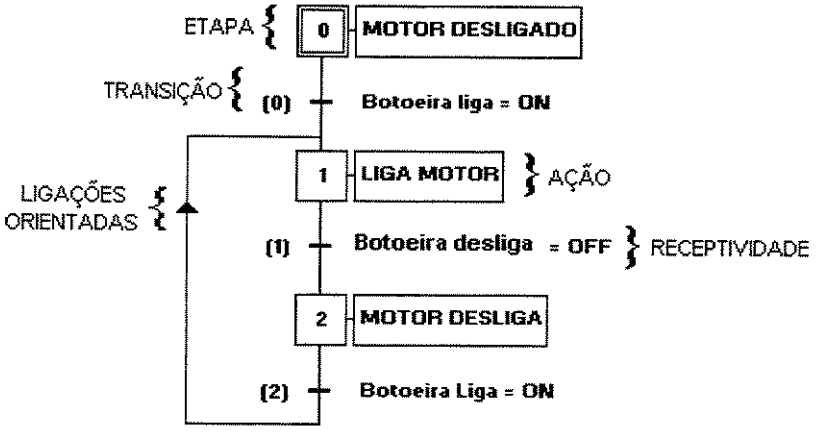


Fig.2.1 - Exemplo de GRAFCET

Este modelo foi testado durante vários anos em empresas privadas e instituições de ensino franceses, revelando ser muito eficiente à representação de pequenos e médios sistemas seqüenciais,

Em 1988, a IEC adotou o GRAFCET como Norma Internacional sob o nome inglês *"Sequencial Function Chart"* – SFC (Diagrama Funcional Seqüencial), com o título *"Preparation of Function Chart for Control Systems"* (Preparação de Diagramas Funcionais para Sistemas de Controle) e referência IEC 848. Com as recentes modificações nas referencias a

Norma IEC, passou a IEC 60848. O método de representação proposto serve como "ferramenta de comunicação" entre as diferentes áreas tecnológicas envolvidas no desenvolvimento e utilização de Sistemas Automatizados.

2.3.1 Etapas

A cada etapa há uma ação associada. Corresponde à condição invariável e definida do sistema descrito. É a situação durante a qual o comportamento de todo ou parte do sistema em relação às suas entradas ou saídas é invariável. A eficiência e precisão de um Diagrama Funcional Seqüencial estão diretamente relacionadas à quantidade de etapas utilizadas para descrever determinado sistema. Portanto, quanto maior o número de etapas possíveis de ser dividido, maior a eficiência da descrição de cada etapa e do diagrama funcional como um todo.

A etapa é representada por um quadrado ou retângulo, referenciada numericamente internamente, como demonstrado na tabela 2.1. A relação entre o tamanho dos lados não é definida, porém recomenda-se a igualdade. Estas referências são dadas de forma arbitrária, não necessitando de utilização seqüencial do mesmo e nem de respeito à ordem numérica – crescente ou decrescente – não sendo permitido somente que etapas distintas tenham a mesma referência.

Uma etapa pode estar ativa ou inativa e o conjunto de etapas ativas em um determinado momento define a situação do sistema ou o estado de automatismo naquele momento. A representação deste estado pode ser dado por valores lógicos "0" ou "1" (inativo ou ativo)de uma variável binária "X". Por exemplo: X5 = 0, ou seja, a etapa 5 está inativa. Para indicar no diagrama a(s) etapa(s) ativa (s) é dada por um ponto (•) localizado na parte inferior dos símbolos correspondente(s).

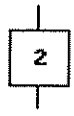
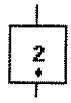
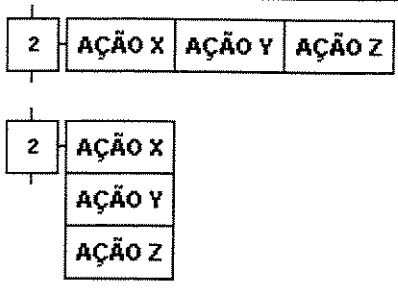
O ponto não pertence à simbologia da etapa, sendo utilizado somente na análise e/ou apresentação do Diagrama.

A etapa inicial é ativada incondicionalmente no início do controle de um sistema e indica a situação inicial do mesmo. Podem existir tantas etapas iniciais, quantas se fizer necessárias,

sendo que todas serão ativadas simultaneamente no início do controle do sistema. A etapa inicial é distinguida através da simbologia que recebe traços duplos.

A cada etapa está associada uma ação, ou seja, a ação a ser efetuada quando ativada tal etapa. Esta ação será executada sempre que, e só quando as respectivas etapas estejam ativas. Enquanto ativadas, as ações podem ser inicializadas, continuadas ou finalizadas. Quando a etapa for desativada, as ações podem ser continuadas ou finalizadas, conforme a definição utilizada. As ações a efetuar quando a etapa estiver ativa, podem ser descritas em notação literal ou simbólica, no interior de retângulos ligados ao lado direito da etapa.

A tabela 2.1 representa as simbologais das etapas:

SIMBOLOGIA	DESCRIÇÃO
	**Etapa Inicial**
	**Etapa**
	**Indicação da Etapa Ativa no SFC**
	**Ação Associada à Etapa**
	**Representação de Várias Ações Associadas a Mesma Etapa**

Tab. 2.1 - Simbologia - Etapas

A representação de mais de uma ação associada à mesma etapa é dada utilizando-se uma das formas apresentadas na tabela 2.1. Embora cada ação esteja em retângulos individuais, a simbologia utilizada não especifica a seqüência entre as ações associadas à etapa.

É importante que a definição da ação associada à etapa seja feita com clareza e sem ambigüidades. Além de definir o comportamento do sistema em determinado momento, deve definir se a ação será mantida ou finalizada após a desativação da etapa. Isto pode ser realizado descrevendo juntamente com a ação, como na figura 2.2.

a. ação iniciada ao ser ativada a etapa 2 e, mantida enquanto a etapa estiver ativa, sendo finalizada pela desativação da etapa 2

b. ação iniciada ao ser ativada a etapa 2 e continuada mesmo após a sua desativação, sendo finalizada somente com a ativação da etapa 10.

Fig. 2.2 – Representação de várias ações associadas a mesma etapa

2.3.1.1 Ações Detalhadas – Qualificadas

A ação não continuada é executada pelo período de tempo em que a etapa estiver ativa. Sendo, portanto necessário em alguns casos condicionar ou limitar a execução da ação. Isto é possível através da utilização das ações detalhadas. A simbologia é representada na figura 2.3, sendo que o campo "b" deve possuir pelo menos o dobro da largura dos campos "a" ou "c".

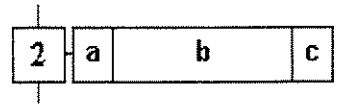


Fig. 2.3 – Representação genérica da ação detalhada

Sendo:

a- o qualificador que define como a ação à etapa será executada;

b- declaração textual ou simbólica da ação;

c- somente quando necessário.

São definidos cinco qualificadores:

- **S** (*stored* – armazenada mantida) – a ação é mantida ou continuada após a desativação da etapa sem a necessidade de especificar textualmente ou simbolicamente na ação, até ser finalizada por outra etapa. Exemplo demostrado na figura 2.4.

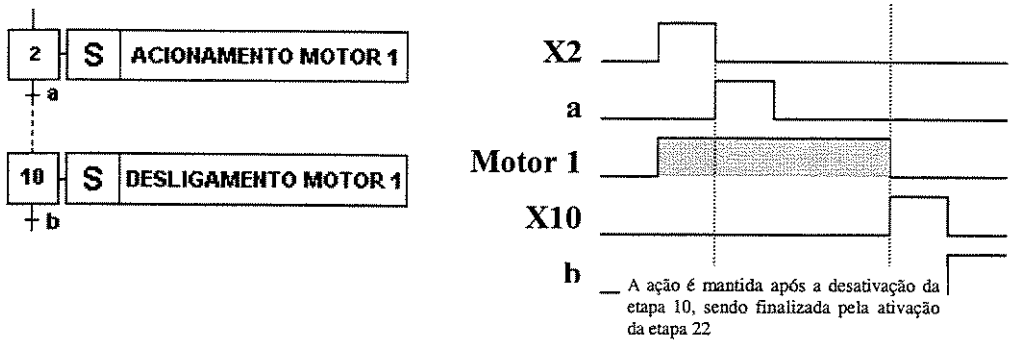


Fig. 2.4 – Qualificador "S"

- **D** (*delayed* – atrasada) – a ação é iniciada após decorrido o tempo (atraso) especificado, e mantida enquanto a etapa estiver ativa. Se a etapa permanecer ativada por um período menor que o especificado, a ação não é iniciada. Conforme demostrado na figura 2.5.

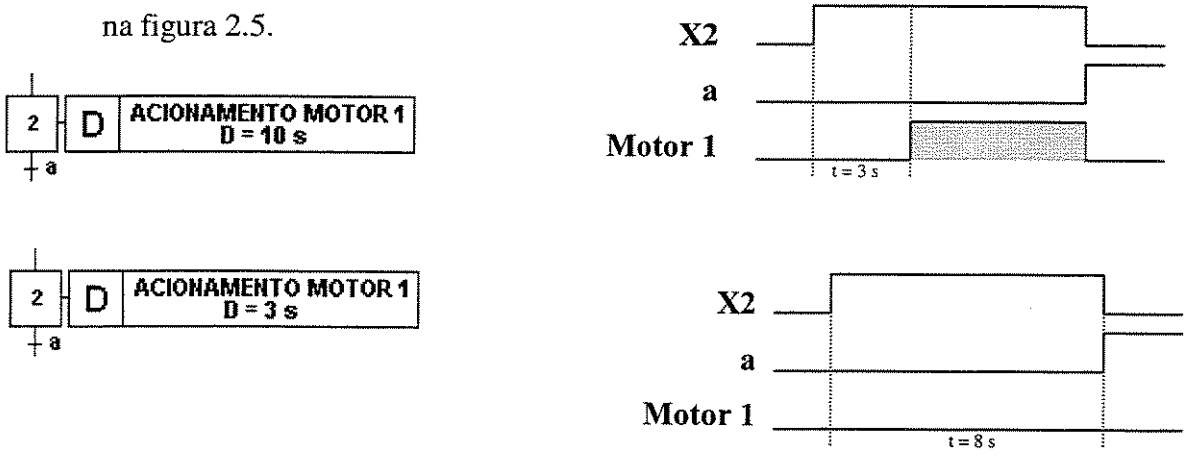


Fig. 2.5 – Qualificador "D"

- **L** (*time limited* – limitada pelo tempo) – a ação é mantida enquanto a etapa estiver ativa ou até ser atingido o tempo especificado. Conforme demonstrado no exemplo da figura 2.6.

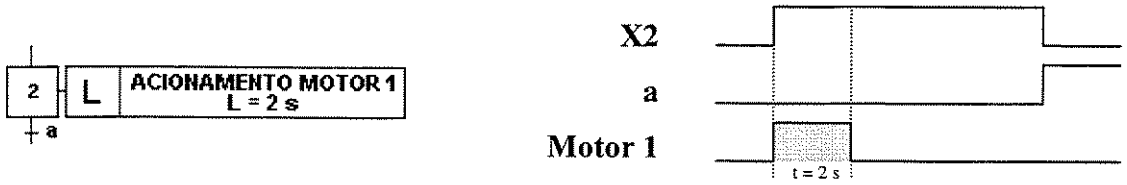


Fig. 2.6 - Qualificador "L"

- **P** (*pulse shape* – pulsada) – quando o tempo de execução for muito pequeno, utiliza-se o especificador **P** em vez de **L**. Conforme demostrado na figura 2.7.

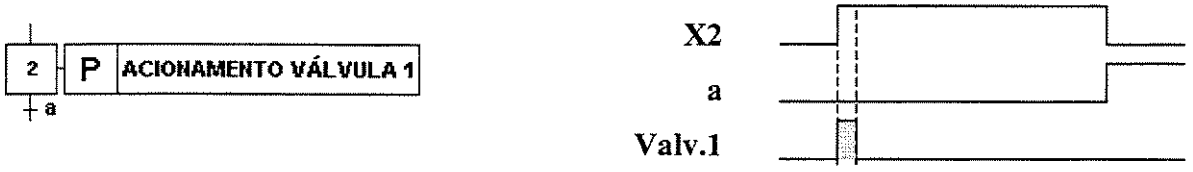


Fig. 2.7 – Qualificador "P"

- **C** (*conditional* – condicional) – a ação é iniciada e mantida enquanto a etapa estiver ativa, desde que a condição lógica especificada seja satisfeita, podendo ser indicada interna ou externamente ao símbolo. Conforme demonstrado na figura 2.8.

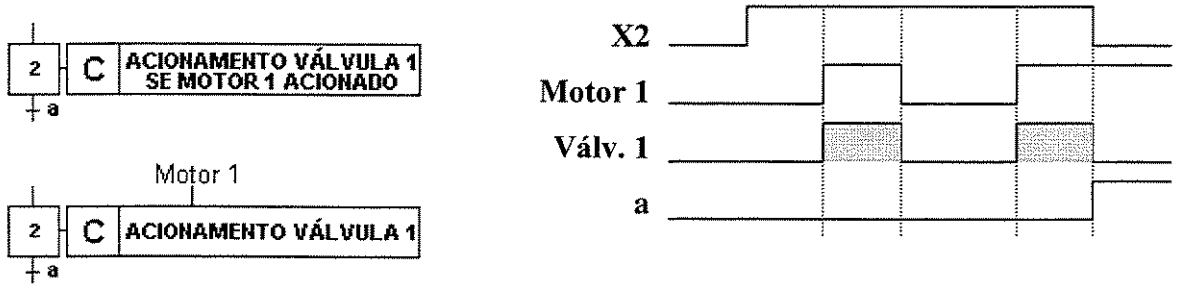


Fig. 2.8 – Qualificador "C"

Além de um único qualificador, uma ação pode ser detalhada através da combinação de qualificadores, sendo que neste caso a ordem de apresentação dos mesmo determina a seqüência

que deve ser satisfeita para a execução da ação associada à etapa. É possível utilizar qualquer combinação de qualificadores, excluindo-se "LP" e "PL", que são funções de tempo.

Uma seqüência de etapas que apareça mais de uma vez no diagrama funcional e em situações distintas, pode ter a representação reutilizada, conforme apresentado na figura 2.9.

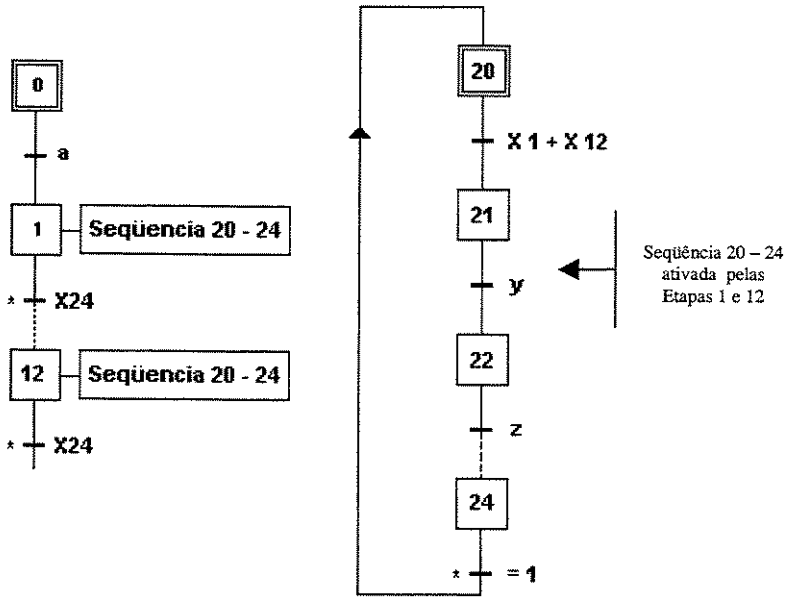


Fig. 2.9 – Reutilização de uma seqüência

Uma etapa também pode ser apresentada mais detalhadamente através de sub-etapas, nas quais pode conter seqüências, estas etapas são conhecidas também como macro etapas.

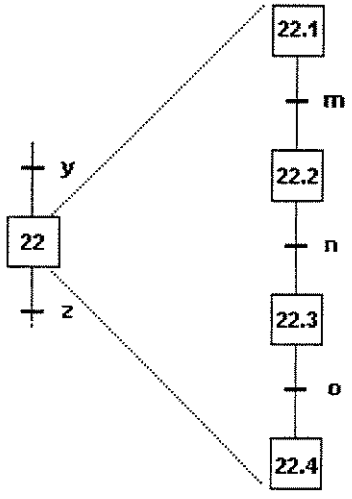


Fig. 2.10 – Macro etapa – detalhamento de uma etapa

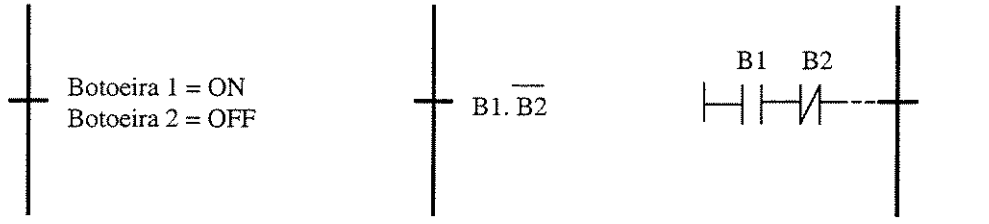
2.3.2 Transição

É a função lógica que coordena a possibilidade de evolução entre as etapas. Em um determinado instante uma transição pode ser válida ou não, permitindo a evolução entre a etapa ativa à etapa inativa. Para que uma transição seja habilitada, ou seja, possível de ser transposta, é necessário que todas as etapas imediatamente precedentes estejam ativas. E, para que a transição seja transposta, é necessário que esteja habilitada e a condição associada, a receptividade, seja verdadeira. Esta transposição ocasiona a ativação de todas as etapas imediatamente seguintes e a desativação de todas as etapas imediatamente precedentes, simultaneamente. A figura 2.11 representa a simbologia de uma transição.

Fig. 2.11 – Simbologia da transição

A receptividade associada à transição é escrita sob a forma de uma preposição lógica, ou seja, é uma função combinatória de informações lógicas, exteriores, de variáveis auxiliares do estado ativo ou inativo de outras etapas e determina o fim da etapa que a precede. Uma etapa permanece ativa até que a receptividade que lhe está associada seja verdadeira.

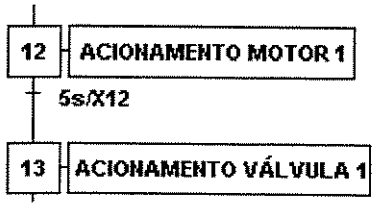
As receptividades, podem ser representadas através de declarações textuais, expressões booleanas ou símbolos gráficos padronizados, colocados à direita ou esquerda das transições, como demostrados na figura 2.12.



Declaração Textual *Expressão Booleana* *Simbologia Gráfica - Ladder*

Fig. 2.12 – Representação das receptividades

Sendo à cada transição associada uma receptividade, esta é a condição lógica que permite distinguir entre todas as informações disponíveis num dado instante: apenas aquela que permite a evolução da etapa. Estas receptividades podem apresentar detalhes que se relacionam ao tempo ou ao estado lógico de uma variável, como no exemplo da figura 2.13.



A receptividade apresenta que a transição deve ocorrer somente após 5 segundos decorridos da ativação da Etapa 12

Fig. 2.13 – Receptividade relacionada ao tempo

Existem transições que devem ser transpostas incondicionalmente, desde que habilitadas, utilizando-se a notação "= 1". Conforme exemplo da figura 2.14.

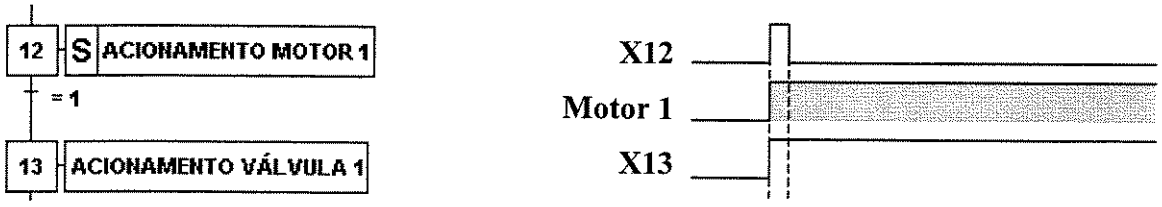


Fig. 2.14 – Transição incondicional

2.3.3 LIGAÇÕES ORIENTADAS

As ligações orientadas indicam o caminho de evolução de estado do Diagrama, ou seja, as etapas são conectadas às transições, e estas às etapas, através das ligações orientadas. As ligações entre as etapas são orientadas e irreversíveis. Sempre que possível as ligações entre etapas devem ser representadas por linhas verticais ou horizontais, podendo ser oblíquas para facilitar a compreensão do diagrama.

Convencionalmente o sentido de evolução é de cima para baixo. Porém, quando o sentido for inverso, será necessário a inclusão de uma seta ou ainda, para melhorar o entendimento em certas ocasiões, conforme demonstrado na figura 2.15.

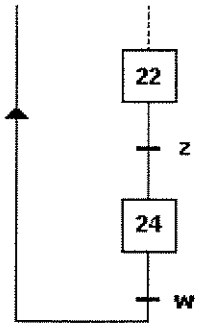


Fig. 2.15 – Ligações orientadas

O cruzamento de ligações deve ser evitado, utilizando-se as estruturas apresentadas na figura 2.16.

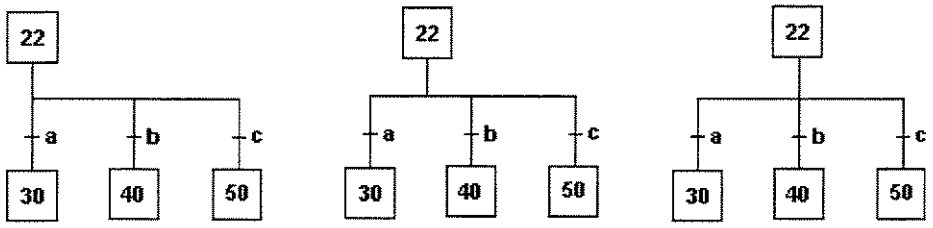


Fig. 2.16 – Ligações orientadas não cruzadas

Necessitando-se interromper uma ligação orientada para continuá-la em outra página, a referência da etapa seguinte e o número da página em que se encontra devem ser indicadas. Como demonstrado na figura 2.17.

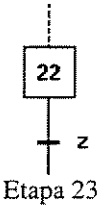


Fig. 2.17 – Indicação de continuidade da ligação orientada

As ligações entre as etapas podem ser: seqüenciais, divergentes em OU, convergentes em OU, divergentes em E e convergentes em E.

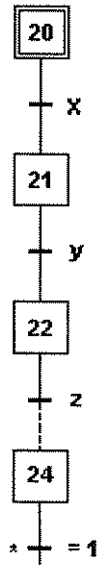


Fig. 2.18 – Ligação seqüencial

Nas ligações seqüenciais, figura 2.18, a transição diz-se validada quando a etapa precedente está ativa. Para que a transição seja transposta é necessário que esta seja validada e que simultaneamente, a receptividade que lhe é associada seja verdadeira.

As ligações entre etapas divergentes em OU ou seqüências seletivas, figura 2.19, são utilizadas para representar decisões, nas quais é definida apenas uma seqüência a ser seguida. As condições associadas às transições do início de cada seqüência devem ser exclusivas, ou seja, apenas uma condição pode ser verdadeira em determinado instante. O início de uma seqüência seletiva é sempre divergente, e o final sempre convergente.

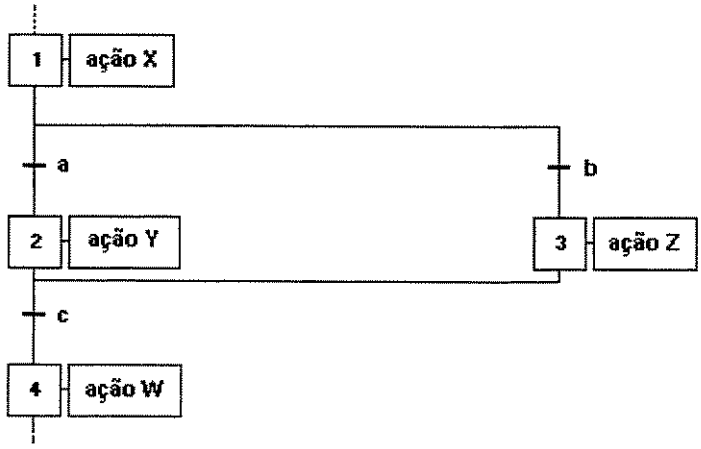


Fig. 2.19 – Seqüência seletiva

As ligações divergentes em E ou seqüências simultâneas, figura 2.20, definem seqüências que são iniciadas paralelamente, através de uma única transição, ou seja uma única transição ativará simultaneamente todas as etapas imediatamente seguintes. Após a ativação, cada seqüência tem sua evolução independente. O início de uma seqüência simultânea é sempre divergente, e o final sempre convergente, ambos indicados por linhas duplas.

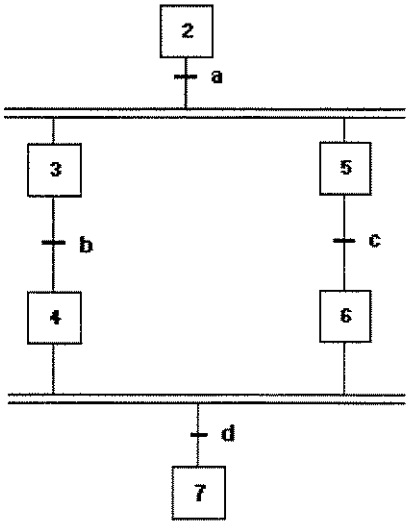


Fig. 2.20 – Seqüências simultâneas

2.3.4 REGRAS DE EVOLUÇÃO

Apresentar o Diagrama através do conjunto de elementos , fornece somente uma visão estática do sistema automatizado que se pretende descrever. Para se obter uma visão dinâmica do mesmo é necessário que se aplique as regras de evolução. Estas regras podem ser ditas como a representação da alternância entre as etapas e transições e vice e versa, ressaltando que nenhuma ação é realizada ou nenhuma receptividade é validada em um espaço de tempo nulo, isto é, estas operações não são simultâneas.

1ª Regra: A situação inicial do sistema é dada pelas etapas iniciais, que são ativadas incondicionalmente no início da operação do mesmo. Podem existir tantas etapas iniciais quantas forem necessárias. Obrigatoriamente, deve existir pelo menos uma etapa inicial em cada Diagrama.

2ª Regra: Uma transição é transposta somente se a mesma estiver habilitada e a receptividade associada a ela for verdadeira.

3ª Regra: A transposição de uma transição ocasiona simultaneamente a ativação da(s) etapas(s) imediatamente seguinte(s) e a desativação da(s) etapa(s) imediatamente precedente(s).

4ª Regra: As transições que serão transpostas simultaneamente devem ser representadas através de linhas duplas agrupando as etapas imediatamente precedentes e as imediatamente procedentes. Caso, estejam dispostas separadamente no Diagrama, utiliza-se a representação demostrada na figura 2.21. Todas as transições simultâneas devem conter o asterístico e a referência da(s) etapa(s) envolvida(s).

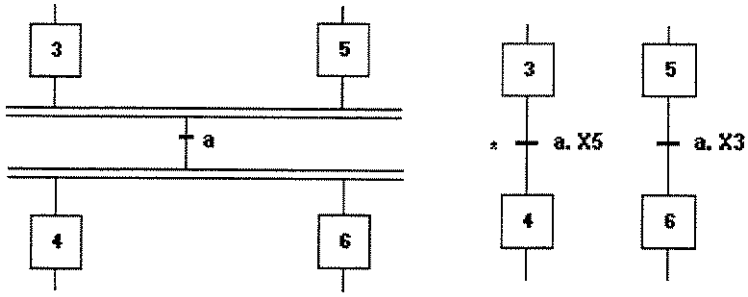


Fig. 2.21 – Transições Simultâneas

5ª Regra: Se, no momento da ativação de uma etapa, a receptividade associada a transição desta à etapa seguinte for verdadeira, a mesma não ocorrerá. Como demonstrado no exemplo da figura 2.22.

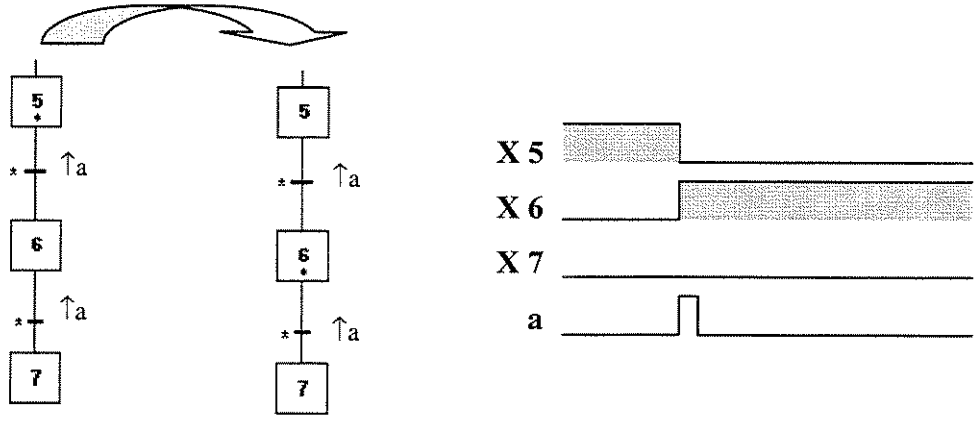


Fig. 2.22 – Condição Verdadeira e Imediatamente Seguinte

6ª Regra: Caso ocorra a ativação e desativação de uma mesma etapa simultaneamente, a ativação é prioritária.

7ª Regra: Normalmente, a tecnologia utilizada para a implantação do Sistema Automatizado determina o tempo de transposição de uma transição, ou a ativação de uma etapa, podendo ser extremamente curtas. Porém, estes tempos nunca podem ser considerados nulos, ou iguais a zero.

2.4 Conclusão

Neste capítulo foram apresentados as principais formas de descrição dos Sistemas Automatizados de Produção. Dentre as formas apresentadas, considera-se o SFC a forma de descrição de sistemas como a melhor linguagem em qualquer nível, tanto da Parte Comando como da Parte Operativa, portanto será adotado neste trabalho tanto para programação como para descrição.

Capítulo 3

Sistema Automatizado de Produção e sua Concepção

Neste capítulo serão abordados os elementos constituintes de um Sistema Automatizado de Produção. Podemos demonstrar estes elementos através de uma plataforma para mistura de cores, constituída de manipuladores robóticos, CLP, sensores, atuadores e conjunto mecânico.

3.1 Plataforma para Mistura de tintas – Especificação Funcional

A plataforma para mistura de tintas, pode ser dividida em Parte Operativa e Parte Comando. A Parte Operativa é constituída de dois manipuladores robóticos, mesa rotativa acionada por motor de C.C, atuadores diversos (bombas e o misturador) e sensores. A Parte Comando, contem um CLP e PC's controlando os dois manipuladores robóticos.

Os manipuladores robóticos foram utilizados em duas células robotizadas, sendo uma responsável pela escolha de cores e outra pela lavagem e mistura da tinta, além de pegar e depositar o recipiente codificado na área de depósito. Para diferenciar estas duas células de trabalho, estaremos definindo como rob1, o manipulador robótico responsável pela escolha de cores e como rob2, o manipulador robótico responsável pelo posto de lavagem e mistura. A integração entre estes postos é demonstrado na figura 3.1.

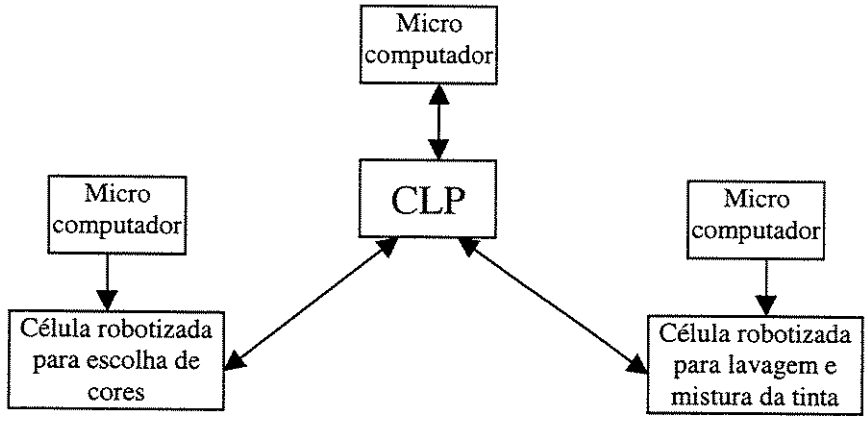


Fig. 3.1 - Esquematização dos elementos da plataforma

Acionando-se um botão de inicialização (START) o conjunto mesa rotativa / manipuladores robóticos deverão inicializar através uma verificação de suas posições obtidas através de sensores. O processo poderá se inicializar através de dois modos: código de barras ou botões de seleção de cores. Uma vez confirmada a combinação de cores, o rob2 irá pegar um recipiente vazio e o colocará na posição de depósito. Uma vez concluído o trajeto, o rob1 realizará o CICLO DE MISTURA. Neste ciclo o rob1 pegará o recipiente de mistura da mesa rotativa levando-o à bomba de cores, que despejará uma quantidade de tinta no recipiente, retornando em seguida o recipiente de mistura à mesa rotativa. Este ciclo será repetido quantas vezes for necessário, em função da combinação de cores escolhidas ou individualmente. O recipiente com a mistura de cores estando na mesa, esta realizará um movimento de rotação 180°, levando-o ao segundo manipulador robótico.

Confirmado o movimento de rotação da mesa, o rob2 realizará o CICLO MIXER. Neste ciclo o rob2 pegará o recipiente de mistura da mesa rotativa e o levará ao misturador. Concluído esta etapa, o manipulador robótico despejará o conteúdo do recipiente no reservatório. Ao término do movimento, o recipiente será devolvido à mesa rotativa. Concluído o ciclo, o rob2 passará a executar o CICLO DE LAVAGEM. Pegará o recipiente da mesa rotativa e levará à bomba d'água e em seguida ao misturador. Após agitado o conteúdo do recipiente, este será despejado no reservatório. Este ciclo será realizado duas vezes para a completa lavagem do recipiente. Concluído o ciclo, o rob2 devolverá o recipiente à mesa rotativa, e um novo ciclo será desenvolvido.

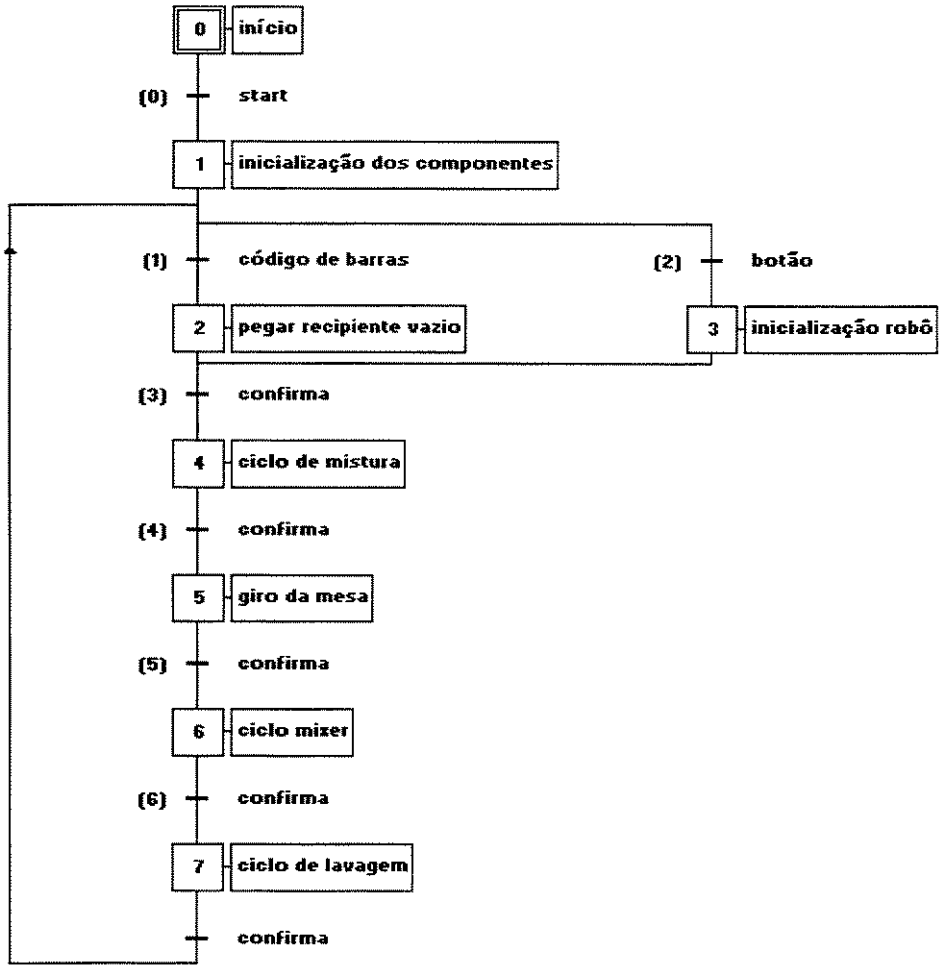


Fig. 3.2 - SFC funcional da plataforma

O diagrama funcional apresentado na figura 3.2 também descreve o funcionamento da plataforma. Pode-se verificar, pela comparação das duas formas de descrição do funcionamento, que a linguagem verbal não é a melhor forma de descrever o funcionamento do sistema automatizado.

Serão descritos a seguir os diferentes elementos de um SAP levando-se em consideração a PO e a PC. Inicialmente serão descritos os elementos integradores da Parte Operativa e depois da Parte Comando. No caso do sistema robótico, como se trata de um sistema de integração da PO e da PC, ele será descrito separadamente.

3.2 Elemento de Transferência

O sistema de transferência utilizado na plataforma apresentada é constituído de uma mesa rotativa acionada por um motor C.C. Considerando-se a mesa como um posto de trabalho, podemos diferenciar a sua Parte Operativa como sendo a própria mesa, conjunto de redução / transmissão e motor elétrico. Como Parte Comando teremos os sensores que determinam o sentido de giro da mesma.

3.3 Elementos de Atuação

Nos elementos de atuação do sistema apresentado, temos as bombas elétricas responsáveis pela deposição das tintas e da água no recipiente de mistura e o motor elétrico responsável pela mistura da tinta.

3.4 Controladores Programáveis

O controlador programável é um equipamento eletrônico digital que possui uma memória programável para armazenar instruções que implementam funções da lógica binária, permitindo a execução de comandos, controle e monitoração de máquinas e de processos (Bolmann, 1997). Ou seja, um computador industrial capaz de armazenar instruções para implementação de funções de controle, além de realizar operações lógicas e aritméticas, manipulação de dados e comunicação em rede, sendo utilizado no controle de processos e máquinas industriais.

Historicamente podemos dizer que os primeiros CP's surgiram em 1968 quando a Divisão Hydromatic da GM determinou os critérios para projeto do CP, sendo que o primeiro dispositivo a atender às especificações foi desenvolvido pela Gould Modicon em 1969. Em 1971, ocorreram as primeiras utilizações fora da industria automobilística. Em 1975, foi introduzido o controlador PID. Até 1977 eram construídos com componentes eletrônicos discretos; a partir desta época estes passaram a ser substituídos por microprocessadores, sendo daí em diante largamente aceito industrialmente.

Os Controladores Lógicos Programáveis, vieram ao mercado com a finalidade de substituir os antigos sistemas a relês, eliminando com isto toda a fiação para intertravamento dos relês de controle e facilitando as alterações necessárias.

Podemos dizer que, uma estrutura básica de um CLP é composto pelas unidades de entrada e saídas e o CPU, contendo as memórias e o processador.

O módulo de entrada é a unidade que recebe os sinais elétricos vindos dos elementos de sinais (botão, fim de curso, sensor ótico, magnético ou indutivo), filtrando e enviando ao processador. O módulo de saída, é a unidade que transmite os sinais gerados pelo processador aos atuadores externos (solenóides, lâmpadas, motores, etc.).

A CPU é a unidade que contém as memórias que armazenam temporariamente ou permanentemente os dados de entrada, o programa a ser executado e os dados de monitoração e controle do próprio CLP; e o processador, que é constituído por microprocessadores usuais, e tem como função coletar os dados de entrada, processar os sinais segundo a programação e enviar os sinais aos módulos de saídas.

A estrutura básica de um CLP pode ser representado conforme o esquema da figura 3.3:

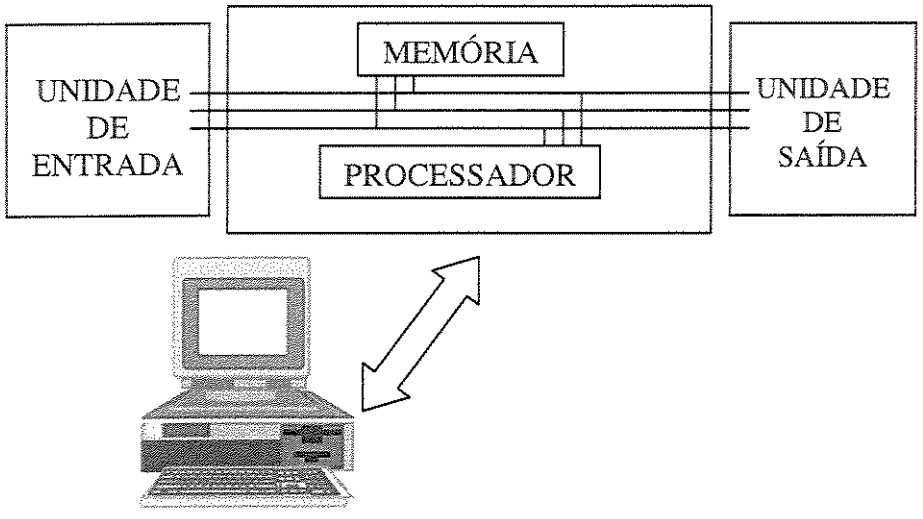


Fig. 3.3 - Estrutura básica de um CLP

As linguagens de programação desenvolvidas para os CLP's, podem ser representadas em três formas:

- Redes de contato (Ladder): similar aos esquemas elétricos de reles e contatores;
- Blocos funcionais; similares aos esquemas elétricos de circuitos digitais (AND, OR, XOR, etc.);
- Lista de instruções: é a programação diretamente apoiada nas funções lógicas binárias e se assemelha bastante aos programas escritos em assembler.

A mais utilizada é a linguagem Ladder, que está presente praticamente em todos os CLP's disponíveis no mercado. Esta linguagem, baseia-se nas redes de contatos dos esquemas elétricos, sendo por isso, uma linguagem gráfica de fácil manipulação e, mesmo existindo diferenças entre os fabricantes quanto às representações das instruções, são facilmente assimiladas pelo usuário. Abaixo, a figura 3.4 apresenta um CLP industrial, contendo as entradas, saídas e módulo de processamento.

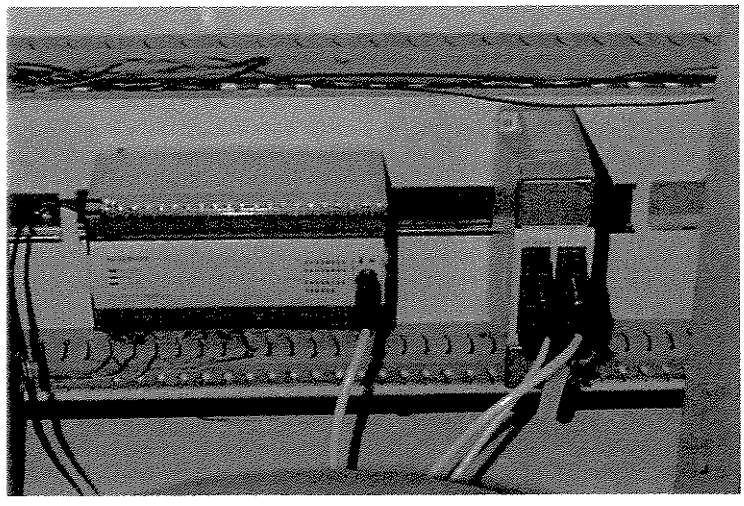


Fig. 3.4 - CLP industrial

A programação de um CLP é demonstrada nos exemplos abaixos e, como comentado anteriormente, esta pode ser realizada utilizando linguagens tais como: Ladder, SFC e Lista de Instruções.

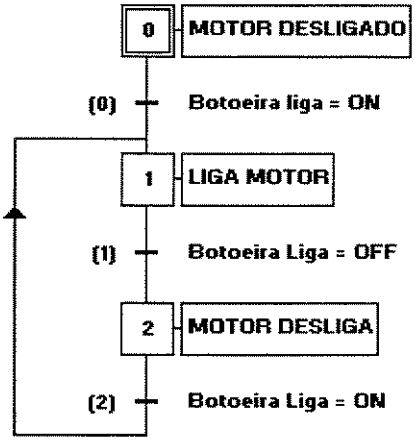


Fig. 3.5 - Programa em SFC

Para o mesmo exemplo apresentado acima podemos ter uma representação em Ladder, como apresentado na figura 3.6:

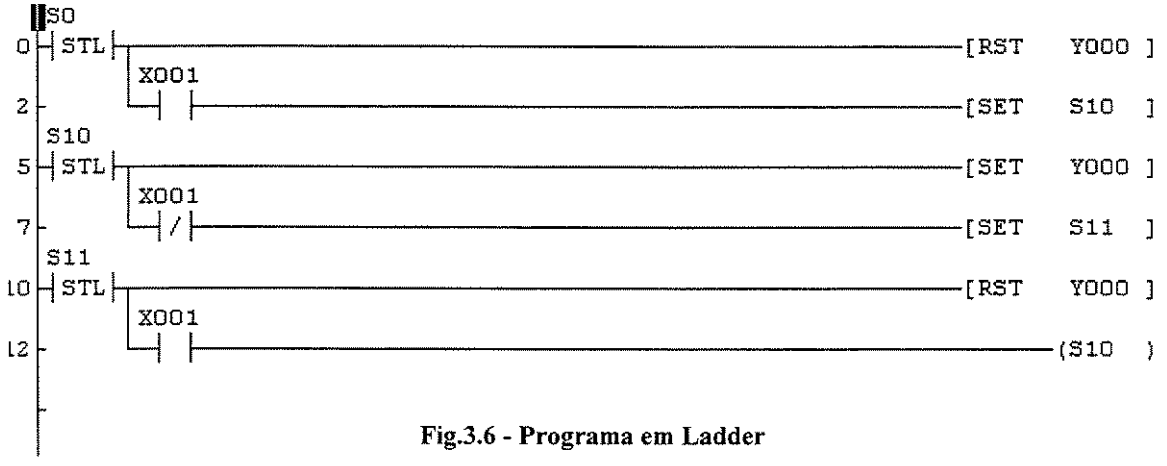


Fig.3.6 - Programa em Ladder

O mesmo exemplo, pode ainda ser apresentado em lista de instruções conforme tabela 3.1:

STEP	INSTRUCTION	
0	STL	S0
1	RST	Y000
2	LD	X001
3	SET	S10
5	STL	S10
6	SET	Y000
7	LDI	X001
8	SET	S11
10	STL	S11
11	RST	Y000
12	LD	X001
13	OUT	S10

Tab. 3.1 - Programa em lista de instruções

3.5 Sistemas Robóticos

Automação e robótica são duas tecnologias intimamente relacionadas. Num contexto industrial podemos definir a automação como uma tecnologia que se ocupa do uso de sistemas mecânicos, eletrônicos e à base de computadores na operação e controle da produção. Como exemplo tem-se: máquinas de montagens mecanizadas, sistemas de controle de realimentação, máquinas operatrizes dotadas de comando numéricos e robôs (Groover, 1988).

A definição de um robô industrial dada pela Associação das Indústrias de Robótica (RIA) é a seguinte:

" Um robô industrial é um manipulador reprogramável, multifuncional, projetado para mover materiais, peças, ferramentas ou dispositivos especiais em movimentos variáveis programados para a realização de uma variedade de tarefas. "

A robótica hoje, continua sendo utilizada para o desenvolvimento, estrutural e funcional de máquinas como: (Ferreira, 1991)

- Robôs manipuladores, com estrutura antropomórfica ou não, capazes de pegar objetos e deslocá-los, ou de atuar sobre objetos com ferramentas específicas;
- Robôs móveis, que se deslocam sobre rodas, patas ou lagartas;
- Robôs de supervisão, que verificam e selecionam objetos.

Sendo considerada hoje a mola mestra de uma nova mutação dos meios de produção, isto devido a sua versatilidade, em oposição a automação fixa. Os robôs, graças ao seu sistema lógico ou informático, podem ser reprogramados e utilizados em uma grande variedade de tarefas, sendo que a reprogramação não é o fator mais importante na versatilidade desejada e sim a adaptação às variações no seu ambiente de trabalho, mediante um sistema adequado de percepção e tratamento de informação.

Este elemento pode ser apresentado através do ROBIX^{TM} que é um kit didático para ensino aprendizagem de robótica e que pode ser montado em diversas configurações. Neste trabalho,

adotamos uma configuração que assemelha-se a um braço mecânico, ou manipulador robótico, com 5 Graus de Liberdade.

Sua programação é dada através de software próprio. A interface de programação é de estilo industrial, sendo operada por teclas ou mouse. Ou seja, os comandos podem ser digitados ou operados por aprendizagem. Possui recursos de debug (execução passo-a-passo e janela de mensagens de erros) e de visualização de status do robô e dos seus sensores. O editor de programas possui todos os recursos de um editor de texto profissional (cortar, colar, salvar, salvar como, etc.) com janelas que ajudam na sua operação.

O Device Driver, tem a função de fazer a interligação, em nível de hardware, da Interface de Programação com o Módulo de Controle (Adapter). Permite a utilização de linguagem de alto nível para comunicação (Basic, C,..)e o programa RCS-6, C e QBASIC, sendo que nestas duas é necessário a inclusão de bibliotecas. Permite também que cada servomotor opere segundo parâmetros estabelecidos pelo usuário como aceleração, desaceleração, posição inicial, etc., interpolando linearmente os movimentos dos servomotores.

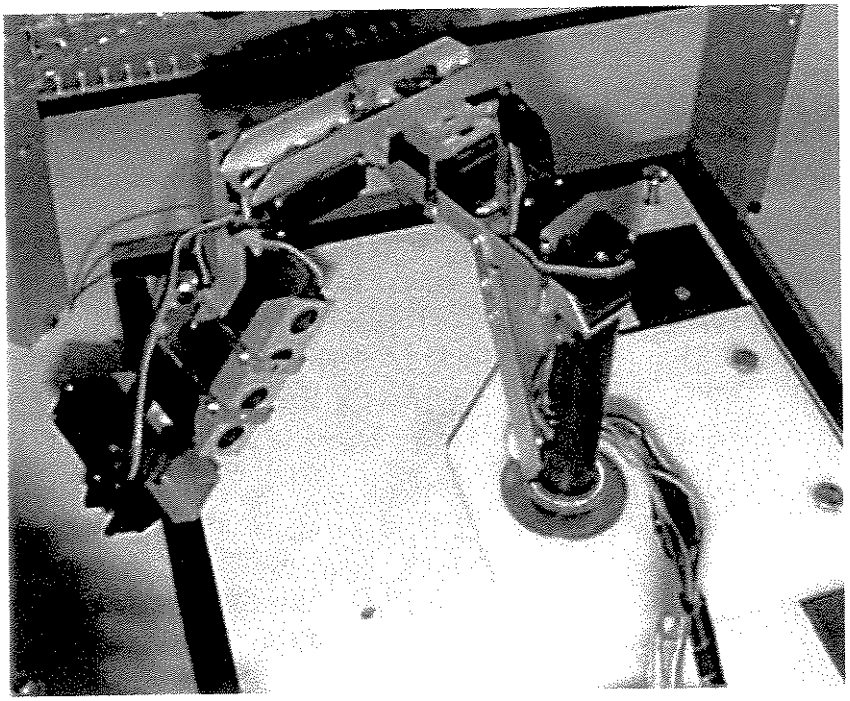


Fig. 3.7 - ROBIX™

A figura 3.7 apresenta o ROBIX™ na configuração utilizada, em que 5 servomotores são responsáveis pela movimentação dos 5 Graus de Liberdade enquanto 1 servo motor comanda a abertura e fechamento da garra.

Desta forma teremos a interface de programação que cria uma janela de edição de programas de controle e o device driver sendo instalado na memória para ser ativado automaticamente pela interface de programação sempre que for necessária a comunicação com o Módulo Controlador.

Todos os arquivos apresentados são carregados e executados pelo arquivo rbx.bat. A tela inicial de programação é apresentada na figura 3.8:

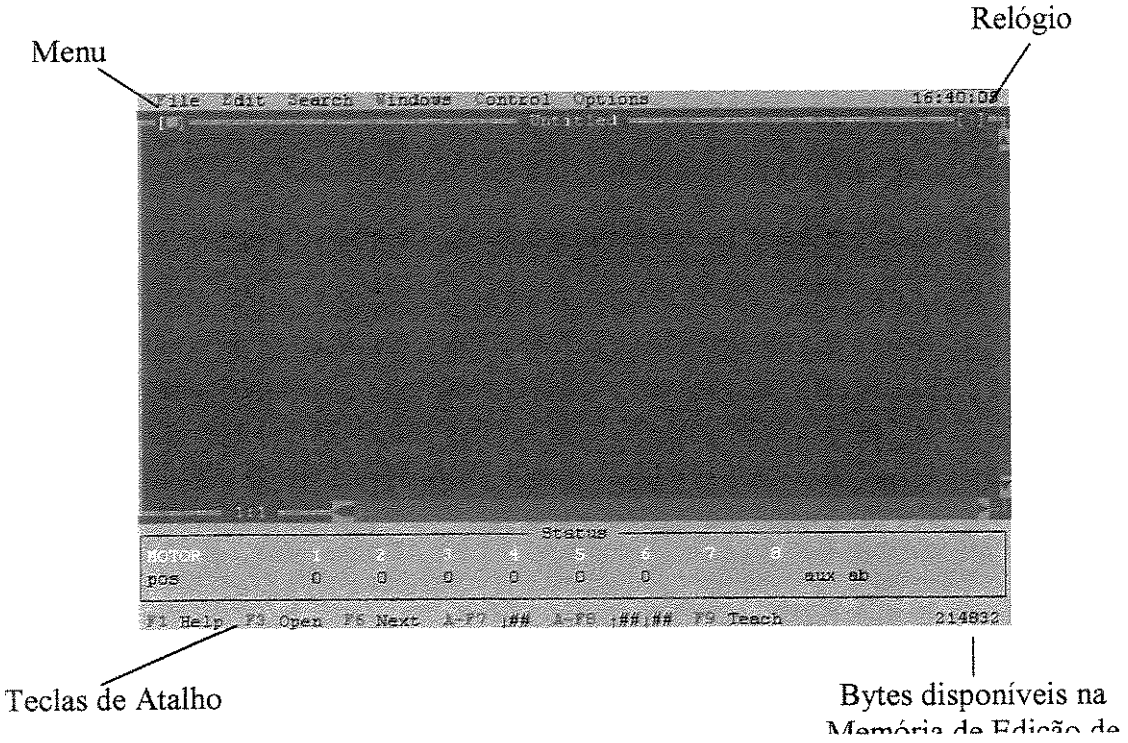


Fig. 3.8 - Tela inicial da interface de programação do ROBIX™

A programação do manipulador robótico pode também ser realizada conforme comentado anteriormente através de aprendizagem, onde cada grau de liberdade é operado individualmente (Teach In), conforme mostrado na figura 3.9.

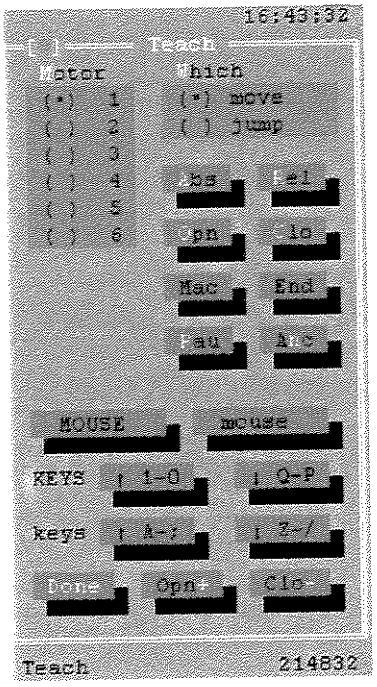


Fig. 3.9 - Tela Teach de programação do ROBIX

A janela apresentada na figura 3.9 é o "painel de controle" do ROBIX^{TM}. Com esta janela aberta é possível mover cada junta do robô, individualmente ou conjuntamente, e gravar as posições assumidas para ir criando um programa de controle.

Desta forma, o teclado e/ou o mouse são transformados num "Teach Pendant", uma caixa de programação de posições para o robô. Sendo possível mover os servomotores de cada junta em duas velocidades: rápida e lenta.

Os principais parâmetros utilizados para o comando dos servomotores são:

- pos: posição corrente
- maxspd: máxima velocidade durante um movimento (move)
- accel: aceleração durante um movimento (move)
- decel: aceleração durante um movimento (move)
- minpos: min posição depois de um movimento (move ou jump)
- maxpos: min posição depois de um movimento (move ou jump)

- initpos: posição depois de iniciar ou movimentar todos os servomotores para a posição inicial
- power: liga / desliga os servomotores
- invert: inverte as coordenadas dos servomotores
- p0pos: posição 0 (físico).

Para se obter o melhor aproveitamento da utilização destes elementos, as tarefas a serem executadas devem estar estruturadas de forma a obter um certo rendimento, seja ela na quantidade de pontos a serem determinados ou garantir a possibilidade de inclusões futuras de novas tarefas. A figura 3.10 exemplifica esta estruturação:

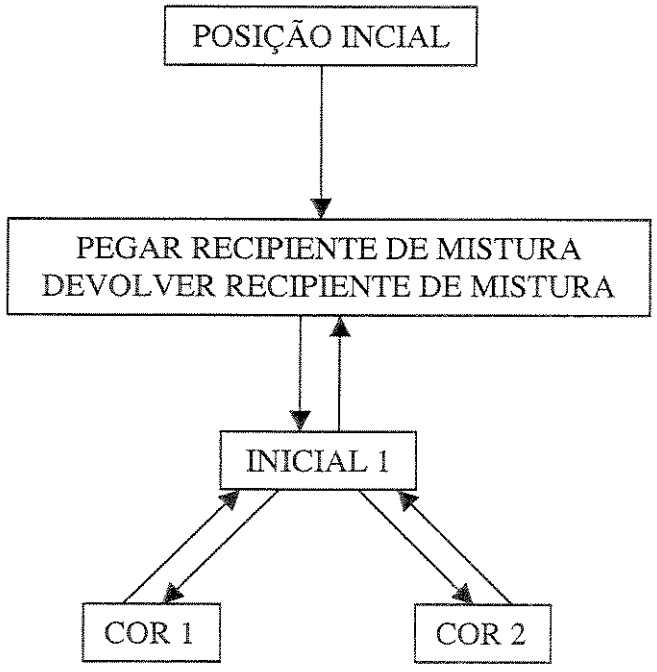


Fig. 3.10 - Estruturação de tarefas

A estrutura apresentada permite que sejam inseridas novas cores. Isto porque sendo o ponto inicial 1 um ponto físico entre as bombas de cor 1 e cor 2, ao adicionarmos outras cores as mesmas podem estar em relação as cores 1 e 2 ou em outro ponto inicial.

Além destes recursos, os robôs possuem saídas e entradas analógicas e digitais que permitem através de um programa de alto nível como o C, de gerenciar e supervisionar tais

entradas e saídas, permitindo desta forma que o CLP realize o gerenciamento das tarefas dos manipuladores robóticos.

3.6 Conclusão

Neste capítulo, foram apresentados os elementos utilizados em um sistema automatizado de produção, enfocando o SAP desenvolvido durante este trabalho. Verifica-se a necessidade cada vez maior da integração estruturada entre vários componentes diversos, o que torna a cada dia, a complexidade destes sistemas ainda maior.

Capítulo 4

Implementação dos Diferentes Elementos do SAP e Integração Final

A integração entre vários elementos automatizados é uma das principais etapas na implantação de um SAP. Portanto, a utilização de uma linguagem consistente para a descrição do sistema e a estruturação a ser utilizada são importantes, pois garantem a flexibilidade e versatilidade do sistema.

Neste capítulo, será apresentado o Sistema Automatizado de Produção desenvolvido como parte integrante da atividade experimental desde trabalho. Este SAP é idealizado em uma Plataforma Didática para Mistura de Cores composta de uma Parte Comando, contendo um CLP e sensores e uma na Parte Operativa contendo, motores elétricos, manipuladores robóticos e bombas.

Para a integração destes elementos, utilizou-se de uma linguagem consistente como o SFC pois, com isto, a facilidade de interpretação e uma estruturação adequada favorece o alcance do objetivo final de integração entre todos os componentes.

Neste trabalho, a estruturação das tarefas e etapas a serem realizadas garante a versatilidade e uma maior flexibilidade do conjunto, através da possibilidade de implementação de novos produtos (cores).

Podemos descrever o funcionamento da plataforma, segundo o SFC funcional apresentado na figura 4.1.

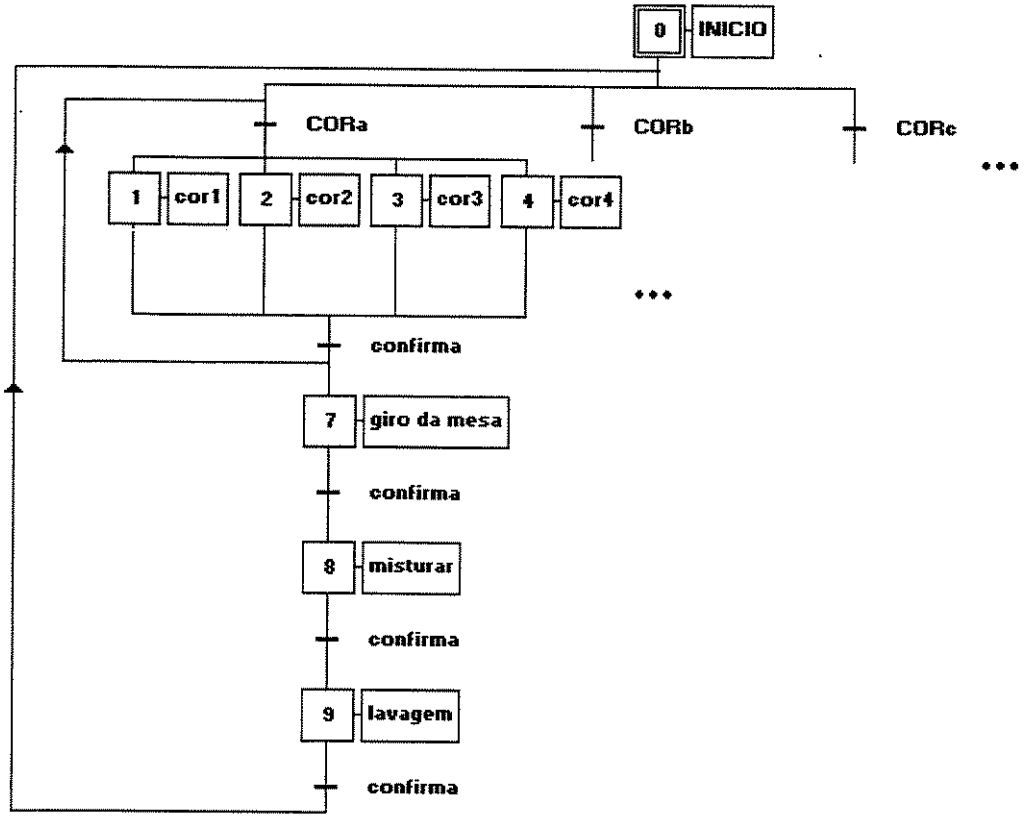


Fig. 4.1 - SFC funcional da plataforma

A Plataforma apresentada, foi dividida em vários postos de trabalho, sendo que cada posto contém sua Parte Comando e sua Parte Operativa, exceto o CLP que possui somente a Parte Comando e a mesa rotativa que possui somente a Parte Operativa . Estes postos são:

- célula robotizada para escolha das cores: tem como PC as entradas e saídas do manipulador robótico e o microcomputador e PO as bombas e o manipulador robótico;
- célula robotizada para lavagem e mistura da tinta: tem como PC as entradas e saídas do manipulador robótico e o microcomputador e PO a bomba, o misturador e o manipulador robótico;
- mesa rotativa;
- CLP.

A figura 4.2, apresenta os postos da Plataforma e interação da PC e PO através de sensores e atuadores, conforme foi descrito anteriormente:

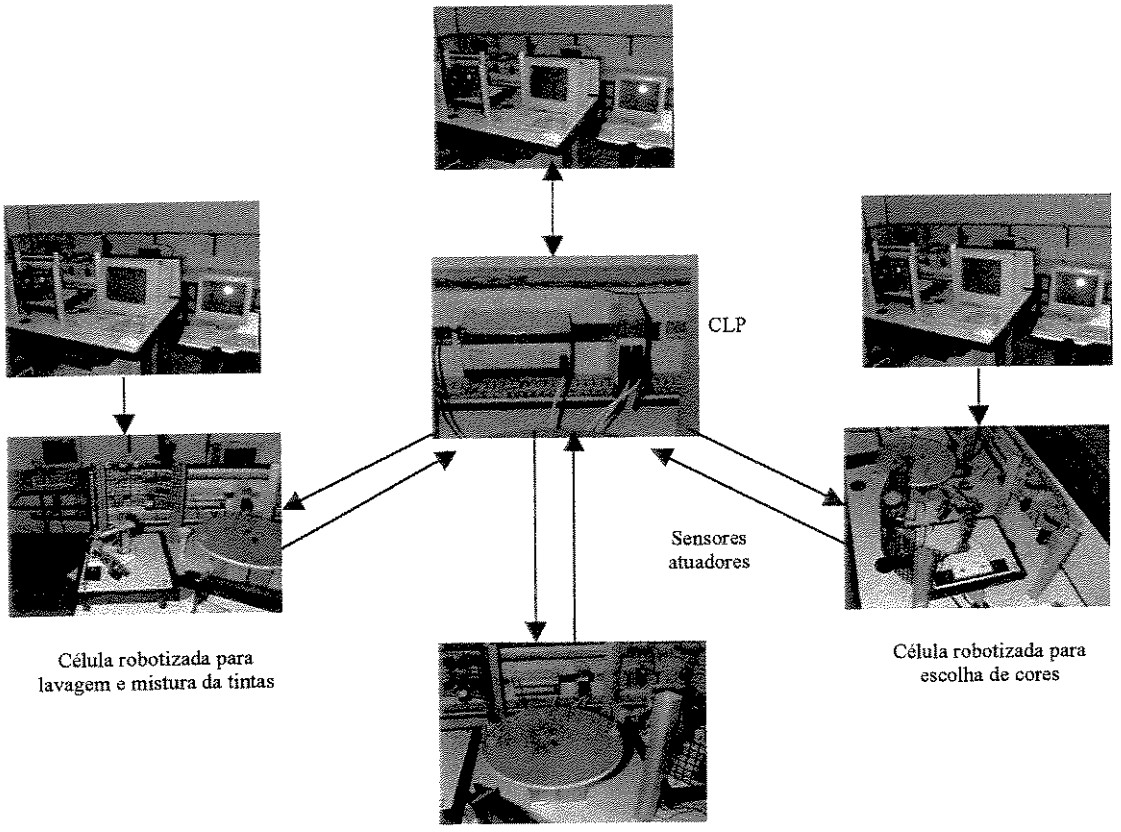


Fig. 4.2 - Plataforma e seus postos

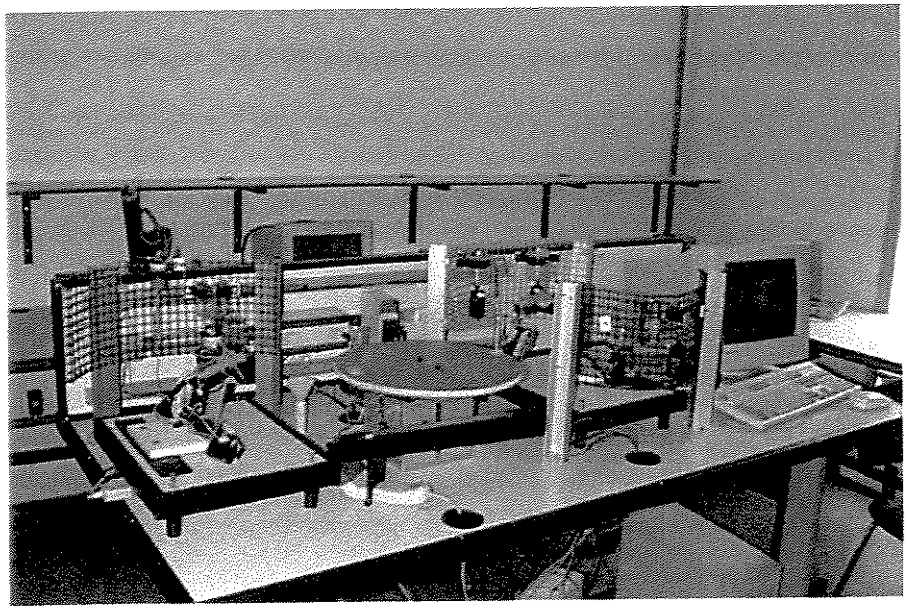


Fig. 4.3 - Plataforma didática para mistura de cores

A figura 4.3 apresenta a foto da plataforma projetada e implementada para o desenvolvimento deste trabalho.

As figuras 4.4 e 4.5 demonstram o funcionamento e os elementos constituintes da célula robotizada para lavagem e mistura da tintas e seus elementos.

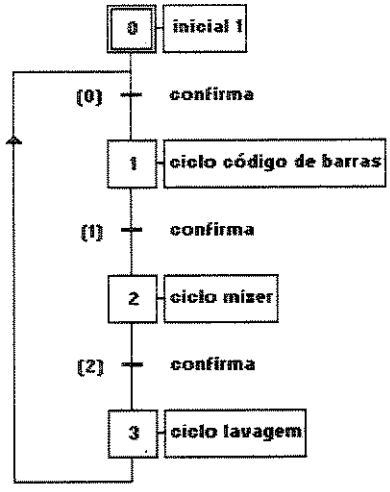


Fig. 4.4 - SFC funcional do posto de lavagem e mistura da tinta

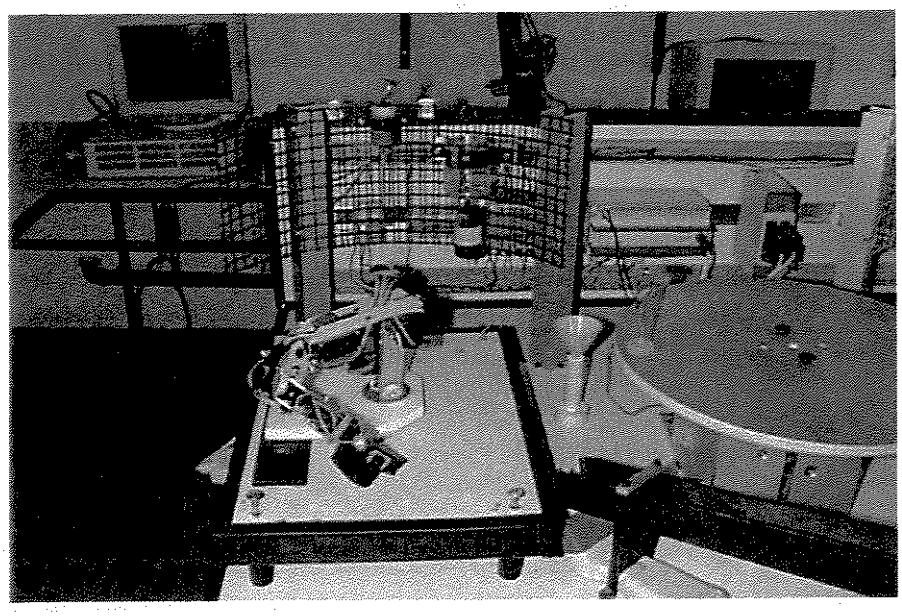


Fig. 4.5 - Foto do posto de lavagem e mistura da tinta

As figuras 4.6 e 4.7 mostram o funcionamento e os elementos constituintes da célula robotizada para escolha de cores.

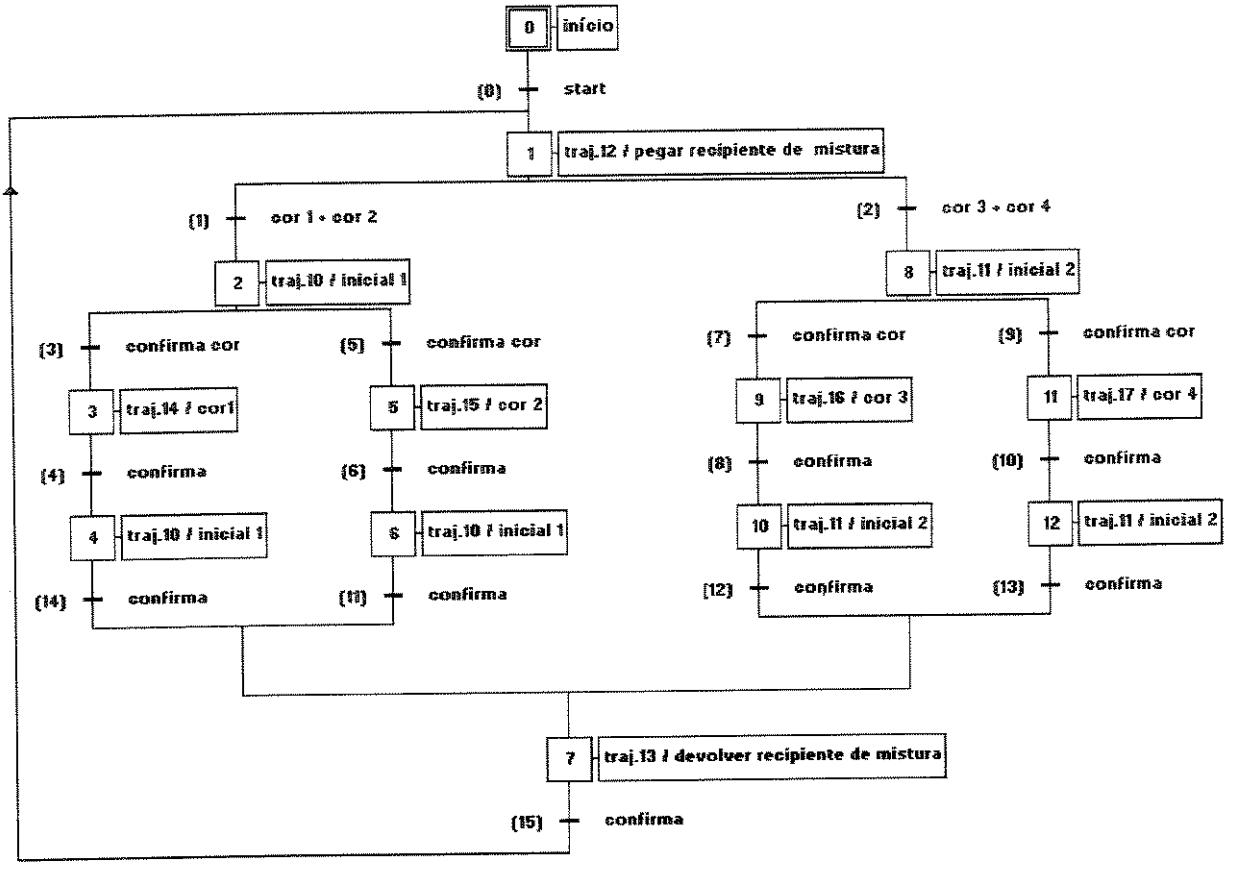


Fig. 4.6 - SFC funcional do posto de escolha de cores

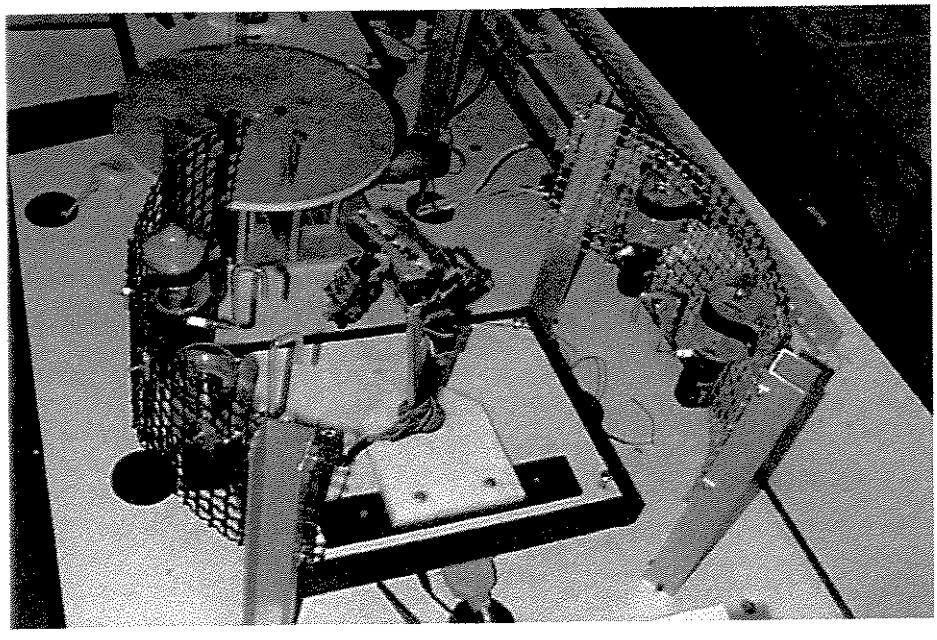


Fig. 4.7 - Posto de escolha de cores

Para a implantação do sistema integrado, foram implementados várias etapas de testes visando o funcionamento individual e conjunta de cada posto de trabalho. Tais testes foram realizados levando-se em consideração os principais elementos constituintes da Parte Comando e da Parte Operativa.

a) células de trabalho:

manipulador robótico:

- realização de trajetórias estruturadas;
- testes de comunicação entre o programa computacional para supervisão de movimentos utilizando a linguagem C e o manipulador.

atuadores:

- bombas: acionamento direto;
- acionamento e dosagem de tintas nas bombas;
- acionamento do misturador com temporização.

código de barras:

- leitura e verificação de dados.

b) mesa rotativa:

- rotação da mesa em sentido horário e anti-horário utilizando-se os sinais dos sensores externos para o comando com o CLP.

c) Controlador Lógico Programável Mitsubishi:

- comunicação com cada elemento individualmente;
- comunicação com o conjunto de elementos de cada posto
- comunicação com o programa em C

A seguir serão apresentadas a descrição das etapas realizadas.

4.1 Realização de trajetórias

As trajetórias do manipulador robótico devem ser testadas individualmente, através do próprio software. Após este teste as mesmas devem ser integradas com o programa em C e da CLP, concluindo desta forma o teste completo de uma tarefa. Para garantir uma melhor

utilização, racionalização e estruturação do programa a ser feito para as trajetórias do manipulador, analisou-se quais os melhores pontos a serem utilizados, concluindo-se que a estrutura apresentada na figura 4.8 seria a melhor.

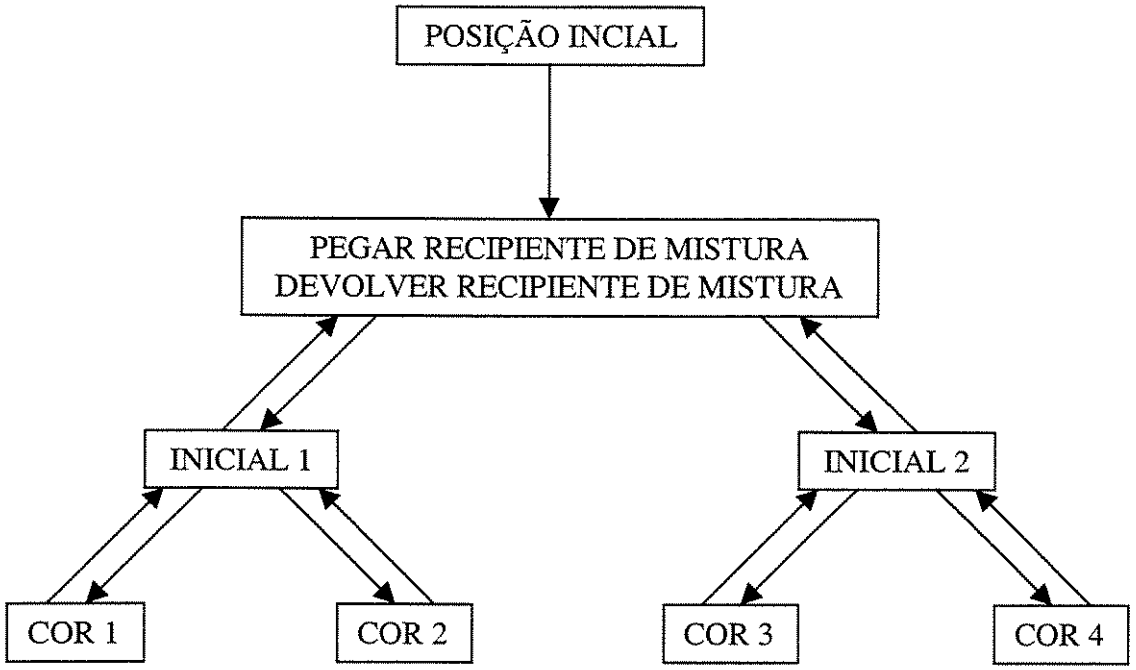


Fig. 4.8 - Estruturação de tarefas através de trajetórias para o posto de escolha de cores

Sendo as tarefas apresentadas através de trajetórias como na estrutura acima, torna-se fácil a implementação de novas trajetórias, possibilitando um crescimento e expansão lateralmente.

O Diagrama apresentado anteriormente refere-se ao posto mostrado na figura 4.7. Este posto de trabalho será portanto, responsável pelas cores de tinta que serão despejadas no recipiente de mistura, levado pelo manipulador robótico, que será o sistema de transferência entre as várias bombas de tinta.

Da mesma forma como apresentada, a estruturação das tarefas em trajetórias para o posto de escolha de tintas, para o posto de lavagem e mistura de tintas, também foi realizado um estudo para a estruturação das tarefas em trajetórias para o manipulador robótico utilizado. Porém, neste caso, a falta de simetria física entre cada ponto a ser utilizado dificultou a obtenção de uma maior

simetria do diagrama. Porém, da mesma forma que no posto anterior, a expansão lateral do mesmo torna-se fácil e estruturado.

Apresentamos na figura 4.9, a estruturação para o posto de lavagem e mistura da tinta, através das tarefas a serem realizadas, por meio de trajetórias.

Este posto de trabalho é demonstrado na figura 4.9, estando presente também elementos como bombas d'água, agitador, reservatório de despejo e recipientes vazios.

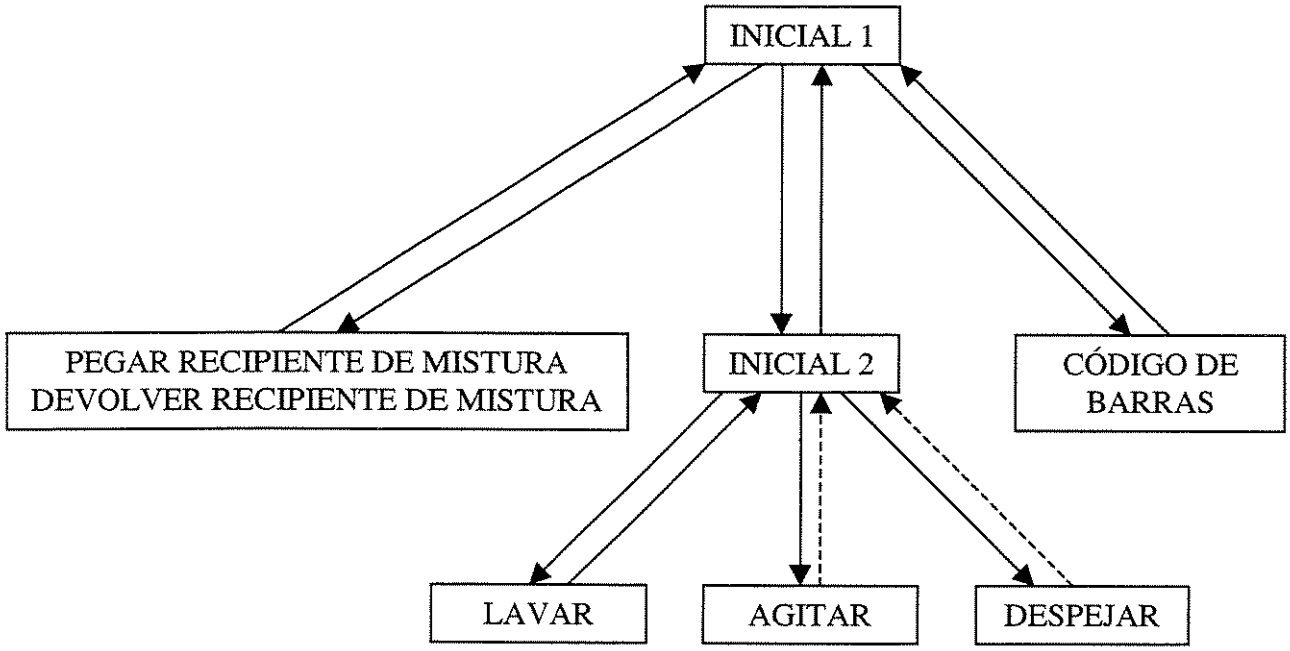


Fig. 4.9 - Estruturação de tarefas através de trajetórias para o posto de lavagem e mistura da tinta

4.2 Verificação dos programas do CLP

Esta etapa foi dividida em várias sub-etapas. Inicialmente foi realizado um teste para verificação das entradas e saídas do CLP. Em seguida, utilizando-se botões, onde cada botão comandará as saídas correspondente as trajetórias do ROBIX™.

Os programas do CLP foram gerados nas três formas de programação Ladder, SFC e lista de instruções, sendo que esta última foi gerada para que pudesse ser utilizada uma das versões do software do CLP.

Nas tabelas a seguir estão descritas as entradas – 4.1a e saídas – 4.1b utilizadas. O CLP adotado é um Mitsubishi, modelo FX0N,

variáveis	in	relê	descrição
SENS	X0	5-24V	sensor de produto - leitor código de barras
AUXA1	X1	5-24V	entrada auxiliar B - ROBIX1
AUXB1	X2	5-24V	entrada auxiliar A – ROBIX1
AUXA2	X3	5-24V	entrada auxiliar B – ROBIX2
AUXB2	X4	5-24V	entrada auxiliar A – ROBIX2
SENS_MESA	X5		sensor de posicionamento de parada - mesa rotativa
SENS_MESA	X6		sensor de posicionamento de parada - mesa rotativa
START	X7		entrada digital 0 – teclado
ENT1	X10		entrada digital 1 – teclado
ENT2	X11		entrada digital 2 – teclado
ENT3	X12		entrada digital 3 – teclado
ENT4	X13		entrada digital 4 – teclado
EMERG	X14		entrada digital 5 – teclado

Tab. 4.1 a - Entradas

variáveis	out	relê	descrição
COR1	Y0	24-5V	acionamento da bomba cor 1
COR2	Y1	24-5V	acionamento da bomba cor 2
COR3	Y2	24-5V	acionamento da bomba cor 3
COR4	Y3	24-5V	acionamento da bomba cor 4
MISTURA	Y4	24-5V	acionamento do misturador
LAVAGEM	Y5	24-5V	acionamento da bomba de lavagem
MOT_H	Y6	24-5V	acionamento do motor mesa rotativa horário
MOT_AH	Y7	24-5V	acionamento do motor mesa rotativa anti-horário
SAI_ROB21	Y10	24-5V	saída 1 robix2
SAI_ROB22	Y11	24-5V	saída 2 robix2
SAI_ROB23	Y12	24-5V	saída 3 robix2
SAI_ROB11	Y13	24-5V	saída 1 robix1
SAI_ROB12	Y14	24-5V	saída 2 robix1
SAI_ROB13	Y15	24-5V	saída 3 robix1

Tab. 4.1b - Saídas

4.2.1 Verificação da comunicação entre o CLP com o PC

Nesta etapa, foi verificado somente se a programação em C comunica-se com CLP comandando as saídas e entradas. Neste teste, além de verificar a comunicação entre os dois componentes, teve como finalidade verificar se as mensagens programadas para aparecer no monitor do computador estavam ocorrendo. Estas mensagens têm como função indicar o movimento em execução ou o aguardo para o próximo movimento.

4.2.2 Verificação da comunicação entre o CLP e o manipulador robótico através do programa em C

Nesta etapa, foram verificadas várias trajetórias completas dos manipuladores robóticos, utilizando-se a programação em C e o CLP. Estes testes, estão descritos nos SFC's das figuras abaixo. Para cada trajetória considerada completa, foi realizado um programa no CLP.

A figura 4.10 apresenta a seqüência de movimentos realizadas para todas as cores que o manipulador robótico do posto de escolha de cores realiza.

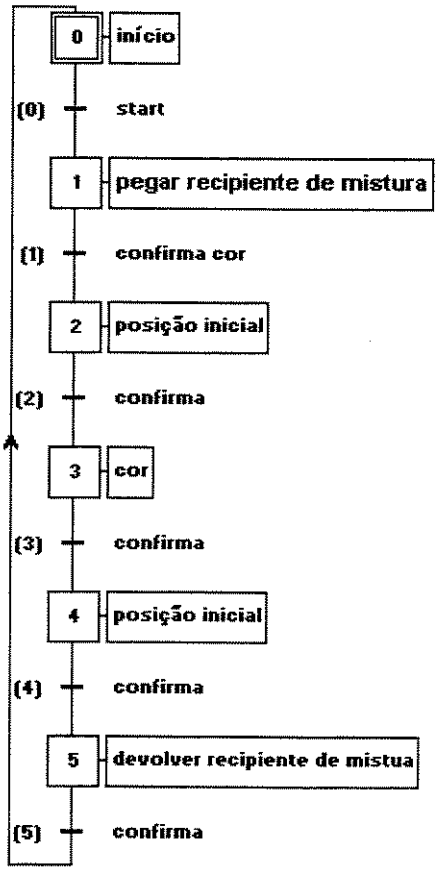


Fig. 4.10 - SFC funcional das trajetórias das cores

Cada tarefa de cores, é ativada através do acionamento de um botão. E uma vez ativado todas as trajetórias envolvidas serão buscadas através do programa em C, na seqüência determinada através do programa desenvolvido na CLP.

Uma vez realizada a verificação com o manipulador robótico do posto de escolha de cores, foram realizados os testes com o manipulador do posto de lavagem e mistura da tinta, este sendo responsável pela mistura, lavagem do recipiente de mistura e despejo da tinta , além do armazenamento dos recipientes vazios contendo os códigos de barras.

A figura 4.11 apresenta o SFC funcional do ciclo mixer, a etapa agitador e despejar, é uma macro etapa possuindo portanto, sub-etapas necessárias para que a tarefa possa ser completada e respeitada a estruturação das trajetórias conforme demonstrado no item 4.1.

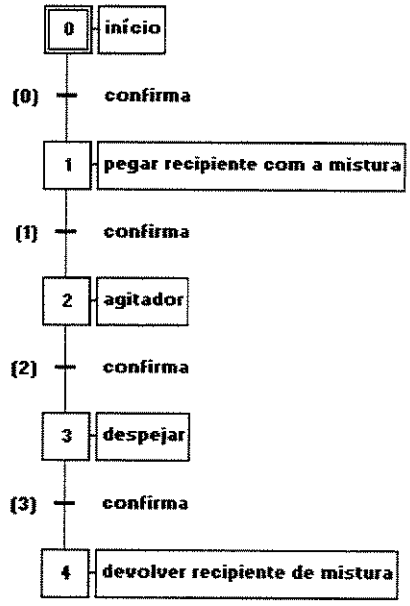


Fig. 4.11 - SFC funcional do ciclo mixer

A figura 4.12 apresenta o SFC funcional do ciclo de lavagem, a etapa agitador e despejar, é uma macro etapa possuindo portanto, sub-etapas necessárias para que a trajetória possa ser completada e respeitada a estruturação das trajetórias conforme demonstrado no item 4.1.

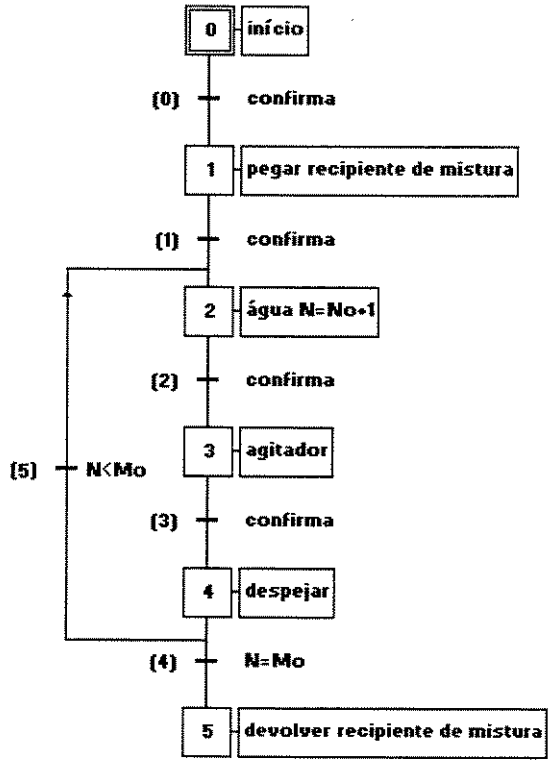


Fig. 4.12 - SFC funcional do ciclo de lavagem

A figura 4.13 apresenta o ciclo do código de barras, em que o manipulador robótico deverá pegar o recipiente vazio codificado da área de alimentação e posicioná-lo no local de depósito.

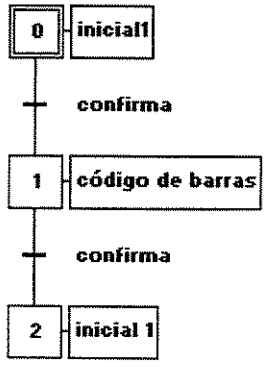


Fig. 4.13 - SFC funcional do ciclo código de barras

Após estes testes, foram realizados testes envolvendo o acionamento das bombas de cores, água e misturador, independentemente, através da emissão de sinais de saídas provenienetes do programa do manipulador robótico.

A figura 4.14, apresenta o acionamento da bomba com a sequencia de trajetórias realizadas pelo manipulador robótico.

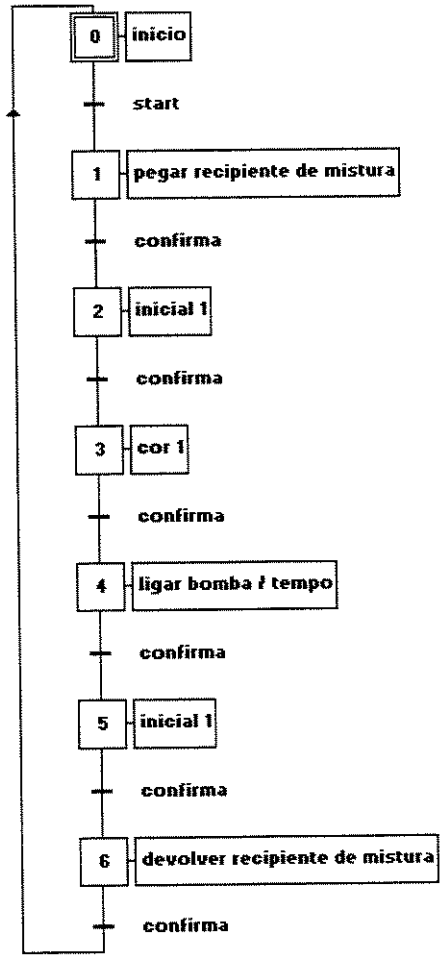


Fig. 4.14 - SFC funcional ciclo de mistura com bomba

A movimentação da mesa rotativa foi verificada diretamente e utilizando-se o CLP, juntamente com os sensores. A sequência de movimentos realizados nesta etapa estão demonstrados no SFC da figura 4.15.

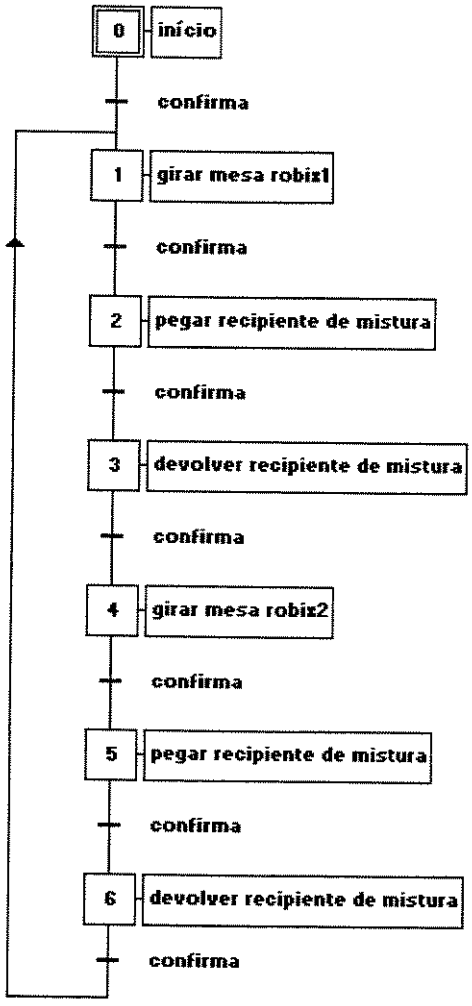


Fig. 4.15 - SFC funcional da mesa rotativa com os manipulador robótico

4.3 Integração final

Nesta etapa, todos os elementos estão integrados, e funcionando simultaneamente. Sendo que a cor a ser produzida pode ser determinada de duas formas: através do código de barras ou manualmente através de botões. O código de barras determinará além da cor a ser produzida, o funcionamento do sistema, como também a utilização simultânea de ambos os manipuladores robóticos.

4.4 Conclusão

Neste capítulo, verificou-se que para a implementação completa de um sistema automatizado é necessário que este esteja estruturado e documentado de forma a impedir a compreensão ambígua. Além disto, para que o sistema seja completamente integrado, é necessário que sejam determinadas as etapas a serem cumpridas, dividindo-se o conjunto todo em subconjuntos a serem testados individualmente.

Capítulo 5

Conclusões e Perspectivas Futuras

Neste trabalho, foram abordados os elementos básicos de um sistema automatizado de produção, envolvendo desde a sua descrição até a sua implementação. Verifica-se portanto, que mais importante que saber implementar tal sistema, é saber definir suas tarefas e documentá-los. Dentre as formas existentes para se descrever um sistema, adotamos o sistema seqüencial (*SFC – Sequential Function Chart*).

Todo o projeto e implementação de um Sistema Automatizado de Produção, foi realizado através da idealização de uma Plataforma Didática para Mistura de Cores, contendo uma Parte Comando com CLP, sensores e microcomputadores e uma Parte Operativa com manipuladores robóticos, mesa rotativa e atuadores diversos. A plataforma foi desenvolvida de forma a permitir a inclusão de novas tecnologias.

A complexidade do sistema automatizado foi possível, devido à sistematização e a estruturação adotada para o desenvolvimento do trabalho. As dificuldades de integração encontradas, foram na realização da comunicação entre as diferentes formas de programação / software adotado para cada elemento utilizado.

Visando os avanços tecnológicos, as redes de comunicação industriais tem cada vez mais apoiado os sistemas de automação e controle, isto devido à crescente complexidade dos processos industriais. Portanto, não tem sido implantados sistemas que não possuam algum forma de comunicação de dados. A busca de estruturas que garantam a segurança na transmissão dos dados, bem como na velocidade de comunicação faz com que haja o desenvolvimento de diferentes esquemas de comunicação de dados em ambientes industriais. Com isto, os sistemas de estrutura aberta, e considerando-se que alguns dispositivos produzem informações e outros consomem tais informações, podemos ter redes eficientes na maximização de troca de dados e também um aumento da flexibilidade da rede.

Portanto, como sugestões para trabalhos futuros, temos a implantação de um sistema supervisório na plataforma, uma vez que este é considerado dentro da estrutura de comunicação,, no nível de controle de processo, sendo responsável pela aquisição de dados do CLP para o computador, pela organização, utilização e gerenciamento dos dados.

Além deste trabalho são sugeridos também a implantação de uma esteira rolante como complementação da Parte Operativa da plataforma; a realização de pesquisas e implantação do Ensino a distância – EAD e a implantação de bancos de dados e gerenciamento da produção através do código de barras.

Referências Bibliográficas

Acar, M. Mechatronics Engineering Education in the UK. In: Joint Hungarian British International Mechatronics Conference, 1994, Budapeste. *Proceedings...* Budapest: Computational Mechatronics Publ., 1994, p 763- 769.

Araujo, Emerson dos Santos, *Modelagem e Descrição da Parte Comando de um Sistema Automatizado de Produção utilizando o GRAFCET - Aplicado à uma Plataforma Industrial em Automação*. Campinas: Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, 1997, 91 p. Tese (Mestrado

Asfahl, Ray C., *Robots and Manufacturing Automation* 2th USA: John Wiley & Sons, Inc., 1992, 487 p.

Bittar, R. C. S. M. *A Utilização do GRAFCET como Ferramenta na Automação Industrial.* Campinas: Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, 1993, 105 p. Tese (Mestrado)

Bolton, W. *Mechatronics - Electronic Control Systems in Mechanical Engineering*. England: Longman Scientific & Technical, 1995, 380 p.

Castrucci, Plínio B. L. *Controle Automático Teoria e Projeto* Editora da Universidade de São Paulo, 1969, 280 p.

Dalbó, R. F. *Simgraf: Um Ambiente Computacional para Simulação e Validação de Sistemas Automatizados de Produção Utilizando o GRAFCET.* Campinas: Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, 1994, 90 p. Tese (Mestrado)

Ferreira, Edson P., *Robótica Básica Modelagem de Robôs*, R. Vieira Gráfica e Editora Ltda. Versão Preliminar Publicada para a V Escola Brasileiro-Argentina de Informática, Rio de Janeiro, 1991.

Georgini, João Marcelo, *Elementos para Estruturação e Implementação de Sistemas Automatizados de Produção.* Campinas: Faculdade de Engenharia Mecânica, Universidade Estadual de Campinas, 1999. 209 p. Tese (Mestrado).

Groover, Mikell P. *Automation, Production Systems, and Computer Integrated Manufacturing* USA: Prentice-Hall International, Inc., 1987, 808 p.

Hunter, Ronald P. *Automated Process Control Systems - Concepts and Hardware.* 2nd edition, USA: Pratice-Hall, Inc. , 1987, 501 p

IEC - International Eletrotechnical Commission, Genebra. IEC-60848; *Preparation of Function Charts for Control Systems.* Genebra, 1988, 99p.

Kuo, B. C., *Automatic Control Systems* 4th ed. USA: Pratice & Hall, Inc., New Jersey, 1982, 720 p.

Miyagi, P. E., Barretto, M. R. P., Silva, J. R. *Domótica – Controle e Automação Volume 2* EDUSP, 1993, 180 p.

Santos, José J. Horta, *Automação Industrial* Rio de Janeiro: Livros Técnicos e Científicos, 1979, p. 10 - 23.

Vendrameto, O. *Bases de Conhecimento para a Automação da Manufatura.* São Paulo: Escola Politécnica - Engenharia de Produção, Universidade de São Paulo, 1994. 155 p. Tese (Doutorado).

Williams, D. J. *Manufacturing Systems an introduction to the technologies* 2nd edition, Chapman & Hall, London, UK, 1994, 246 p.

Wolovich, William A. *Automatic Control Systems - Basic Analysus and Design.* USA: International Edition, 1994, 450 p.

ANEXO I

SFC, Ladder e Lista de Instruções

A seguir são apresentados alguns Diagramas Funcionais Sequenciasis, os programas em Linguagem Ladder e as Lista de Instruções das células de trabalho da Plataforma.

As referências de sinais de saídas do CLP, utilizadas na implantação dos programas de aplicação para os manipuladores robóticos são demonstrados segundo as tabelas a seguir:

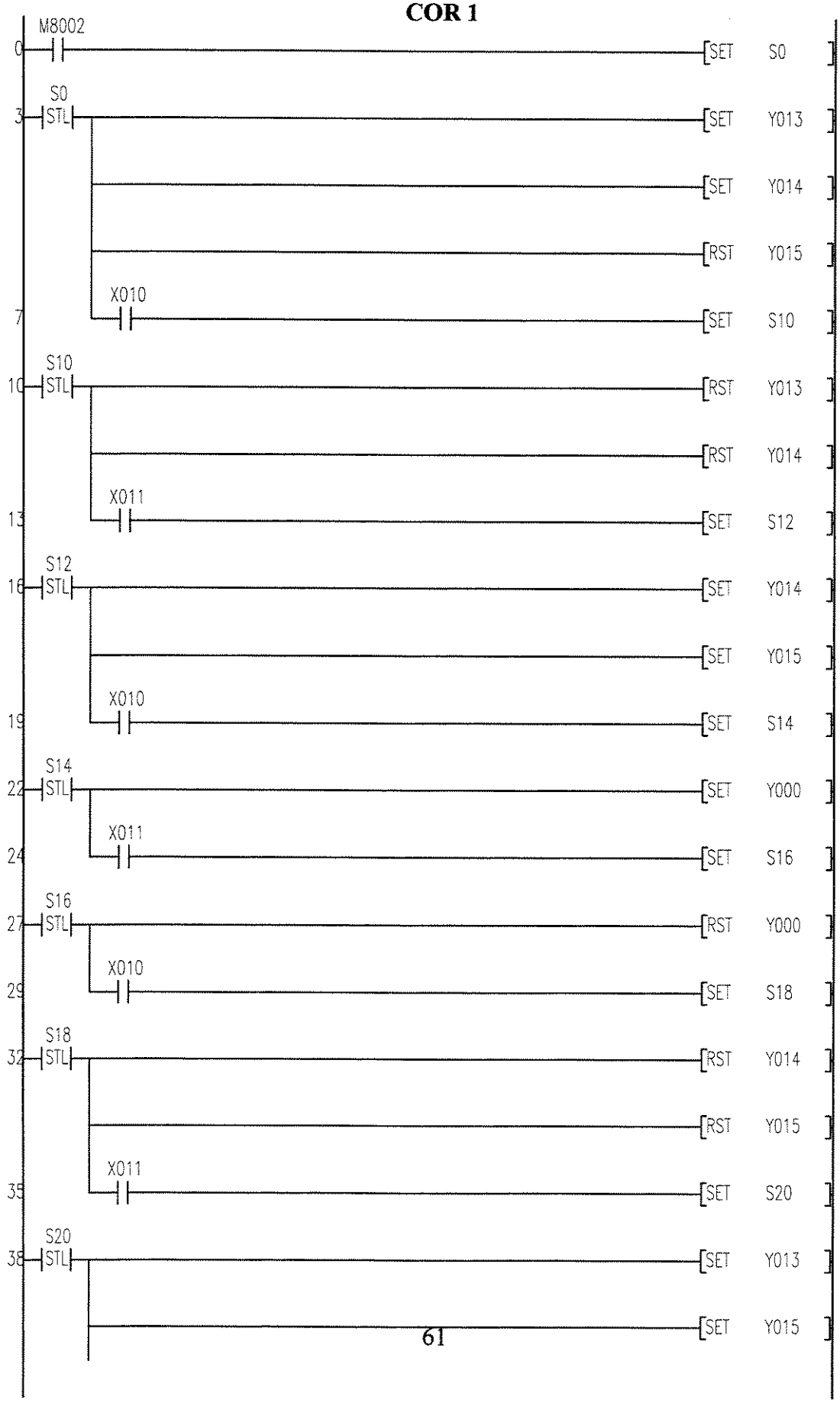
a) Célula robotizada para escolha de cores:

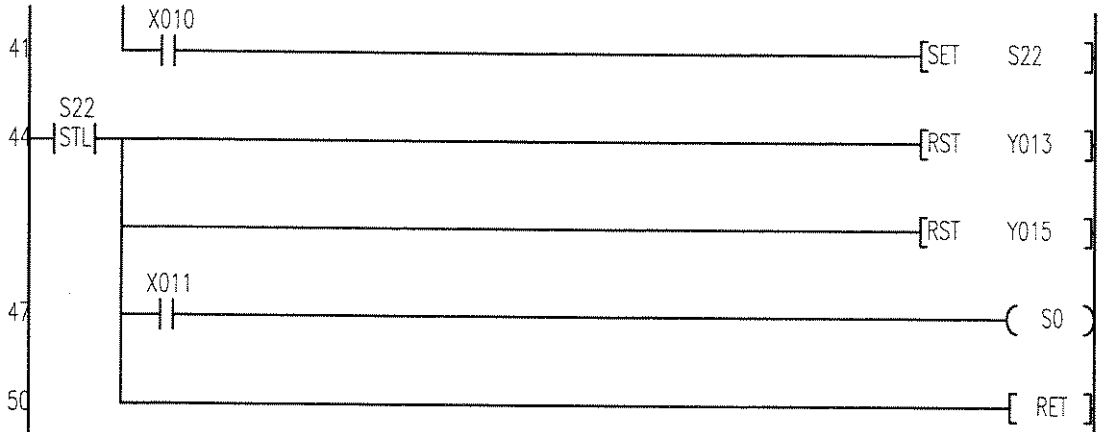
saídas				observações	
Y6	**Y7**	**Y10**	**Y11**	**Trajetoria**	**Descrição trajetória**
0	0	0	1	Traj10	Posição inicial 1
1	1	1	0	Traj11	Posição inicial 2
1	1	0	0	Traj12	Pegar recipiente de mistura
1	0	1	0	Traj13	Devolver recipiente de mistura
0	1	1	0	Traj14	Cor 1
0	0	1	0	Traj15	Cor 2
0	1	0	0	Traj16	Cor 3
1	0	0	0	Traj17	Cor 4
0	0	0	0	Traj18	Posição de espera / restart

b) Célula robotizada para lavagem e mistura de tintas:

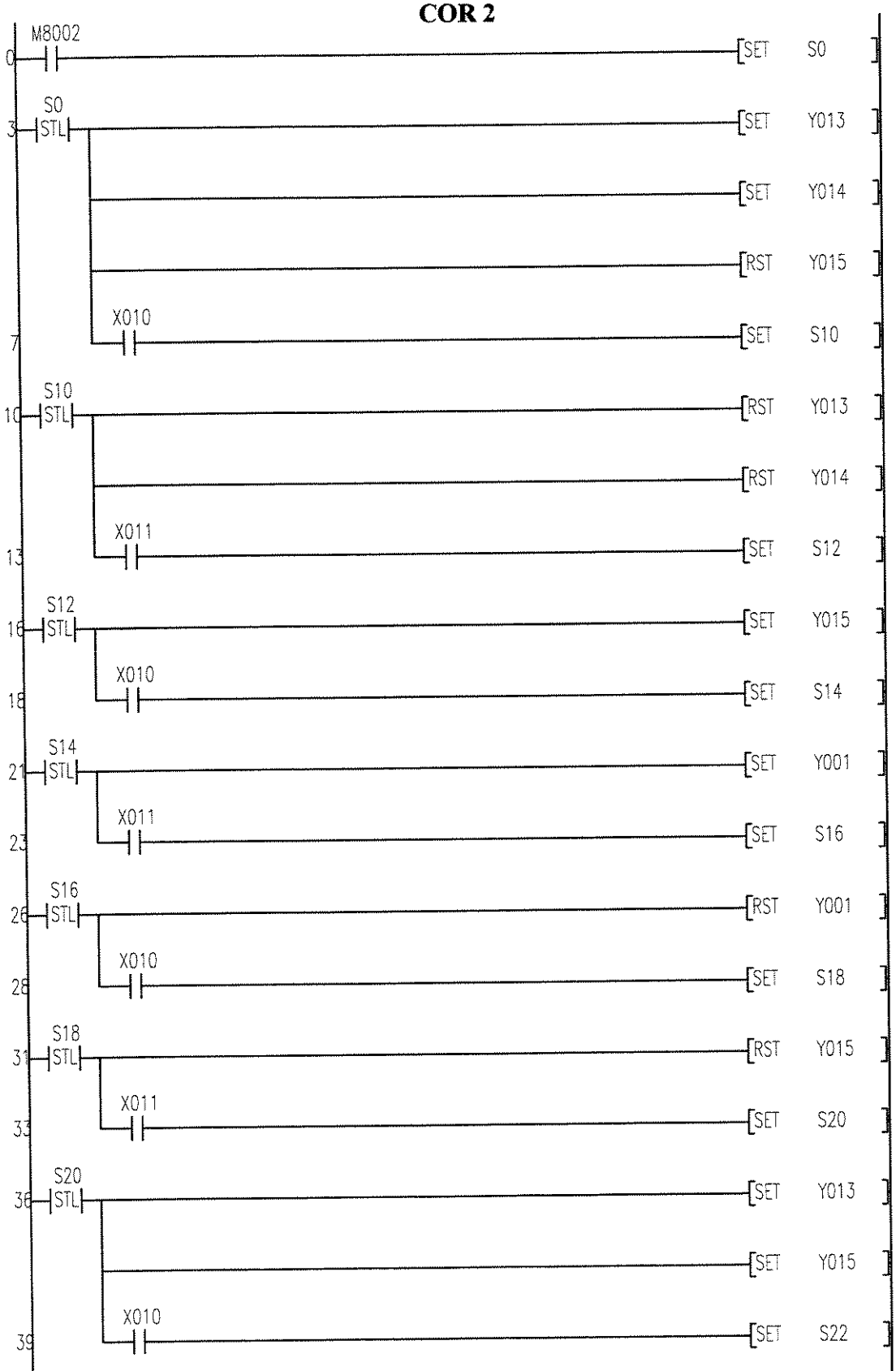
saídas				observações	
Y12	**Y13**	**Y14**	**Y15**	**Trajetórias**	**Descrição trajetórias**
0	0	0	1	Traj20	Posição inicial 1
1	1	1	0	Traj21	Posição inicial 2
1	1	0	0	Traj22	Pegar recipiente de mistura
1	0	1	0	Traj23	Devolver recipiente de mistura
0	1	1	0	Traj24	Lavagem com água
0	0	1	0	Traj25	Misturador de cores
0	1	0	0	Traj26	Despejar após lavagem
1	0	0	0	Traj27	Código de barras
0	0	0	0	Traj28	Posição de espera/ restart

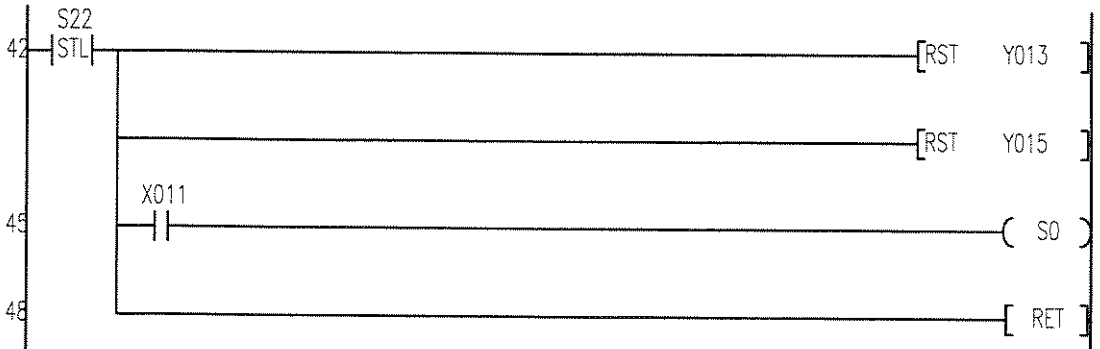
COR 1


M8002
0
SET S0
S0
STL
3
SET Y013
SET Y014
RST Y015
X010
7
SET S10
S10
STL
10
RST Y013
RST Y014
X011
13
SET S12
S12
STL
16
SET Y014
SET Y015
X010
19
SET S14
S14
STL
22
SET Y000
X011
24
SET S16
S16
STL
27
RST Y000
X010
29
SET S18
S18
STL
32
RST Y014
RST Y015
X011
35
SET S20
S20
STL
38
SET Y013
SET Y015

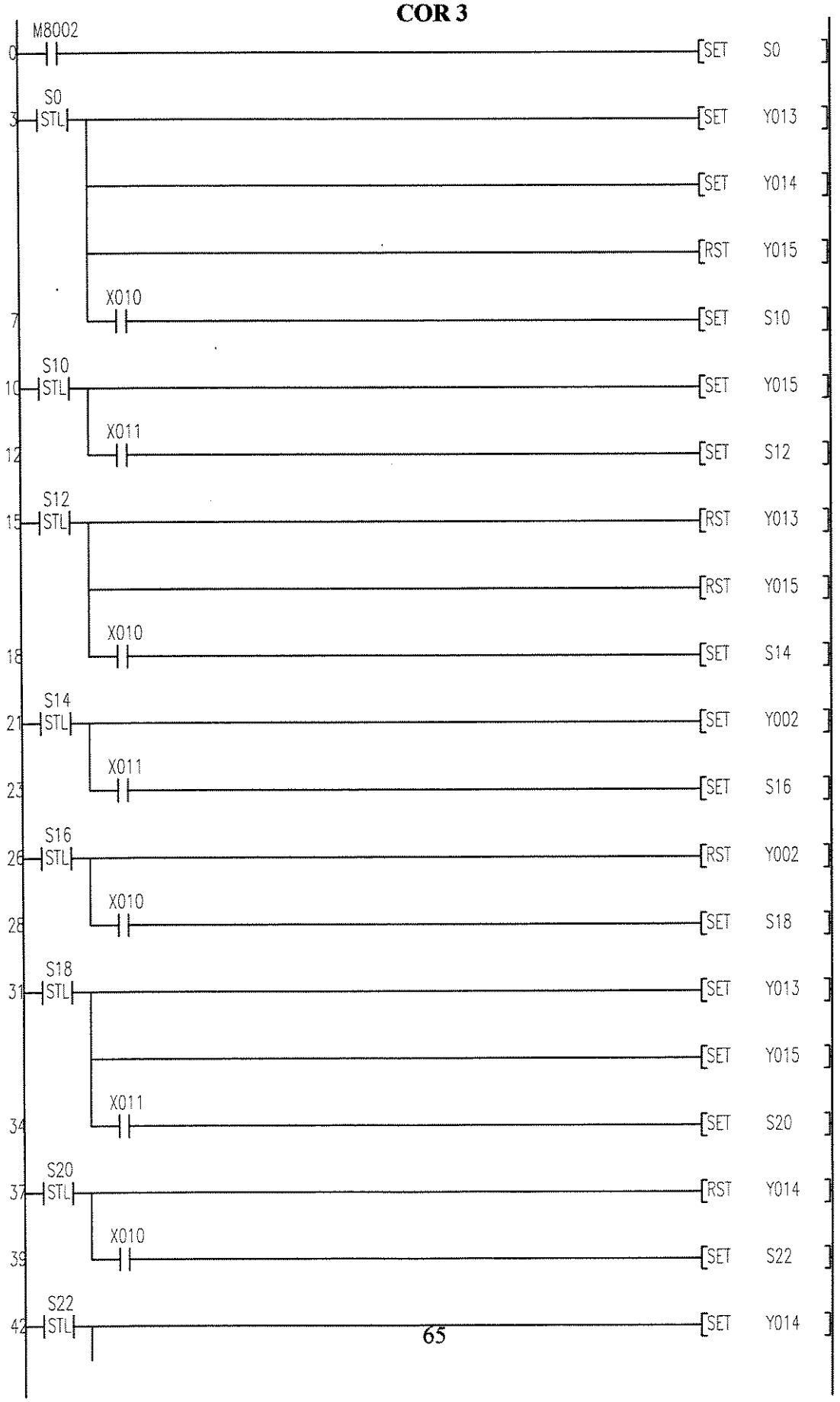

41
X010
SET S22
44
S22
STL
RST Y013
RST Y015
47
X011
S0
50
RET

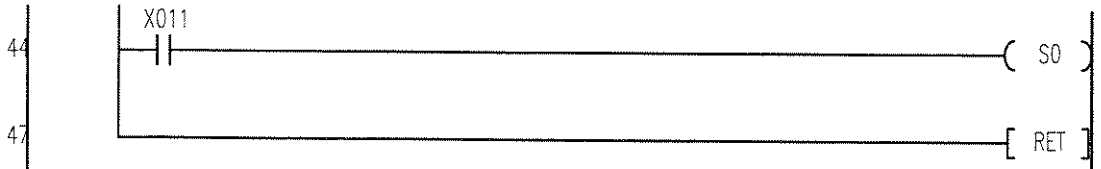
COR 2


M8002
0
SET S0
S0
STL
3
SET Y013
SET Y014
RST Y015
X010
7
SET S10
S10
STL
10
RST Y013
RST Y014
X011
13
SET S12
S12
STL
16
SET Y015
X010
18
SET S14
S14
STL
21
SET Y001
X011
23
SET S16
S16
STL
26
RST Y001
X010
28
SET S18
S18
STL
31
RST Y015
X011
33
SET S20
S20
STL
36
SET Y013
SET Y015
X010
39
SET S22

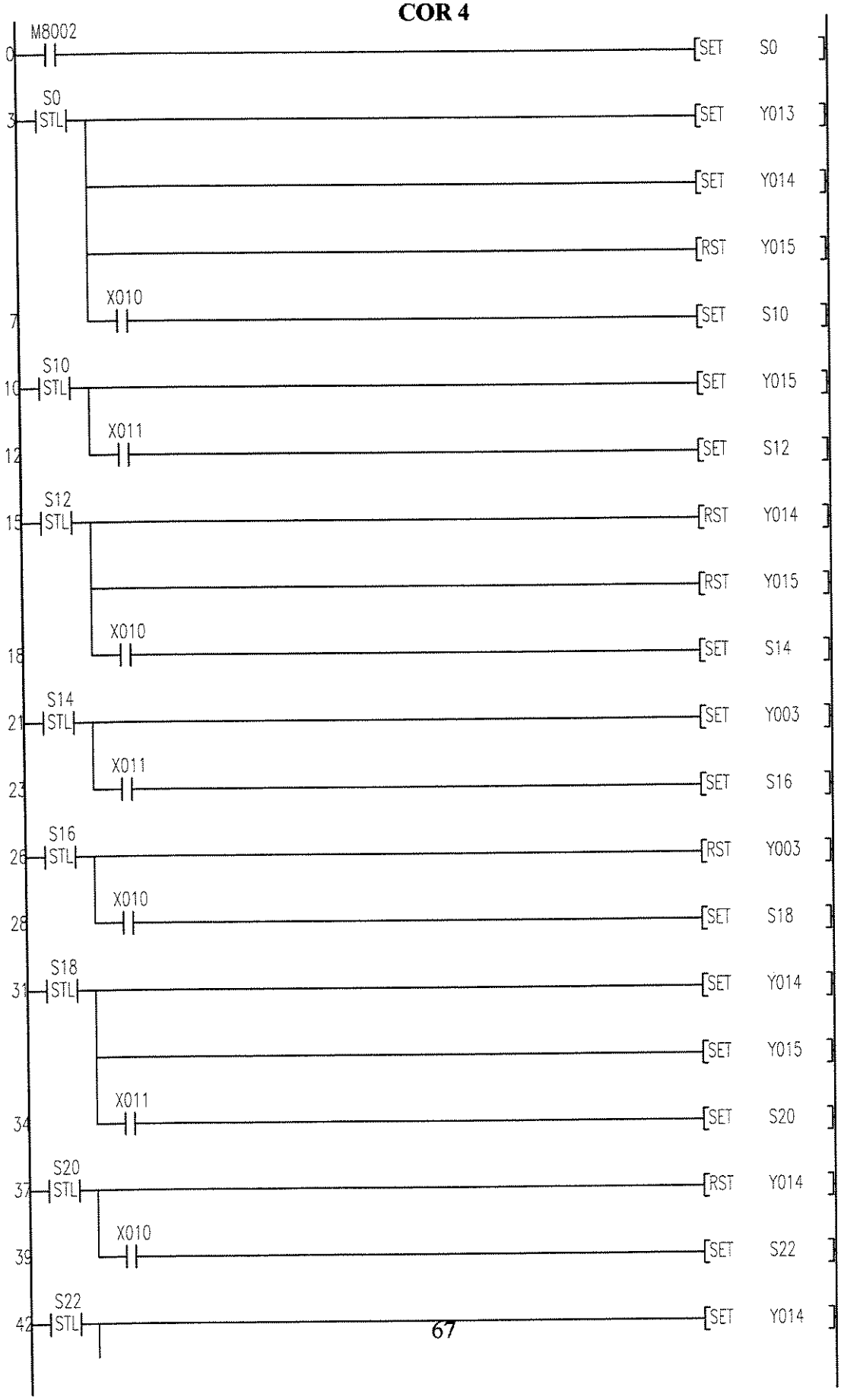

42
S22
STL
RST Y013
RST Y015
X011
49
S0
48
RET

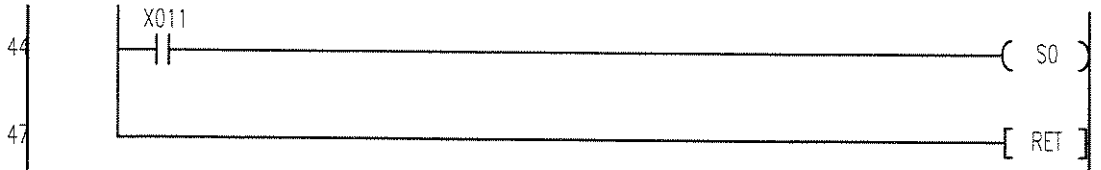
COR 3


M8002
0
SET S0
S0
STL
3
SET Y013
SET Y014
RST Y015
X010
7
SET S10
S10
STL
10
SET Y015
X011
12
SET S12
S12
STL
15
RST Y013
RST Y015
X010
18
SET S14
S14
STL
21
SET Y002
X011
23
SET S16
S16
STL
26
RST Y002
X010
28
SET S18
S18
STL
31
SET Y013
SET Y015
X011
34
SET S20
S20
STL
37
RST Y014
X010
39
SET S22
S22
STL
42
SET Y014

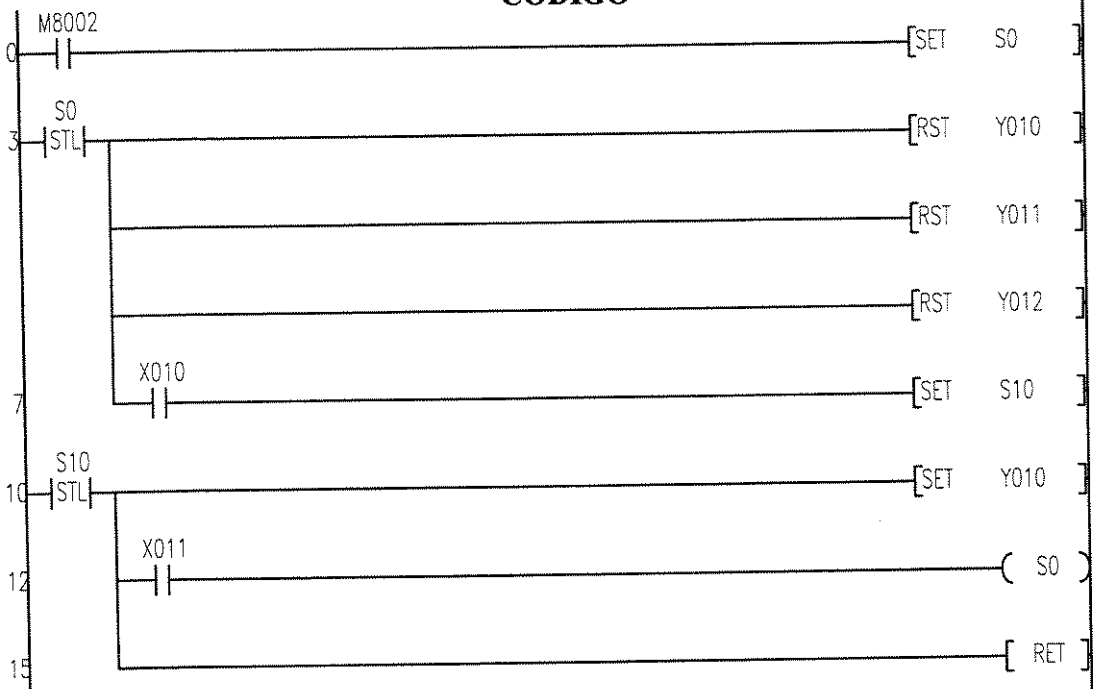

X011
44
47
SO
RET

COR 4

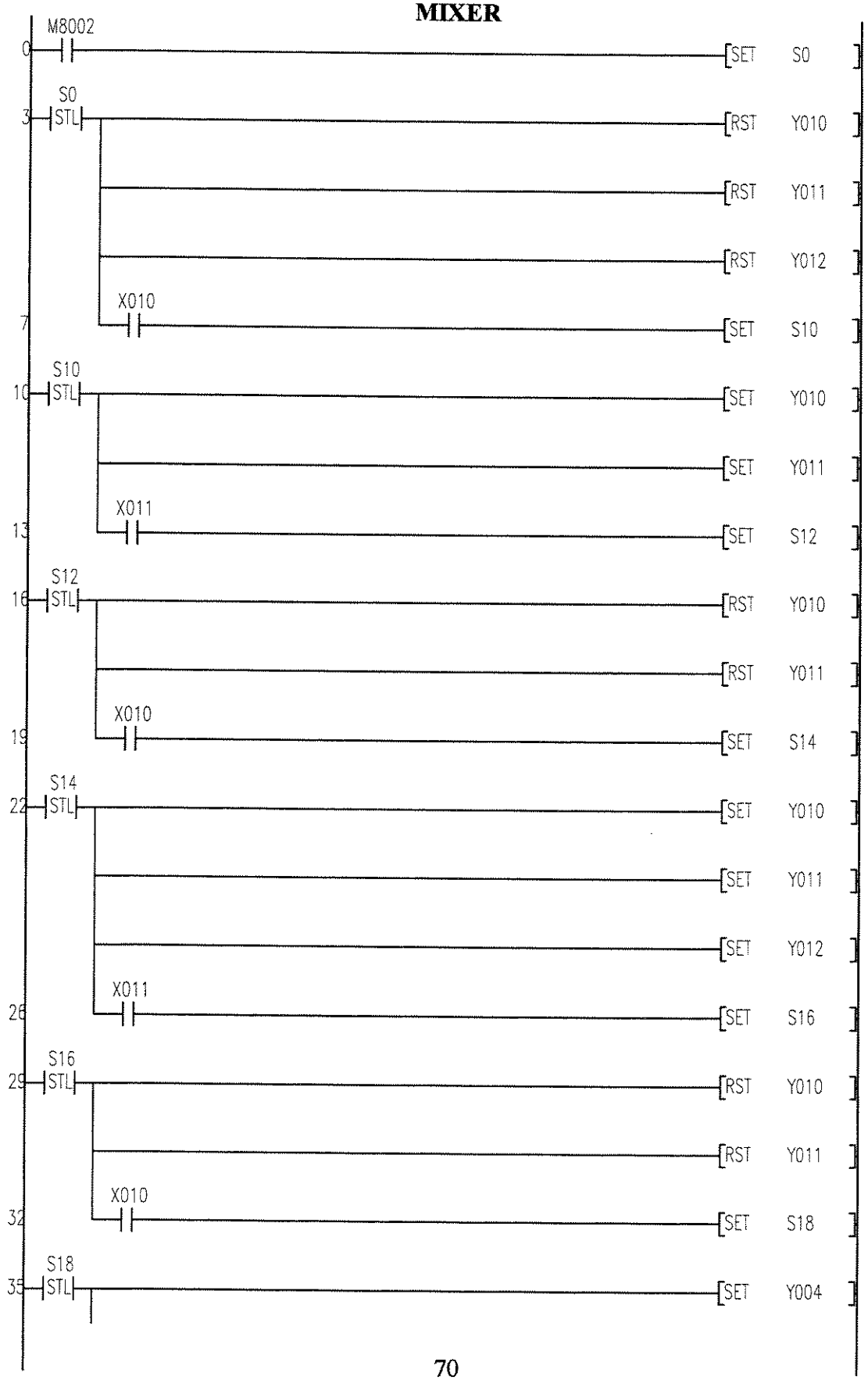

M8002
0
SET S0
S0
STL
3
SET Y013
SET Y014
RST Y015
X010
7
SET S10
S10
STL
10
SET Y015
X011
12
SET S12
S12
STL
15
RST Y014
RST Y015
X010
18
SET S14
S14
STL
21
SET Y003
X011
23
SET S16
S16
STL
26
RST Y003
X010
28
SET S18
S18
STL
31
SET Y014
SET Y015
X011
34
SET S20
S20
STL
37
RST Y014
X010
39
SET S22
S22
STL
42
SET Y014


X011
44
SO
47
RET

CÓDIGO

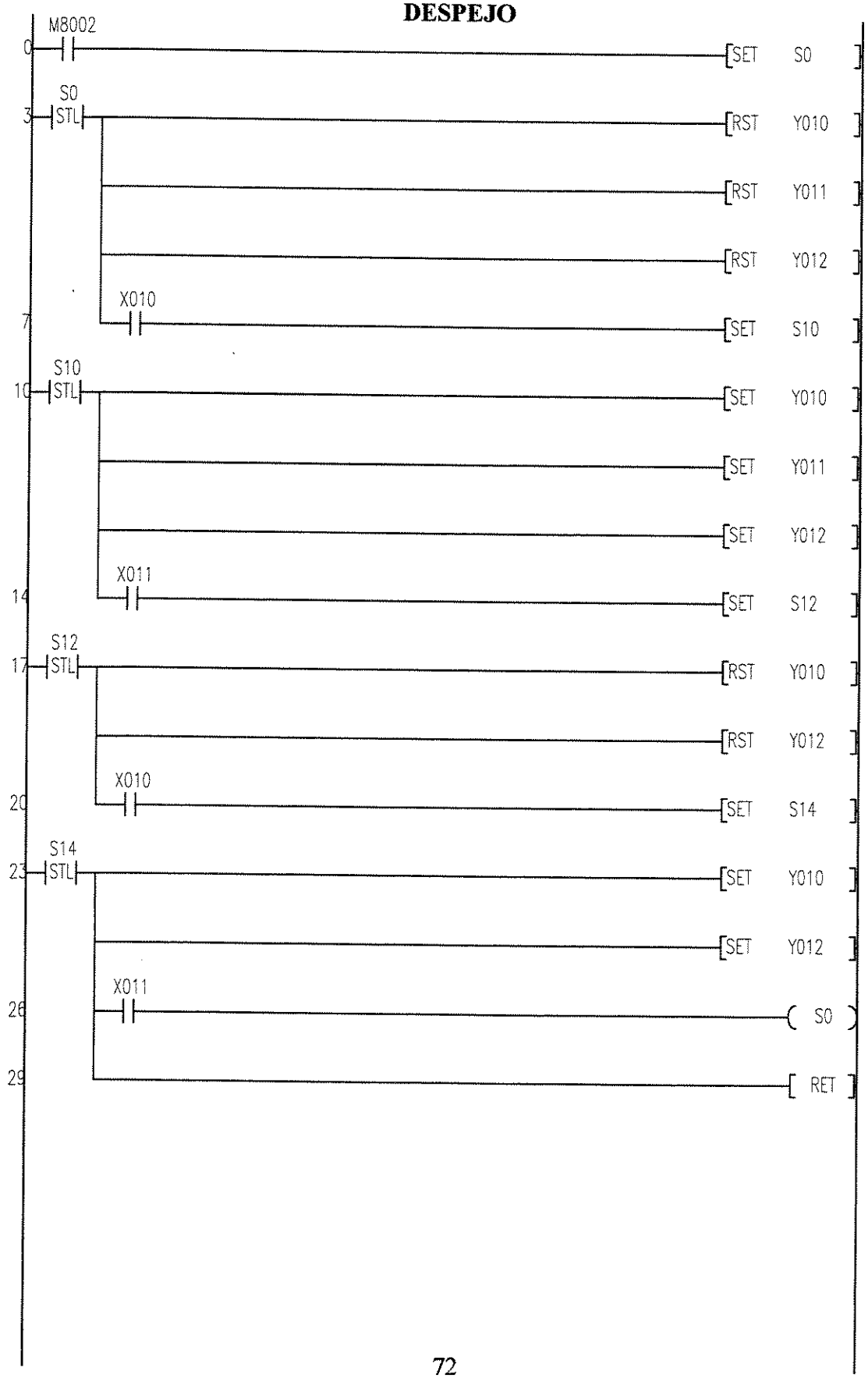

M8002
SET S0
S0
STL
RST Y010
RST Y011
RST Y012
X010
SET S10
S10
STL
SET Y010
X011
S0
RET

MIXER

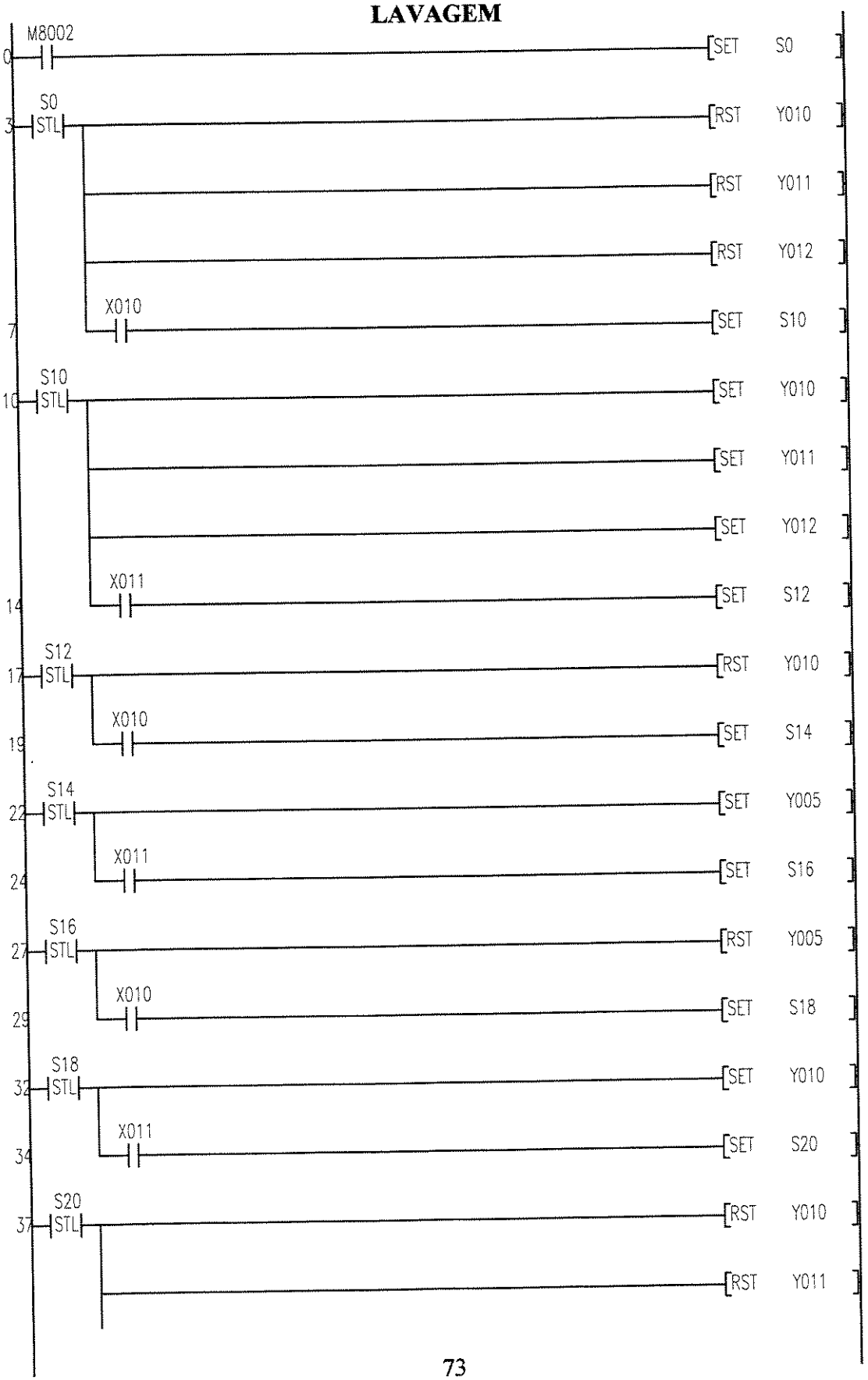

M8002
0
SET S0
S0
STL
3
RST Y010
RST Y011
RST Y012
X010
7
SET S10
S10
STL
10
SET Y010
SET Y011
X011
13
SET S12
S12
STL
16
RST Y010
RST Y011
X010
19
SET S14
S14
STL
22
SET Y010
SET Y011
SET Y012
X011
26
SET S16
S16
STL
29
RST Y010
RST Y011
X010
32
SET S18
S18
STL
35
SET Y004

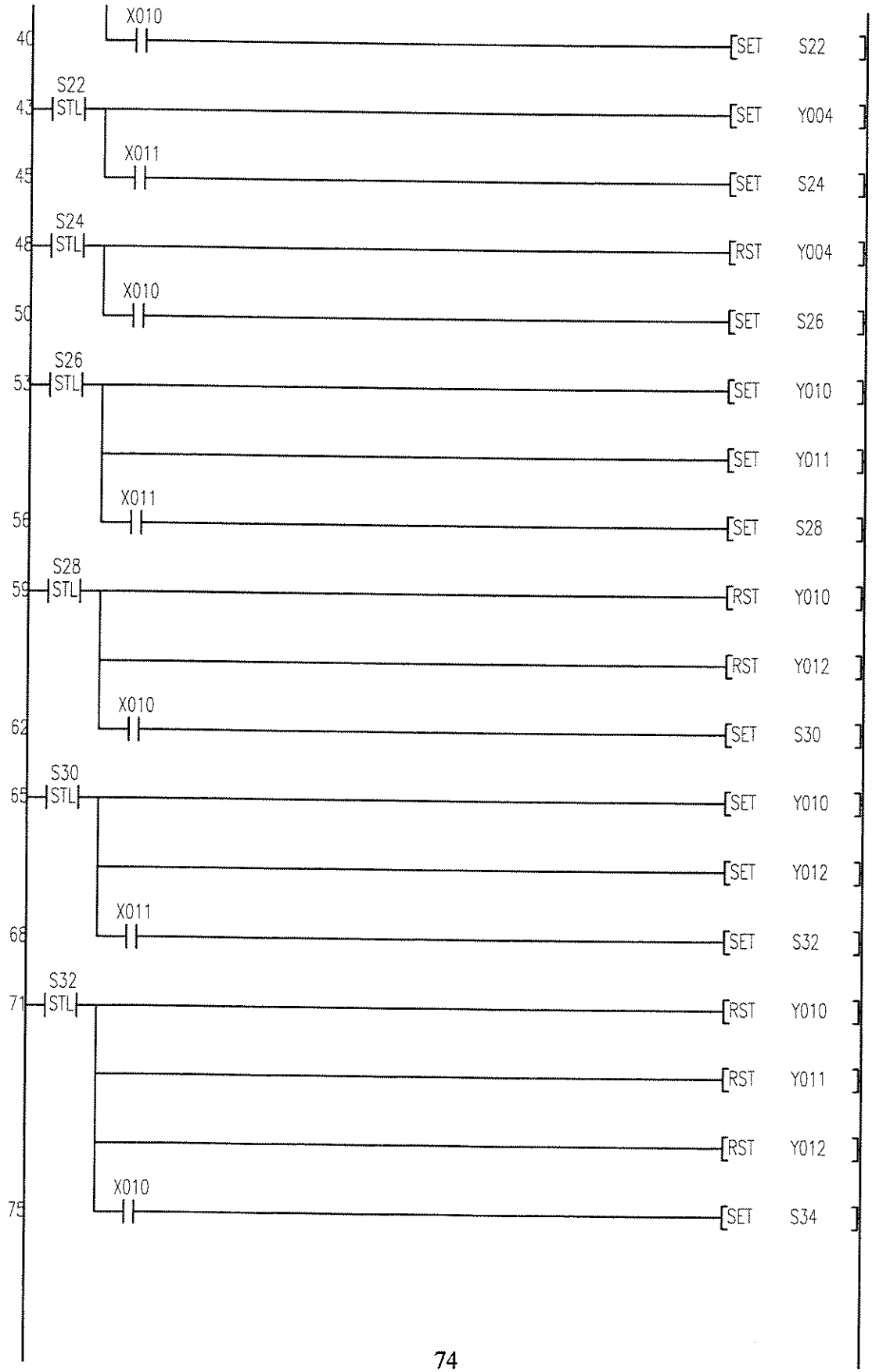

37
X011
SET S20
40
S20
STL
RST Y004
42
X010
SET S22
45
S22
STL
SET Y010
SET Y011
48
X011
S0
51
RET

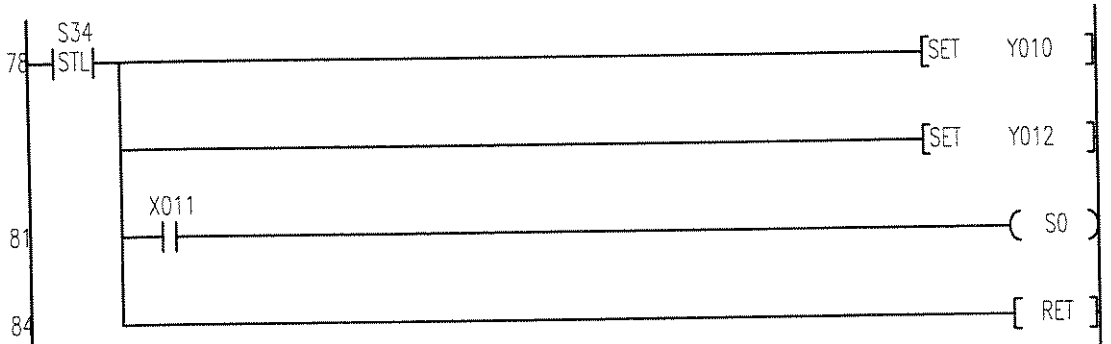
DESPEJO


M8002
0
SET S0
S0
STL
3
RST Y010
RST Y011
RST Y012
X010
7
SET S10
S10
STL
10
SET Y010
SET Y011
SET Y012
X011
14
SET S12
S12
STL
17
RST Y010
RST Y012
X010
20
SET S14
S14
STL
23
SET Y010
SET Y012
X011
26
S0
29
RET

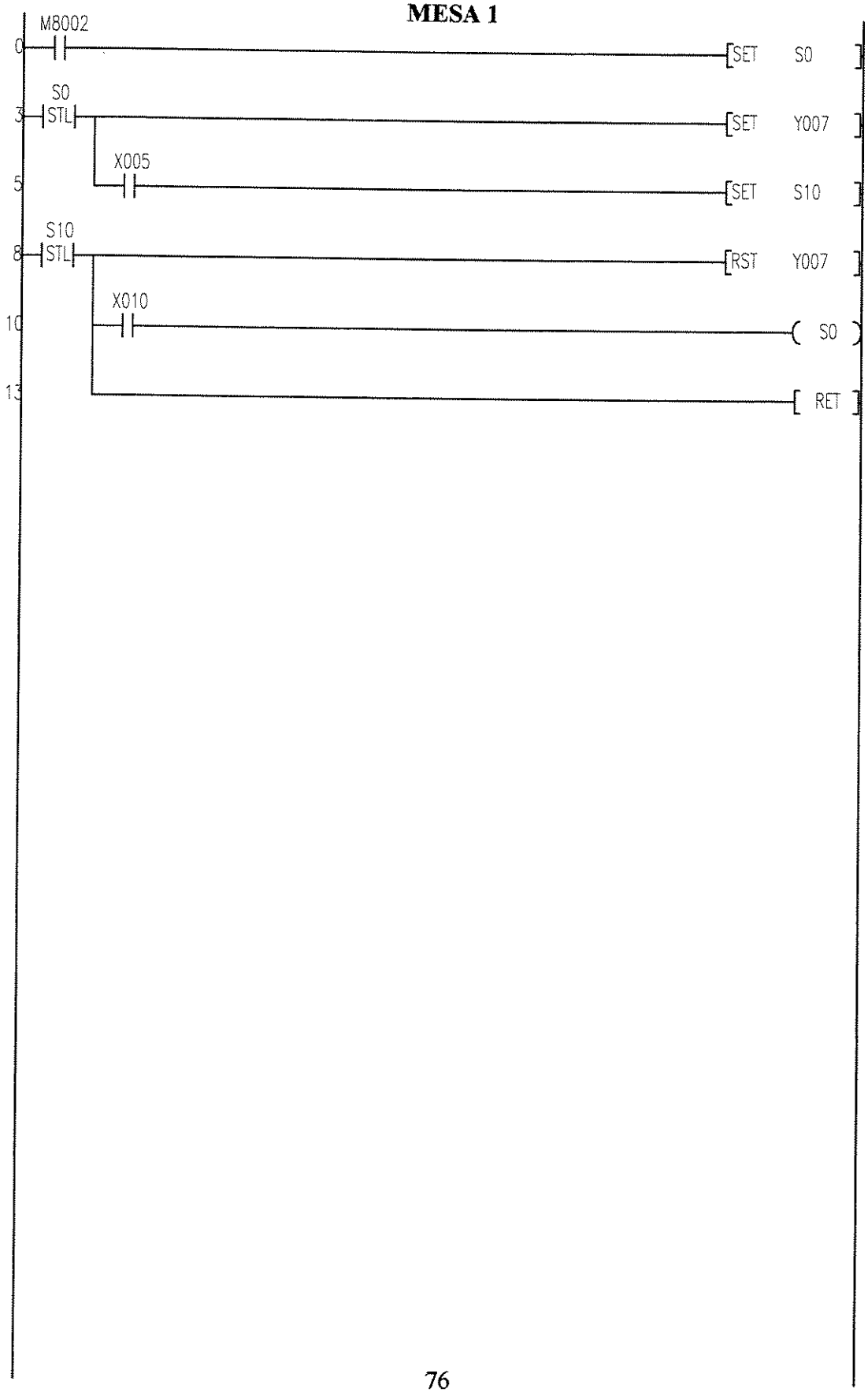
LAVAGEM


M8002
SET S0
S0
STL
RST Y010
RST Y011
RST Y012
X010
SET S10
S10
STL
SET Y010
SET Y011
SET Y012
X011
SET S12
S12
STL
RST Y010
X010
SET S14
S14
STL
SET Y005
X011
SET S16
S16
STL
RST Y005
X010
SET S18
S18
STL
SET Y010
X011
SET S20
S20
STL
RST Y010
RST Y011

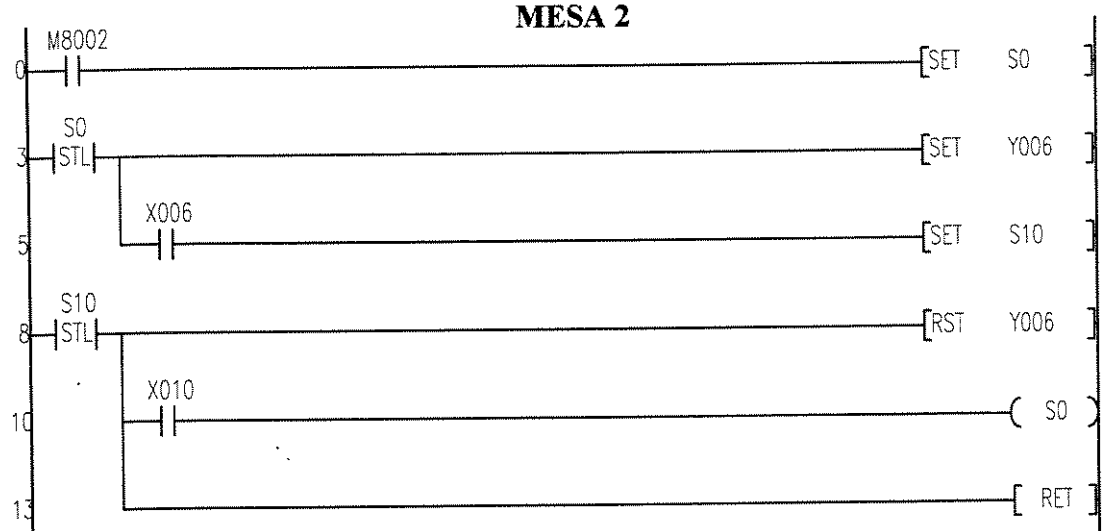

40
X010
SET S22
43
S22
STL
SET Y004
45
X011
SET S24
48
S24
STL
RST Y004
50
X010
SET S26
53
S26
STL
SET Y010
SET Y011
56
X011
SET S28
59
S28
STL
RST Y010
RST Y012
62
X010
SET S30
65
S30
STL
SET Y010
SET Y012
68
X011
SET S32
71
S32
STL
RST Y010
RST Y011
RST Y012
75
X010
SET S34


S34
STL
SET Y010
SET Y012
X011
S0
RET

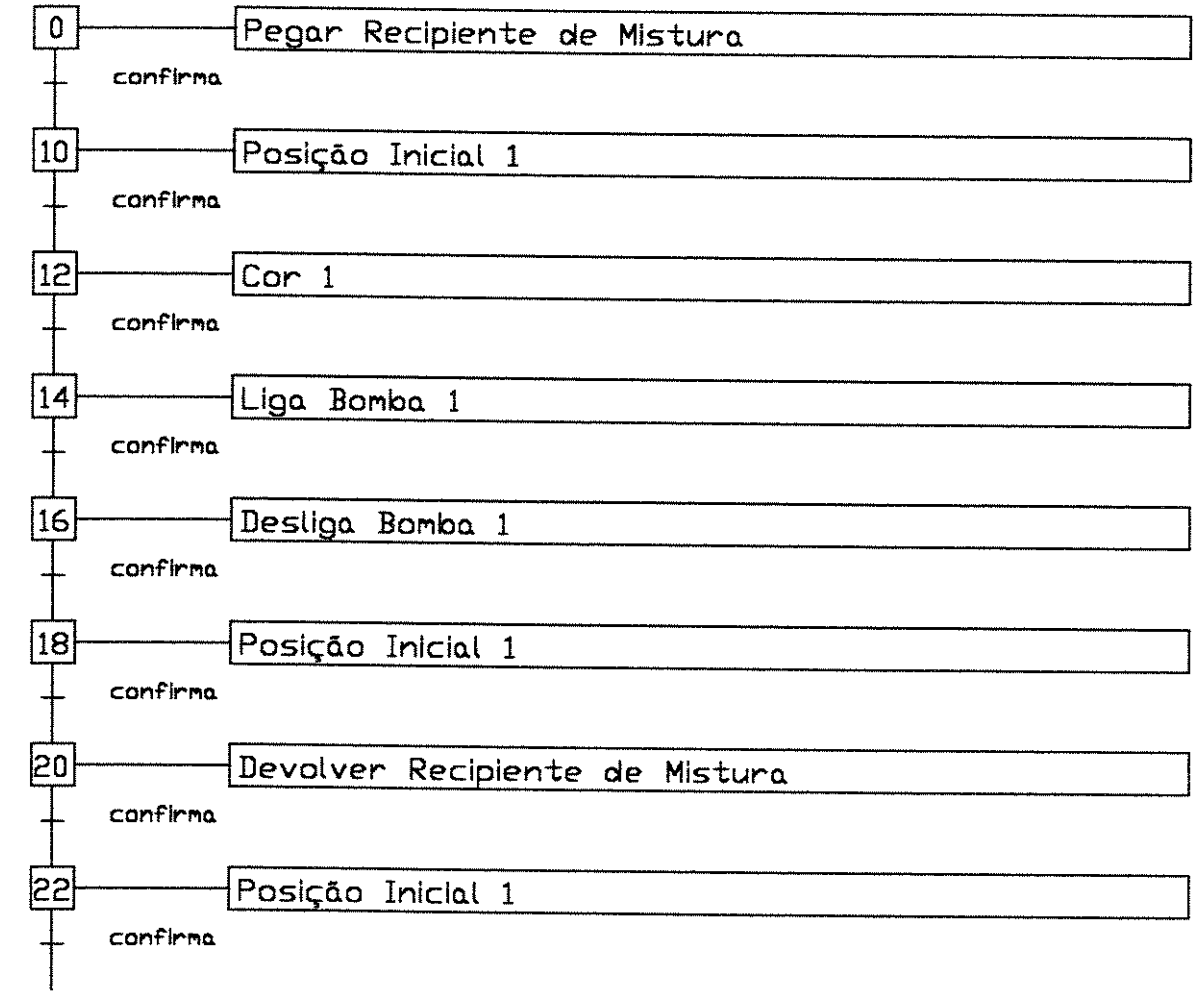
MESA 1


M8002
SET S0
S0
STL
SET Y007
X005
SET S10
S10
STL
RST Y007
X010
S0
RET

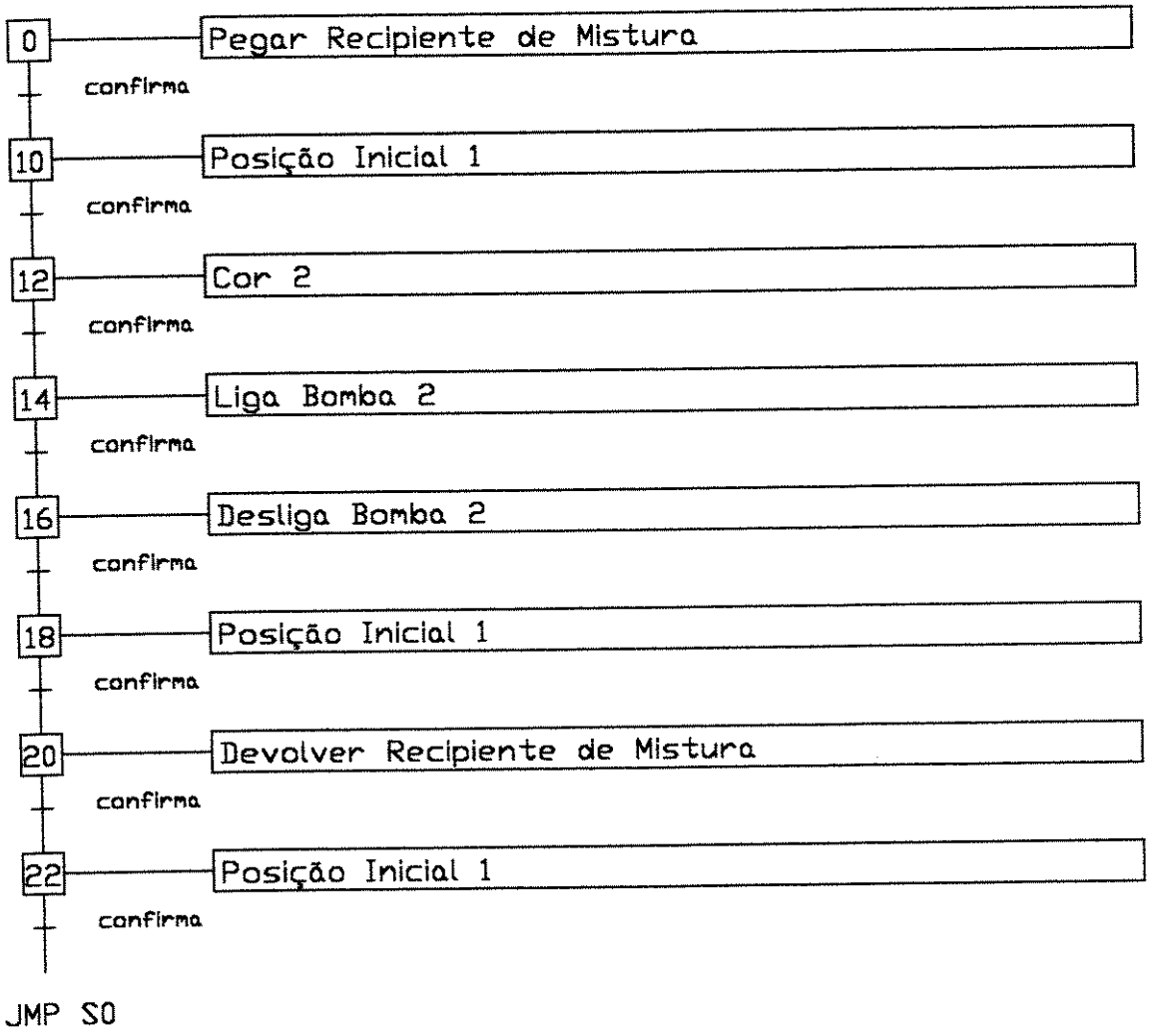
MESA 2


M8002
SET S0
S0
STL
SET Y006
X006
SET S10
S10
STL
RST Y006
X010
S0
RET

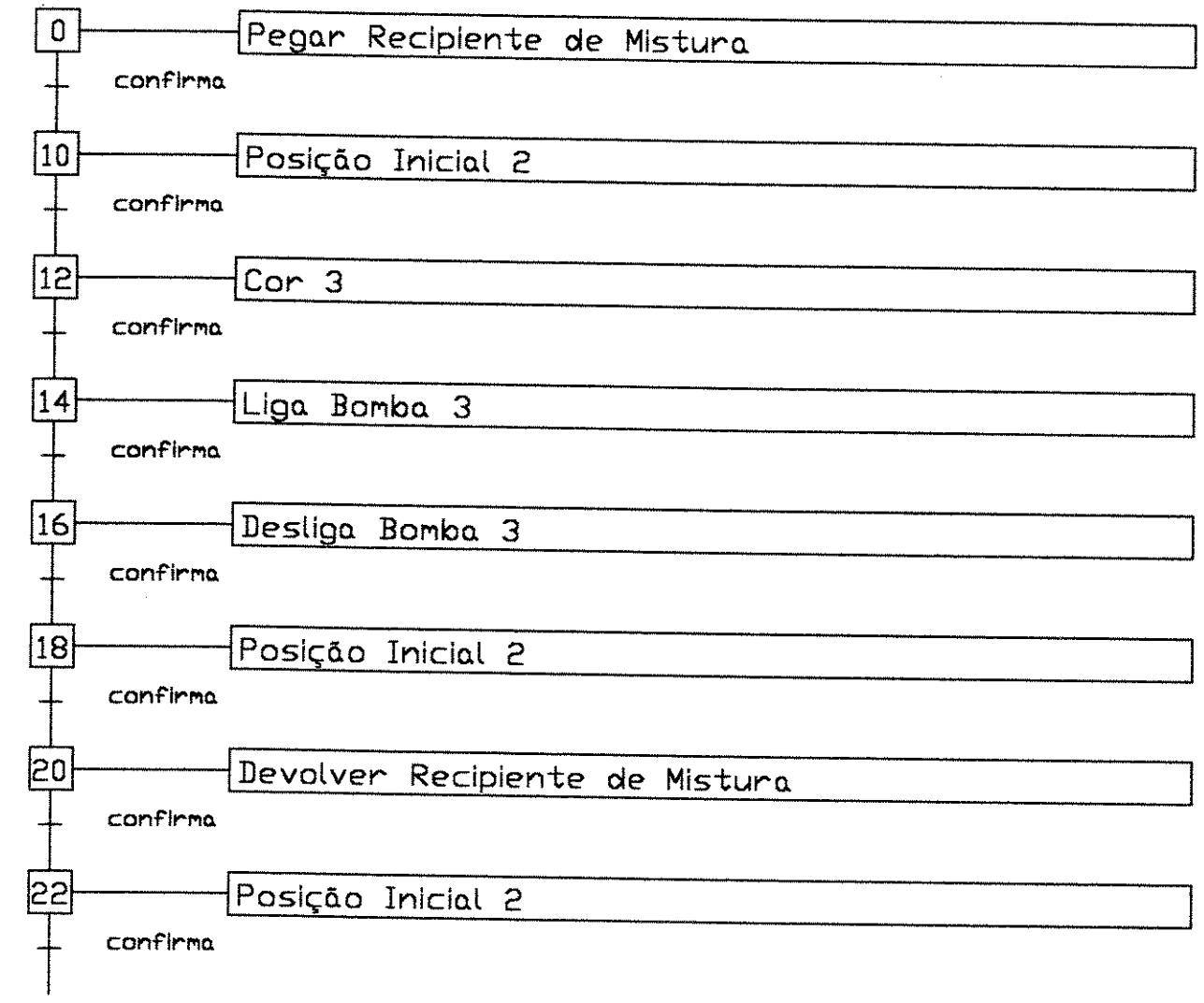
COR - 1


0
Pegar Recipiente de Mistura
confirma
10
Posição Inicial 1
confirma
12
Cor 1
confirma
14
Liga Bomba 1
confirma
16
Desliga Bomba 1
confirma
18
Posição Inicial 1
confirma
20
Devolver Recipiente de Mistura
confirma
22
Posição Inicial 1
confirma
JMP S0

COR - 2

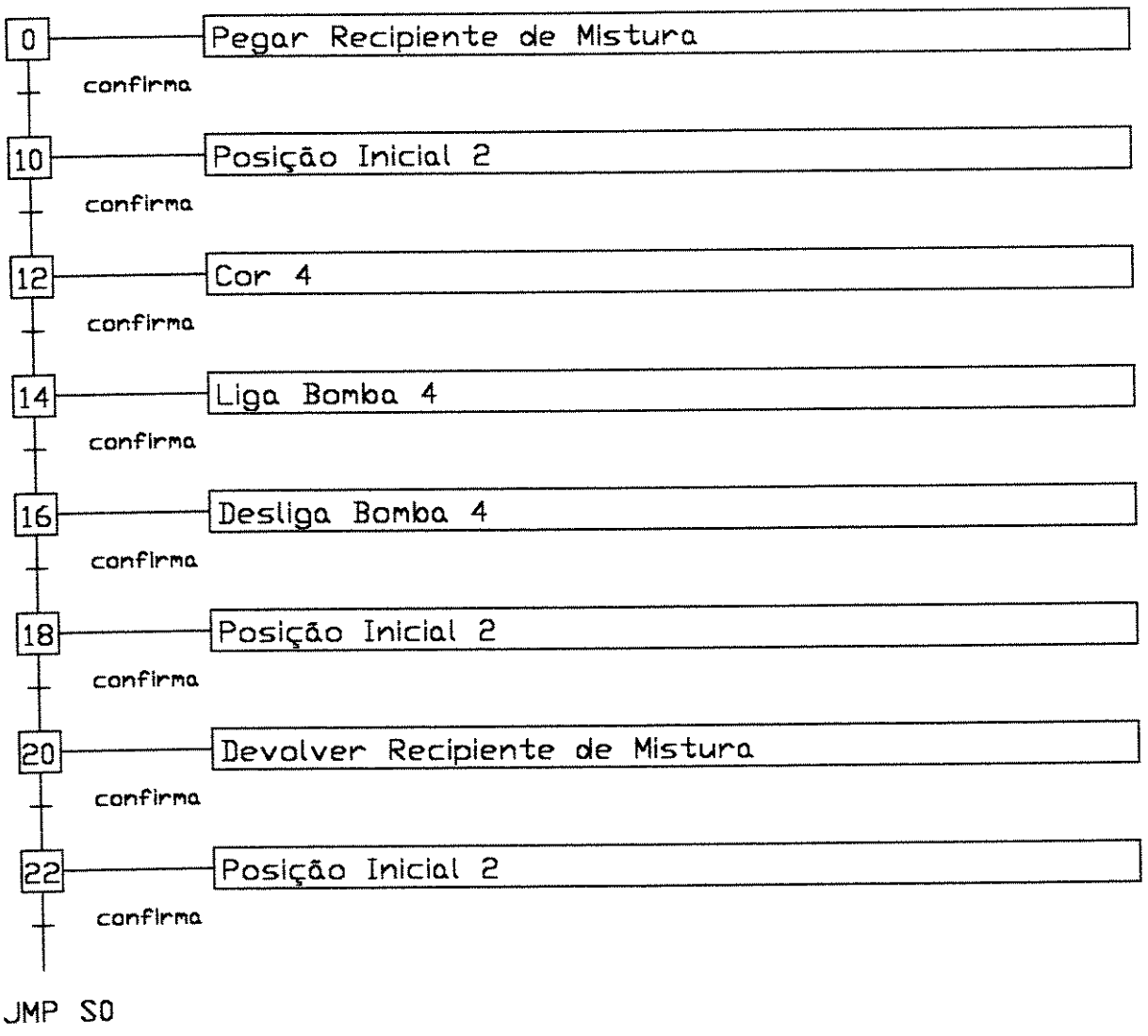

0
Pegar Recipiente de Mistura
confirma
10
Posição Inicial 1
confirma
12
Cor 2
confirma
14
Liga Bomba 2
confirma
16
Desliga Bomba 2
confirma
18
Posição Inicial 1
confirma
20
Devolver Recipiente de Mistura
confirma
22
Posição Inicial 1
confirma
JMP S0

COR - 3

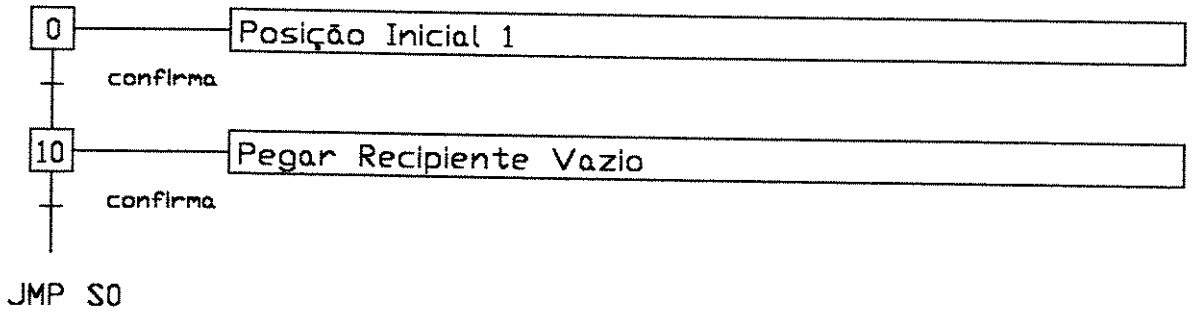

0
Pegar Recipiente de Mistura
confirma
10
Posição Inicial 2
confirma
12
Cor 3
confirma
14
Liga Bomba 3
confirma
16
Desliga Bomba 3
confirma
18
Posição Inicial 2
confirma
20
Devolver Recipiente de Mistura
confirma
22
Posição Inicial 2
confirma

JMP S0

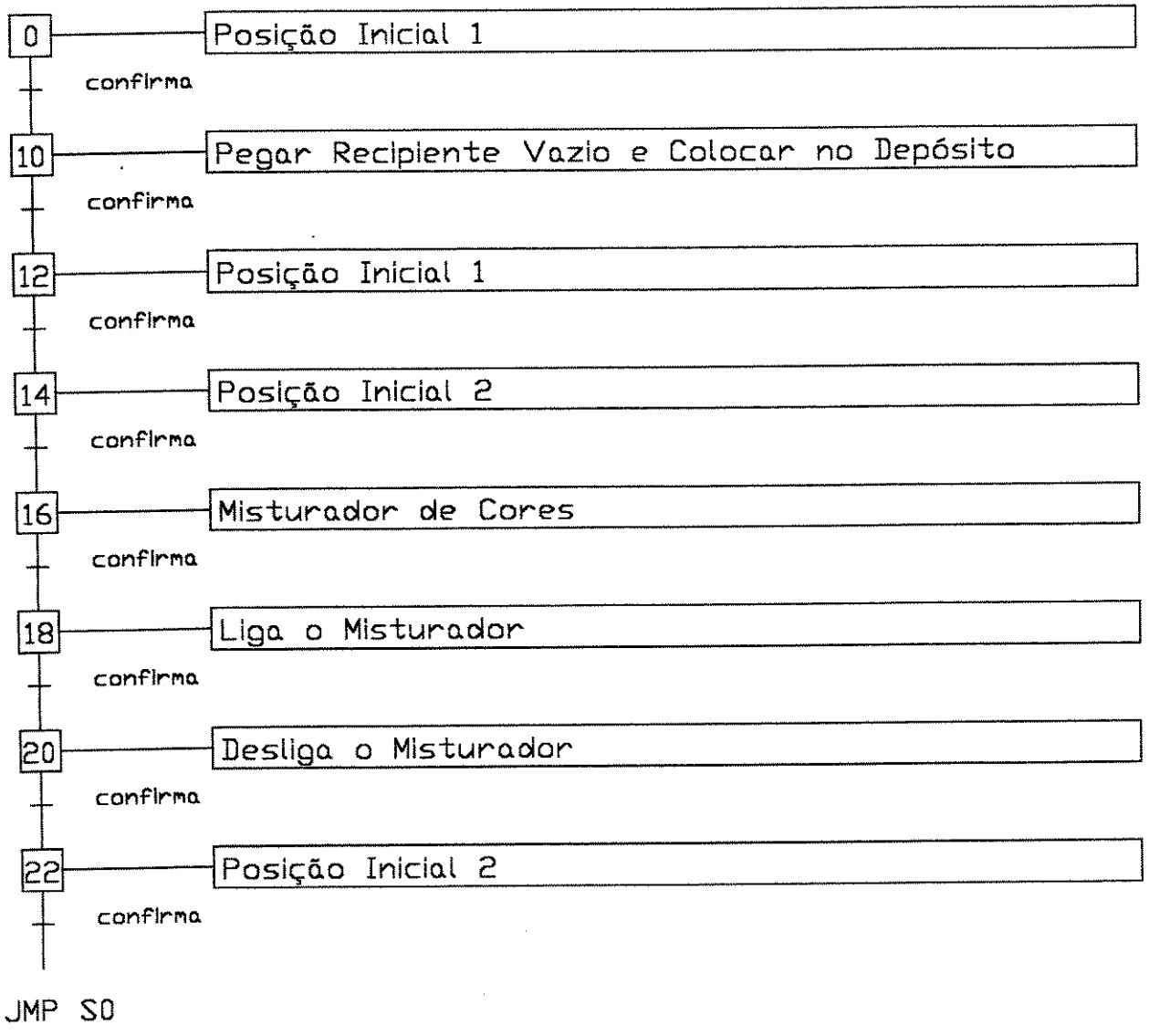
COR - 4


0
Pegar Recipiente de Mistura
confirma
10
Posição Inicial 2
confirma
12
Cor 4
confirma
14
Liga Bomba 4
confirma
16
Desliga Bomba 4
confirma
18
Posição Inicial 2
confirma
20
Devolver Recipiente de Mistura
confirma
22
Posição Inicial 2
confirma
JMP S0

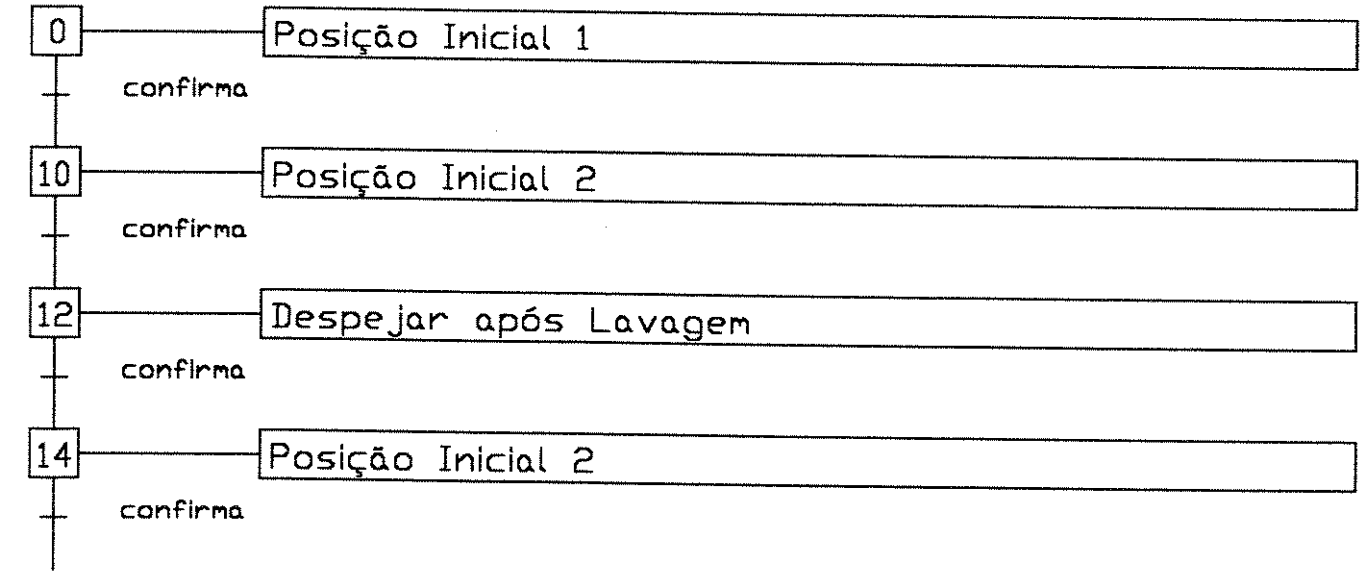
CÓDIGO


0
Posição Inicial 1
confirma
10
Pegar Recipiente Vazio
confirma
JMP S0

MIXER

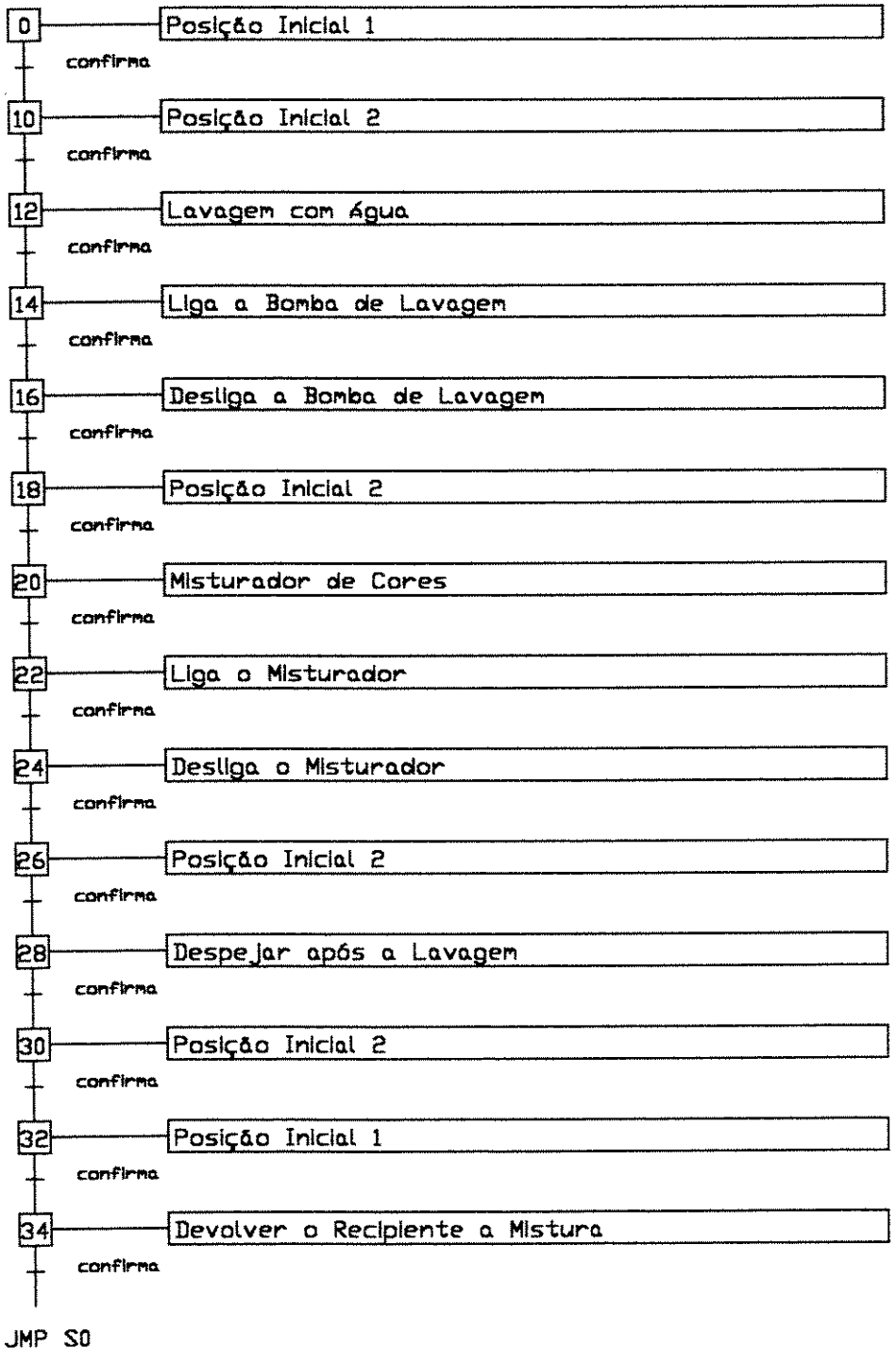

0
Posição Inicial 1
confirma
10
Pegar Recipiente Vazio e Colocar no Depósito
confirma
12
Posição Inicial 1
confirma
14
Posição Inicial 2
confirma
16
Misturador de Cores
confirma
18
Liga o Misturador
confirma
20
Desliga o Misturador
confirma
22
Posição Inicial 2
confirma
JMP S0

DESPEJO

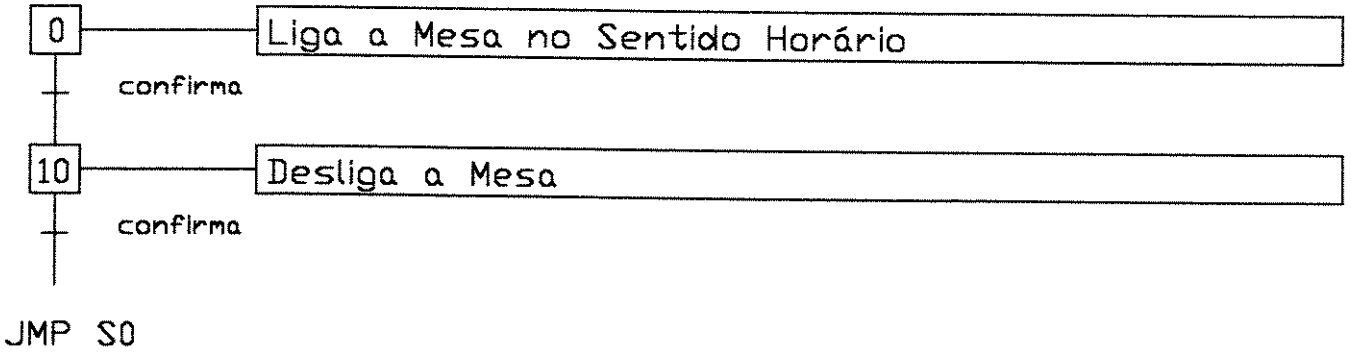


JMP S0

LAVAGEM

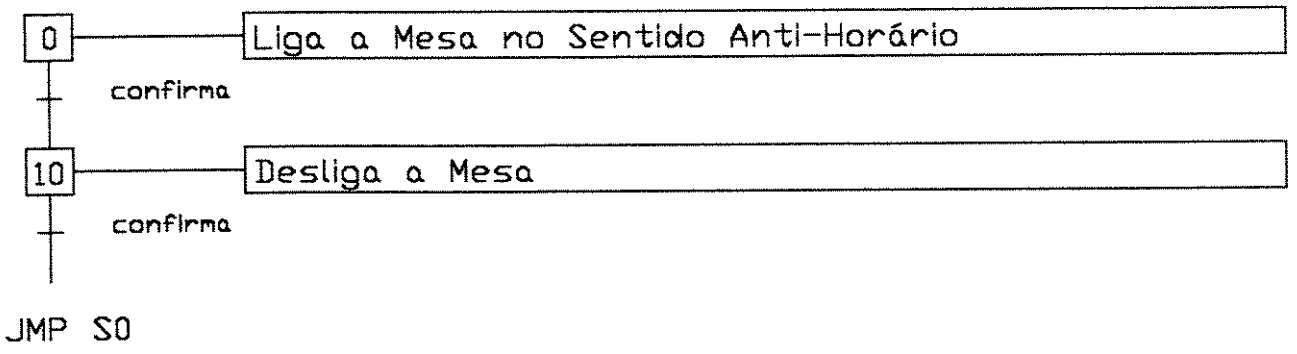

0
Posição Inicial 1
confirma
10
Posição Inicial 2
confirma
12
Lavagem com água
confirma
14
Liga a Bomba de Lavagem
confirma
16
Desliga a Bomba de Lavagem
confirma
18
Posição Inicial 2
confirma
20
Misturador de Cores
confirma
22
Liga o Misturador
confirma
24
Desliga o Misturador
confirma
26
Posição Inicial 2
confirma
28
Despejar após a Lavagem
confirma
30
Posição Inicial 2
confirma
32
Posição Inicial 1
confirma
34
Devolver o Recipiente a Mistura
confirma
JMP S0

MESA - 1


0
Liga a Mesa no Sentido Horário
confirma
10
Desliga a Mesa
confirma

JMP S0

MESA - 2


0
Liga a Mesa no Sentido Anti-Horário
confirma
10
Desliga a Mesa
confirma
JMP S0

COR 1

```
01    LD    M8002
02    SET   S0
03    STL   S0
04    SET   Y013
05    SET   Y014
06    RST   Y015
07    LD    X010
08    SET   S10
09    STL   S10
10    RST   Y013
11    RST   Y014
12    LD    X011
13    SET   S12
14    STL   S12
15    SET   Y014
16    SET   Y015
17    LD    X010
18    SET   S14
19    STL   S14
20    SET   Y000
21    LD    X011
22    SET   S16
23    STL   S16
24    RST   Y000
25    LD    X010
26    SET   S18
27    STL   S18
28    RST   Y014
29    RST   Y015
30    LD    X011
31    SET   S20
32    STL   S20
33    SET   Y013
34    SET   Y015
35    LD    X010
36    SET   S22
37    STL   S22
38    RST   Y013
39    RST   Y015
40    LD    X011
41    OUT   S0
42    RET
```

COR 2

```
01    LD    M8002
02    SET   S0
03    STL   S0
04    SET   Y013
05    SET   Y014
06    RST   Y015
07    LD    X010
08    SET   S10
09    STL   S10
10    RST   Y013
11    RST   Y014
12    LD    X011
13    SET   S12
14    STL   S12
15    SET   Y015
16    LD    X010
17    SET   S14
18    STL   S14
19    SET   Y001
20    LD    X011
21    SET   S16
22    STL   S16
23    RST   Y001
24    LD    X010
25    SET   S18
26    STL   S18
27    RST   Y015
28    LD    X011
29    SET   S20
30    STL   S20
31    SET   Y013
32    SET   Y015
33    LD    X010
34    SET   S22
35    STL   S22
36    RST   Y013
37    RST   Y015
38    LD    X011
39    OUT   S0
40    RET
```

COR 3

```
01    LD    M8002
02    SET   S0
03    STL   S0
04    SET   Y013
05    SET   Y014
06    RST   Y015
07    LD    X010
08    SET   S10
09    STL   S10
10    SET   Y015
11    LD    X011
12    SET   S12
13    STL   S12
14    RST   Y013
15    RST   Y015
16    LD    X010
17    SET   S14
18    STL   S14
19    SET   Y002
20    LD    X011
21    SET   S16
22    STL   S16
23    RST   Y002
24    LD    X010
25    SET   S18
26    STL   S18
27    SET   Y013
28    SET   Y015
29    LD    X011
30    SET   S20
31    STL   S20
32    RST   Y014
33    LD    X010
34    SET   S22
35    STL   S22
36    SET   Y014
37    LD    X011
38    OUT   S0
39    RET
```

COR 4

```
01    LD    M8002
02    SET   S0
03    STL   S0
04    SET   Y013
05    SET   Y014
06    RST   Y015
07    LD    X010
08    SET   S10
09    STL   S10
10    SET   Y015
11    LD    X011
12    SET   S12
13    STL   S12
14    RST   Y014
15    RST   Y015
16    LD    X010
17    SET   S14
18    STL   S14
19    SET   Y003
20    LD    X011
21    SET   S16
22    STL   S16
23    RST   Y003
24    LD    X010
25    SET   S18
26    STL   S18
27    SET   Y014
28    SET   Y015
29    LD    X011
30    SET   S20
31    STL   S20
32    RST   Y014
33    LD    X010
34    SET   S22
35    STL   S22
36    SET   Y014
37    LD    X011
38    OUT   S0
39    RET
```

CÓDIGO

```
01    LD      M8002
02    SET     S0
03    STL     S0
04    RST     Y010
05    RST     Y011
06    RST     Y012
07    LD      X010
08    SET     S10
09    STL     S10
10    SET     Y010
11    LD      X011
12    OUT     S0
13    RET
```

DESPEJO

```
01    LD      M8002
02    SET     S0
03    STL     S0
04    RST     Y010
05    RST     Y011
06    RST     Y012
07    LD      X010
08    SET     S10
09    STL     S10
10    SET     Y010
11    SET     Y011
12    SET     Y012
13    LD      X011
14    SET     S12
15    STL     S12
16    RST     Y010
17    RST     Y012
18    LD      X010
19    SET     S14
20    STL     S14
21    SET     Y010
22    SET     Y012
23    LD      X011
24    OUT     S0
25    RET
```

MIXER

```
01    LD      M8002
02    SET     S0
03    STL     S0
04    RST     Y010
05    RST     Y011
06    RST     Y012
07    LD      X010
08    SET     S10
09    STL     S10
10    SET     Y010
11    SET     Y011
12    LD      X011
13    SET     S12
14    STL     S12
15    RST     Y010
16    RST     Y012
17    LD      X010
18    SET     S14
19    STL     S14
20    SET     Y010
21    SET     Y011
22    SET     Y012
23    LD      X011
24    SET     S16
25    STL     S16
26    RST     Y010
27    RST     Y011
28    LD      X010
29    SET     S18
30    STL     S18
31    SET     Y004
32    LD      X011
33    SET     S20
34    STL     S20
35    RST     Y004
36    LD      X010
37    SET     S22
38    STL     S22
39    SET     Y010
40    SET     Y011
41    LD      X011
42    OUT     S0
43    RET
```

LAVAGEM

```
01      LD      M8002
02      SET     S0
03      STL     S0
04      RST     Y010
05      RST     Y011
06      RST     Y012
07      LD      X010
08      SET     S10
09      STL     S10
10      SET     Y010
11      SET     Y011
12      SET     Y012
13      LD      X011
14      SET     S12
15      STL     S12
16      RST     Y010
17      LD      X010
18      SET     S14
19      STL     S14
20      SET     Y005
21      LD      X011
22      SET     S16
23      STL     S16
24      RST     Y005
25      LD      X010
26      SET     S18
27      STL     S18
28      SET     Y010
29      LD      X011
30      SET     S20
31      STL     S20
32      RST     Y010
33      RST     Y011
34      LD      X010
35      SET     S22
36      STL     S22
37      SET     Y004
38      LD      X011
39      SET     S24
40      STL     S24
41      RST     Y004
42      LD      X010
43      SET     S26
44      STL     S26
45      SET     Y010
46      SET     Y011
47      LD      X011
48      SET     S28
49      STL     S28
50      RST     Y010
51      RST     Y012
52      LD      X010
53      SET     S30
54      STL     S30
55      SET     Y010
56      SET     Y012
57      LD      X011
58      SET     S32
59      STL     S32
60      RST     Y010
61      RST     Y011
62      RST     Y012
63      LD      X010
64      SET     S34
65      STL     S34
66      SET     Y010
67      SET     Y012
68      LD      X011
69      OUT     S0
70      RET
```

MESA 1

```
01      LD      M8002
02      SET     S0
03      STL     S0
04      SET     Y007
05      LD      X005
06      SET     S10
07      STL     S10
08      RST     Y007
09      LD      X010
10      OUT     S0
11      RET
```

MESA 2

```
01      LD      M8002
02      SET     S0
03      STL     S0
04      SET     Y006
05      LD      X006
06      SET     S10
07      STL     S10
08      RST     Y006
09      LD      X010
10      OUT     S0
11      RET
```

ANEXO II

Porgrama para Comunicação

A seguir são apresentados o programa para comunicação em linguagem de alto nível (C) – PRG1.C – e o arquivo INICIO.BAT, utilizados na comunicação dos manipuladores robóticos.

PRG1.C

```
/* file prog_rob.c, build with makecdem.bat, uses prog_rob.h, prog_rob.inc */
/* Sample command line interpreter for scripting language.
/* Demonstrates use of many functions from rbx?.lib's */

#include "prg1.h"
#include "Luizleo.h"

   /* increments for jog commands */
#define LARGE_INC 30
#define SMALL_INC  3

static char acCommandBuf[RBX_COMMAND_LEN_MAX+1];

   /* get the literals tables; this is a convenient C technique    */
   /* to keep literals to in an easy-to-edit table, while          */
   /* using the "names" of the literals in the code.               */
   /* The technique is general and is not unique to this program; */
   /* it is useful during preparation of multi-lingual apps.       */

enum
{
   #define MAC(a,b) a,
   #include "prg1.inc"
   #undef MAC
};
```

```
char *aszLits[] =
{
   #define MAC(a,b) b,
   #include "prg1.inc"
   #undef MAC
};


/* wrapper around rbxSaveCfg call to save configuration file */
static void SaveConfig(char *szFspec)
{
   switch(rbxSaveCfg(szFspec))
   {
      case 0:
       printf("%s: %s\n", aszLits[CfgSavedLit],
          aszLits[DefaultCfgFnameLit]);
       return;

      case 1: goto bad_open;
      case 2: goto bad_close_write;
   }

   bad_close_write:
      unlink(szFspec);

   bad_open:
      printf("%s: %s\n", aszLits[FileWriteErrLit], szFspec);
}

/* noise to make when no action */
void NoActionBeep(void)
{
   sound(100);
   delay(200);
   nosound();
}


void OutOfRangeBeep(void)
{
   sound(2000);
   delay(200);
   nosound();
}


void EnterBeep(void)
{
   sound(2000);
   delay(200);
   nosound();
}


void ErrorBeep(void)
{
   sound(2000);
```

```
   delay(200);
   nosound();
}

void Stat(void)
{
   short sEBase = rbxEnumBaseR();
   short si, sStat;

   static short *psMotorsParam = NULL; /* for clarity; statics are anyway */

      /* could also just declare the array with RBX_MOTOR_PARAM_COUNT_MAX */
   if (!psMotorsParam)
      psMotorsParam = malloc(sizeof(short) * rbxMotorCount);

   printf("%s  ", rbxMotorTtl);

   for (si = sEBase; si < rbxMotorCount + sEBase; si++)
      printf("%*d", RBX_READING_FMT_LEN+1, si);
   printf("\n");

   for (si = sEBase; si < rbxMotorParamCount + sEBase; si++)
   {
      short si1;

      printf("%s  ", rbxMotorParamTtl[si-sEBase /* it's a C-array */]);
      rbxMotorsParamR(si, psMotorsParam);
      for (si1 = 0 /* not sEBase, it's a C-ary */; si1 < rbxMotorCount; si1++)
       printf("%*d", RBX_READING_FMT_LEN+1, psMotorsParam[si1]);
      printf("\n");
   }
   printf("\n");

#if 0
   /* alternate, less "automatic" approach */
   printf("%s\n", aszLits[ShowTitleLit]);

   for (si = sEbase; si < rbxParameterCount + sEBase; si++)


   for (si = sEBase; si < rbxMotorCount + sEBase; si++)
   {
      short sPos, sMn, sMx, sInit, sAcc, sDec, sSpd, sInvrt, sPwr, sPin;

      sPos  = rbxPosR(si);
      sInit = rbxInitPosR(si);
      sMn   = rbxMinPosR(si);
      sMx   = rbxMaxPosR(si);
      sAcc  = rbxAccelR(si);
      sDec  = rbxDecelR(si);
      sSpd  = rbxMaxSpdR(si);
      sInvrt= rbxInvertR(si);
      sPwr  = rbxPowerR(si);
      sPin  = rbxPinnedR(si);

      printf(" %2d %5d %4d %4d %4d   %d  %5d %5d %5d %5d %5d\n",
       si, sPos, sSpd, sAcc, sDec, sPwr, sMn, sMx, sInit, sInvrt, sPin);
```

```
    }
    printf("\n");
#endif


    sStat = rbxStatusR();

    printf("statuses:\n"
        "EnumBase:  %d      "
        "Compiling: %c      "
        "Running:   %c      "
        "Ready:     %c\n"
        "Continued: %c      "
        "CmdWasTxt: %c      "
        "AuxA:      %c      "
        "AuxB:      %c\n",

        rbxEnumBaseR(),
        (sStat & RBX_COMPILING_STAT) ? 'X' : '-',
        (sStat & RBX_RUNNING_STAT) ? 'X' : '-',
        (sStat & RBX_READY_STAT) ? 'X' : '-',
        (sStat & RBX_CONTINUED_STAT) ? 'X' : '-',
        (sStat & RBX_CMD_WAS_TEXT_STAT) ? 'X' : '-',
        (sStat & RBX_AUX_A_STAT) ? 'X' : '-',
        (sStat & RBX_AUX_B_STAT) ? 'X' : '-'
    );

    printf("\n");

    /* read switches and adc's */
    {
        short asAdc[8];
        short asSwitch[8];
        short si;

        printf(rbxSensorTtl);
        rbxAdcsR(asAdc);
        for (si = 0; si < rbxAdcCount; si++)
         printf("%6d", asAdc[si]);

        rbxSwitchesR(asSwitch);
        printf("   ");
        for (si = 0; si < rbxSwitchCount; si++)
         printf("%c", asSwitch[si] ? 'X' : '-');
    }

    printf("\n\n");
}


/* Jog a motor.  Returns 0 if character not recognized; 1 if processed; */
short Jog(int iCh)
{
    //printf("jogging");          //LeoSoft
    short siM, sInc;
    char *pc;
```

```
if (!iCh)
{
   iCh = getch();
   goto extended;
}

iCh = (char)toupper(iCh);

if ((pc = strchr(aszLits[JogLargePlusLit], iCh)) != NULL)
{
   siM = (short)(pc - aszLits[JogLargePlusLit]);
   sInc = LARGE_INC;
   goto movit;
}

if ((pc = strchr(aszLits[JogLargeMinusLit], iCh)) != NULL)
{
   siM = (short)(pc - aszLits[JogLargeMinusLit]);
   sInc = -LARGE_INC;
   goto movit;
}

if ((pc = strchr(aszLits[JogSmallPlusLit], iCh)) != NULL)
{
   siM = (short)(pc - aszLits[JogSmallPlusLit]);
   sInc = SMALL_INC;
   goto movit;
}

if ((pc = strchr(aszLits[JogSmallMinusLit], iCh)) != NULL)
{
   siM = (short)(pc - aszLits[JogSmallMinusLit]);
   sInc = -SMALL_INC;
   goto movit;
}

return 0;

movit:
if (siM > rbxMotorCount - 1)
   return 0;

rbxJumpRel(siM + rbxEnumBaseR(), sInc);
printf("jogging");            //LeoSoft

if (rbxPinnedR(siM))
   OutOfRangeBeep();

return 1;

extended:

if (iCh >= 16 && iCh <= 24)
{
   short sPwr;

   siM = iCh - 16;
```

```
        if (siM > rbxMotorCount - 1)
         return 0;

         /* toggle power */
        rbxPowerW(siM, !rbxPowerR(siM));

        printf("%s %d: %s %d\n",
         rbxMotorTtl, siM+1,
         rbxMotorParamTtl[rbxPowerMotorParamNdx], sPwr);
        return 1;
    }

    return 0;
}


void ReportError(void)
{
    char *tkn;

    if (rbxCmdWasTextR())
    {
        printf("%s\n%*c\n", rbxLastCmdR(), rbxErrColR(), '^');
    }

    printf("%s", rbxErrMsgR());
    tkn = rbxErrTokenR();
    if (*tkn)
        printf(" --  \"%s\"", tkn);
    printf("\n");
}


static void DirectControl(void)
{
    printf("%s\n\n", aszLits[DirectPromptLit]);

    for (;;)
    {
        int ch;
        switch(ch = getch())
        {
         case '^':  Stat(); break;
         case '\x1b':   goto done; /* escape */
         default:
            if (!Jog(ch))
            {
                NoActionBeep();
                printf("%s\n", aszLits[DirectPromptLit]);
            }
            break;
        }
    }
    done:
        ;
}
```

```
static void Help(void)
{
   short sEBase = rbxEnumBaseR();
   short si, scItems;

      /* macros */
   scItems = rbxMacroCountR();

   printf("\n\n%s:\n", aszLits[DrvMacrosTitleLit]);
   for (si = sEBase; si < scItems + sEBase; si++)
      printf("%-20s", rbxMacroNameR(si));

   if (!scItems)
      printf(aszLits[NoMacrosLit]);

      /* resident primitive commands */
   scItems = rbxPrimitiveCount;

   printf("\n\n%s:\n", aszLits[DrvPrimitivesTitleLit]);

   for (si = sEBase; si < scItems + sEBase; si++)
      printf("%-20s", rbxPrimitiveNameR(si));


      /* resident keywords */
   scItems = rbxKeywordCount;
   printf("\n\n%s:\n", aszLits[DrvKeywordsTitleLit]);

   for (si = sEBase; si < scItems + sEBase; si++)
      printf("%-20s", rbxKeywordR(si));

      /* console commands */
   printf("\n\n%s:\n", aszLits[ConCommandsTitleLit]);
   for (si = CommandBeginsLit + 1; si < CommandEndsLit; si++)
      printf("%-20s", aszLits[si]);

   done:
   printf("\n\n");
}


static void GetCommands(void)
{
   char acXBuf[RBX_COMMAND_LEN_MAX+1];     /* extra buffer for strtok'ing */
   char *npos;

   printf("\n%s\n\n", aszLits[TypeHelpForHelpLit]);

   for (;;)
   {
      char *pcPrompt;
      ushort usStat;
      char *pcTkn;
      short scTkns;

      usStat = rbxStatusR();
```

```
if (usStat & RBX_COMPILING_STAT)
{
 if (usStat & RBX_CONTINUED_STAT)
    pcPrompt = aszLits[CompPlusPromptLit];
 else
    pcPrompt = aszLits[CompPromptLit];
 printf("%3d %s ", rbxPartialMacroSizeR(), pcPrompt);
}
else
{
 if (usStat & RBX_CONTINUED_STAT)
    pcPrompt = aszLits[ExecPlusPromptLit];
 else
    pcPrompt = aszLits[ExecPromptLit];
 printf("%s ", pcPrompt);
}

fgets(acCommandBuf, sizeof(acCommandBuf), stdin);
if ((npos = strchr(acCommandBuf, '\n')) != NULL)
 *npos = '\0';

strcpy(acXBuf, acCommandBuf);

scTkns = 0;
if ((pcTkn = strtok(acXBuf, " ")) != NULL)
{
 ++scTkns;
 while (strtok(NULL, " "))
    ++scTkns;
}

/* note that there is no explicit check for a blank line: */
/* blank lines fall through to the rbx call. */

if (scTkns == 1)
{
 if (!stricmp(pcTkn, aszLits[ExitLit]))
    return;

 if (!stricmp(pcTkn, aszLits[SaveConfigLit]))
 {
    SaveConfig(aszLits[DefaultCfgFnameLit]);
    continue;
 }

 if (!stricmp(pcTkn, aszLits[StatusLit]))
 {
    Stat();
    continue;
 }

 if (!stricmp(pcTkn, aszLits[MemleftLit]))
 {
    printf("%d\n", rbxMemleftR());
    continue;
 }
```

```
      if (!stricmp(pcTkn, aszLits[HelpLit]))
      {
         Help();
         continue;
      }

      if (!stricmp(pcTkn, aszLits[DirectLit]))
      {
         DirectControl();
         continue;
      }
     }

     rbxCmd(acCommandBuf);

     if (rbxErrNo)
     {
      ReportError();
      ErrorBeep();
     }
     continue;
   }
}


void main(void)
{
   short lsensor1;                    //LeoSoft Estado do Sensor 1
   short lsensor2;                    //LeoSoft Estado do Sensor 2
   short lsensor3;                    //LeoSoft Estado do Sensor 3
   short lsensor4;                    //LeoSoft Estado do Sensor 4
   short lsensor5;                    //LeoSoft Estado do Sensor 5
   short lsensor6;                    //LeoSoft Estado do Sensor 6
   short lsensor7;                    //LeoSoft Estado do Sensor 7

   short sInitErr;
   switch (sInitErr = rbxAttach())

   {
      case 0:
       break;

      case 1:
       printf("%s\n", aszLits[DrvNotInstalledLit]);
       exit(1);

      default:
       printf("rbx_Attach error [%d], not recognized", sInitErr);
       exit(1);
   }


   printf("%s: %s...", aszLits[CfgLoadingLit], aszLits[DefaultCfgFnameLit]);
   switch (rbxLoadCfg(aszLits[DefaultCfgFnameLit]))
   {
      case 1:
```

```
     printf("%s: %s\n", aszLits[FileLoadErrLit],
            aszLits[DefaultCfgFnameLit]);
     printf("%s: %d\n", aszLits[LineLit], rbxErrLine);
     ReportError();
     break;

    case 2:
     printf("%s: %s\n", aszLits[CfgNotFoundLit],
            aszLits[DefaultCfgFnameLit]);
     break;

    default: /* went ok */
     printf("%s\n", aszLits[DoneLit]);
     break;
}

rbxRestart(); /* reprogam Adapter in case power has been interrupted */

printf("%s: %s...",
   aszLits[StartupLoadingLit], aszLits[DefaultStartupFnameLit]);
switch (rbxLoad(aszLits[DefaultStartupFnameLit]))
{
    case 1:
     printf("%s: %s\n", aszLits[FileLoadErrLit],
            aszLits[DefaultStartupFnameLit]);
     printf("%s: %d\n", aszLits[LineLit], rbxErrLine);
     ReportError();
     break;

    case 2:
     printf("%s: %s\n", aszLits[StartupNotFoundLit],
            aszLits[DefaultStartupFnameLit]);
     break;

    default: /* went ok */
     printf("%s\n", aszLits[DoneLit]);
     break;
}

while(!kbhit())                 // LeoSoft leitura dos Sensores e comparacao
{
int  valor;
   for (valor = 0; valor < 100; valor++)
{
delay (1);
lsensor1=rbxSwitchR(1);         //Leitura do sensor 1 (traj1)
lsensor2=rbxSwitchR(2);         //Leitura do sensor 2 (traj2)
lsensor3=rbxSwitchR(3);         //Leitura do sensor 3
lsensor4=rbxSwitchR(4);         //Leitura do sensor 4
lsensor5=rbxSwitchR(5);         //Leitura do sensor 5
lsensor6=rbxSwitchR(6);         //Leitura do sensor 6 (retorno)
lsensor7=rbxSwitchR(7);         //Leitura do sensor 7 (fim)
}
if (lsensor1==0 && lsensor2==0 && lsensor3==0 && lsensor4==1)
{
 printf ("\n posicao inicial 1");
 /* printf ("\n 0001 --> traj10" ); */
```

```
    rbxLoad("TRAJ10.RBX");
    }

if (lsensor1==1 && lsensor2==1 && lsensor3==1 && lsensor4==0)
   {
    printf ("\n posicao inicial 2");
    /* printf ("\n 1110 --> traj11" ); */
    rbxLoad("TRAJ11.RBX");
    }

   if (lsensor1==1 && lsensor2==1 && lsensor3==0 && lsensor4==0)
   {
    printf ("\n trajetoria pegar recipiente de mistura");
    /* printf ("\n 1100 --> traj12" ); */
    rbxLoad("TRAJ12.RBX");
    }

   if (lsensor1==1 && lsensor2==0 && lsensor3==1 && lsensor4==0)
   {
    printf ("\n trajetoria devolver o recipiente de mistura");
    /* printf ("\n 101 --> traj13" ); */
    rbxLoad("TRAJ13.RBX");
    }

   if (lsensor1==0 && lsensor2==1 && lsensor3==1 && lsensor4==0)
   {
    printf ("\n trajetoria cor1");
    /* printf ("\n 011 --> traj14" ); */
    rbxLoad("TRAJ14.RBX");
    }

   if (lsensor1==0 && lsensor2==0 && lsensor3==1 && lsensor4==0)
   {
    printf ("\n trajetoria cor2");
    /* printf ("\n 001 --> traj15" );  */
    rbxLoad("TRAJ15.RBX");
    }

   if (lsensor1==0 && lsensor2==1 && lsensor3==0 && lsensor4==0)
   {
    printf ("\n trajetoria cor3");
    /* printf ("\n 010 --> traj16" ); */
    rbxLoad("TRAJ16.RBX");
    }

   if (lsensor1==1 && lsensor2==0 && lsensor3==0 && lsensor4==0)
   {
    printf ("\n trajetoria cor4");
    /* printf ("\n 100 --> traj17" ); */
    rbxLoad("TRAJ17.RBX");
    }

   if (lsensor1==0 && lsensor2==0 && lsensor3==0 && lsensor4==0)
   {
    printf ("\n espera / restart");
    /* printf ("\n 100 --> traj18" ); */
    rbxLoad("TRAJ18.RBX");
```

```
    }

   if (lsensor5==1)
   {
    printf ("\n Sensor 5");
   }

   if (lsensor6==1)
   {
    printf ("\n Sensor 6");
   }

   if (lsensor7==1)
   {
    printf ("\n Sensor 7");
   }
 }
}
```

INICIO.BAT

```
echo off
cls
type mensagem.txt
pause
cls

rbxdrv -n -p1
if errorlevel 2 goto error
rbxcmd /q p0pos all 3000;
maxpos 1 3000; maxpos 2 3000; maxpos 3 3000; maxpos 4 3000; maxpos 5 3000;
maxpos 6 3000;
minpos 1 -1400; minpos 2 -1400; minpos 3 -1400; minpos 4 -1400; minpos 5 -
1400; minpos 6 -1400;

if errorlevel 1 goto error
rbxcmd /q initpos 1 1020; initpos 2 -400; initpos 3 -320; initpos 4 -330;
initpos 5 -920; initpos 6 -600;

if errorlevel 1 goto error
rbxcmd /q restart

prg1
```

ANEXO III

Elementos da plataforma

Os elementos utilizados na montagem da plataforma descrita estão relacionados abaixo. Os valores aproximado dos componentes estão baseados no mês de abril/2000.

Qt	Descrição	Valor unitário	Total
	Acessórios	R$ 75,00	R$ 75,00
5	Bomba de gasolina	R$ 13,50	R$ 67,50
1	CLP Mitsubishi Fx0N15 entradas e 14 saídas	R$ 600,00	R$ 600,00
1	Conjunto de redução	R$ 300,00	R$ 300,00
1	Demultiplexador	R$ 15,00	R$ 15,00
1	Fonte 5V – 1A	R$ 50,00	R$ 50,00
1	Fonte CC 0V-30V – 4A	R$ 500,00	R$ 500,00
1	Leitora de código de barras	R$ 400,00	R$ 400,00
2	Manipuladores robóticos didáticos	R$ 1500,00	R$ 3000,00
3	Microcomputadores 486 configuração básica	R$ 500,00	R$ 1500,00
1	Motor DC 12V – 1A	R$ 10,00	R$ 10,00
1	Motor DC 24V	R$ 30,00	R$ 30,00
6	Relês 24V – 5V	R$ 6,00	R$ 36,00
6	Relês 5V - 24V	R$ 6,00	R$ 36,00
2	Sensores indutivos	R$ 20,00	R$ 20,00
	Total		R$ 6639,50

Alguns componentes podem ser adquiridos em casa que trabalham com produtos de segunda linha, ou segunda mão, sendo possível portanto adquiri-los sem custo nenhum.