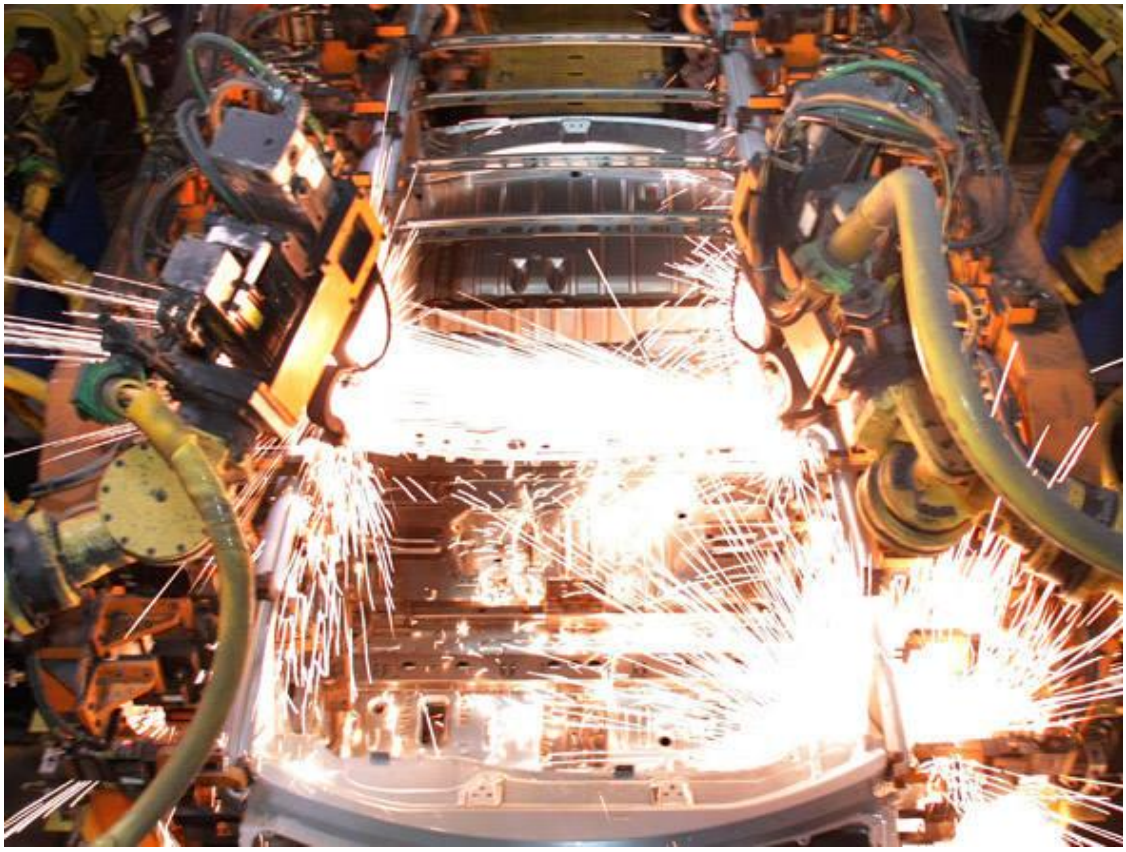
LABORATÓRIO DE AUTOMAÇÃO DE SISTEMAS ELÉTRICOS



2012

2. Controle de Esteira Transportadora

Prof. Eduardo Cesar Senger ***PEA/EPUSP***

Esta experiência visa dar continuidade à familiarização do aluno com os modernos equipamentos e soluções utilizados na automação dos sistemas elétricos industriais. Para tanto, será implementado um sistema para controle de uma esteira transportadora baseado em inversor de frequência, controlador programável e painel de IHM *touch-screen.*

Controle de Esteira Transportadora

Nesta experiência o aluno deverá desenvolver, implementar e testar duas versões de um sistema para controle de uma esteira transportadora utilizando o controlador programável CompactLogix. A esteira é acionada por um motor de indução trifásico através de redutor de velocidade e as duas versões do sistema de controle são bastante distintas em termos tecnológicos.

A primeira versão, mais simples e tradicional, utiliza contatores para implementar o controle do sentido de rotação do motor e a limitação da corrente de partida (chave estrela-triângulo). A partida/parada da esteira nessa versão é realizada através de botoeiras físicas (*push-button*).

Na segunda versão, mais moderna e sofisticada, todo o controle do motor é realizado através de um inversor de frequência. Além disso, os comandos de partida/parada e o monitoramento do estado de todo o sistema é realizado através de um painel de IHM, do tipo *touch-screen.* Nos itens a seguir, cada uma dessas duas versões é discutida detalhadamente.

1. Versão 1: Controle da esteira utilizando chave estrela-triângulo

Como mostrado na figura 1, nesta versão o motor é energizado com a utilização de uma placa que possui quatro contatores cujo acionamento é controlado pelo CP. Nessa versão o sistema de controle a ser desenvolvido deverá atender aos seguintes requisitos:

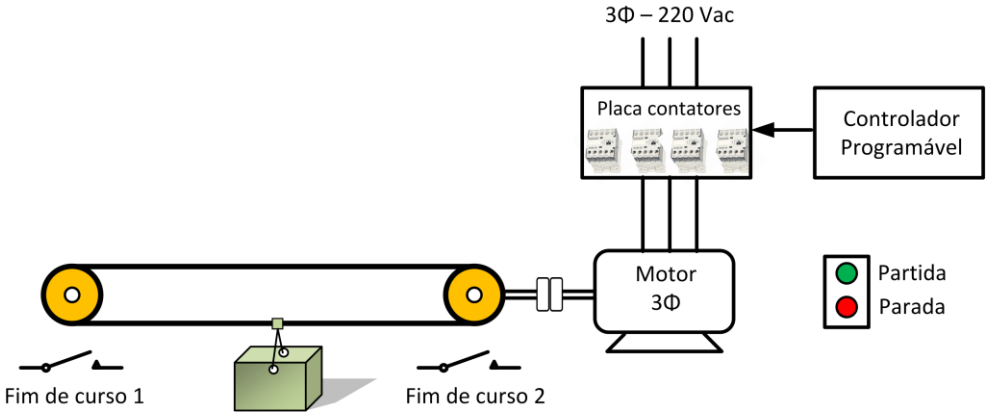


Figura 1. Controle da esteira utilizando chave estrela-triângulo

a)- A partida da esteira se dá através do acionamento do *push-button* de partida

b)- De forma a reduzir a corrente de partida, o motor de indução trifásico (220 V) que aciona a esteira deve inicialmente ser ligado em estrela. Após 5 segundos de funcionamento a ligação deve ser automaticamente alterada para triângulo.

c)- O motor irá acionar a esteira até que a caçamba de transporte de material atinja a posição 1 (esta situação é detectada pela chave fim de curso 1). O motor deve ser desligado por 10 segundos (tempo necessário para a carga/descarga do material) e em seguida sua rotação deve ser automaticamente revertida de modo que a caçamba retorne à posição 2.

d)- O *push-button* de parada deve interromper o movimento da esteira em qualquer etapa do processo. Ao ser novamente acionado o *push-button* de partida, o processo deve ser completado até o retorno da caçamba à posição 2.

e)- Em qualquer etapa do processo, a partida do motor deve sempre ser feita na ligação estrela e revertida para triângulo após 5 segundos de funcionamento.

f)- Em um sistema real seria necessário também incluir a chave para seleção do modo *Manual/Automático* e a botoeira de *Partida/Parada* manual como realizado no exemplo da experiência 1. No sistema didático considerado neste item, no entanto, o modo manual não será considerado para simplificar a montagem no laboratório. Além disso, também visando a simplificação do sistema, o monitoramento de falhas na partida do motor, utilizado na experiência 1, não será implementado neste exemplo.

1.1 Circuitos de Controle e de Potência

Como mostrado na figura 2, nessa versão do sistema, o circuito de potência alimenta o motor com tensões trifásicas, 220 V, através de uma placa com quatro contatores. Dois desses contatores (A e B) são utilizados no controle do sentido de rotação do motor. O contator A é responsável por girar o motor no sentido de levar a caçamba da posição 2 para 1. O contator B, por sua vez, inverte o sentido de rotação através de uma mudança na sequência de fase do trifásico. Os outros dois contatores (C e D) são utilizados para fazer as ligações das bobinas do motor em estrela ou em triângulo. O contator C, quando energizado, liga as bobinas do motor em estrela e o contator D altera essa ligação para triângulo. A tensão nominal das bobinas desses contatores é 24 Vcc, compatível com a placa de saídas digitais do CP.

Os circuitos de controle, constituídos pelos sistemas de entradas e saídas digitais, operam com tensão de 24 Vcc e são mostrados nas figuras 3 e 4, respectivamente. Ao cartão de entradas digitais são conectados os dois push-button de partida e parada e as duas chaves fim de curso. A figura 4 ilustra a forma de conexão das bobinas dos contatores com o cartão de saída digital do CP. Esse circuito permite ao controlador acionar convenientemente cada contator ao longo de todas as etapas do processo de controle da esteira. Os contatos normalmente fechado em série com essas bobinas proporcionam um intertravamento físico entre os pares de contatores A,B e C,D. Esse intertravamento é necessário por razões de segurança, uma vez que o acionamento simultâneo dos contatores A e B ou C e D provoca um curto circuito no trifásico.

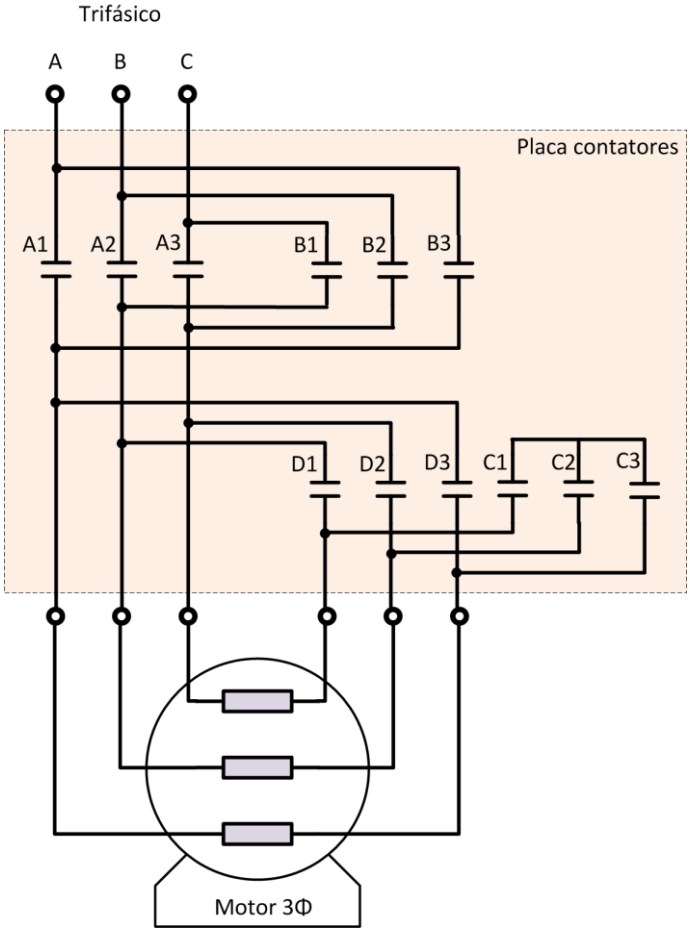


Figura 2. Circuito de Potência utilizado para energizar o motor trifásico da esteira na versão 1

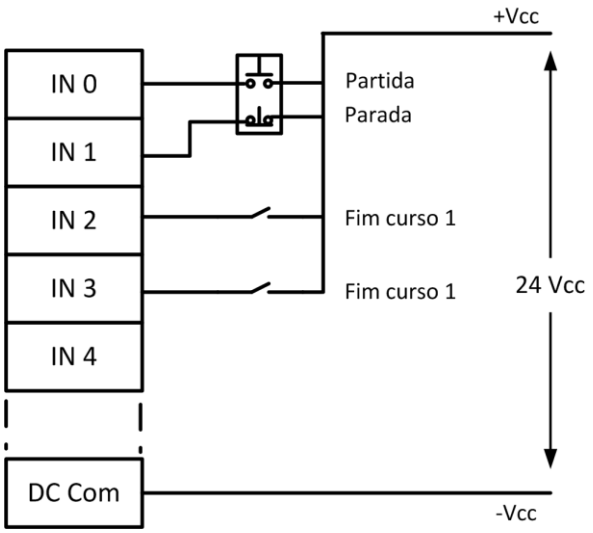


Figura 3. Circuito de controle: cartão de entradas digitais

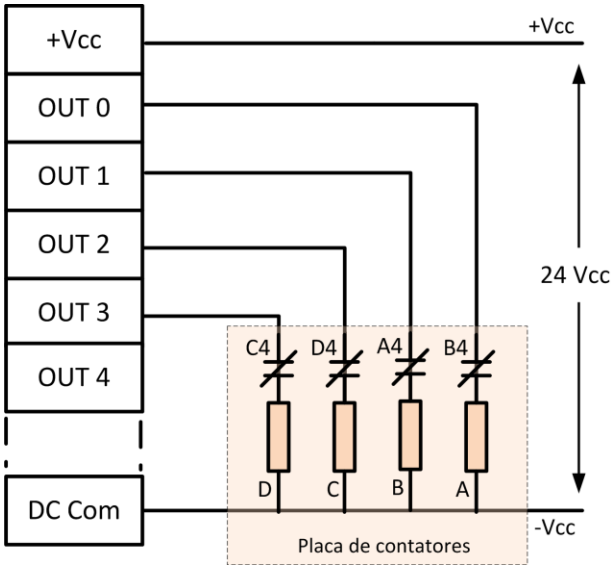


Figura 4. Circuito de Controle: cartão de saídas digitais

1.2 Aplicativo para controle da esteira na versão 1

Na figura 5 são mostradas a estrutura e a base de dados do aplicativo proposto para a versão 1 do sistema (as variáveis mostradas foram definidas no escopo do controlador, isto é, funcionam como variáveis globais estando disponíveis para todos os programas).

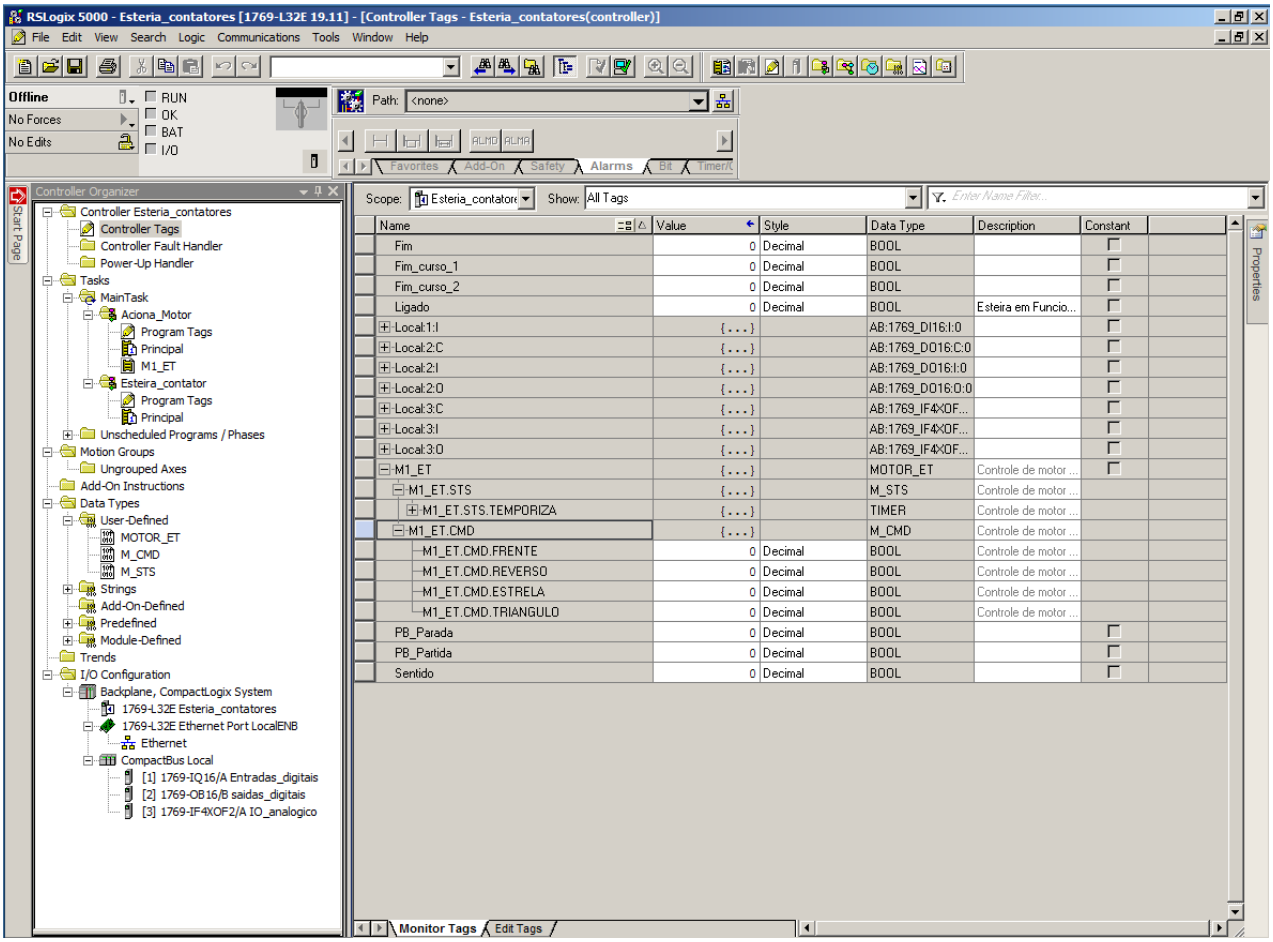


Figura 5. Estrutura e base de dados

Observe que, a exemplo do realizado na experiência 1, utiliza-se uma estrutura, denominada *M1_ET*, cujos campos armazenam toda a informação, tanto de status quanto de comando, associada com o motor. Os campos dessa estrutura, no entanto, são diferentes da anterior devido aos seguintes pontos: a)- o controle do motor agora altera o tipo de ligação das bobinas do estator (estrela-triangulo) e também o sentido de rotação (frente e reverso); b)- o monitoramento de falhas na partida do motor não foi implementado.

Os programas e rotinas que compõem esse aplicativo são apresentados na tabela 1 e detalhados a seguir.

Tabela 1. Estrutura do Software de controle para a versão 1

Task	Programas	Rotinas	Linguagem
MainTask	Aciona_Motor	Principal	Ladder
		M1_ET	Ladder
	Esteira_contator	Principal	Ladder

A. Programa: ***Aciona_Motor***

Este programa possui duas rotinas:

- Rotina: ***Principal***

Esta rotina é escrita em linguagem Ladder e constituída por uma única linha, como mostrado na figura 6, a qual faz a chamada das demais rotinas (no caso, somente a rotina *M1_ET).*

- Rotina: ***M1_ET***

De forma análoga ao realizado na experiência 1, esta rotina é responsável pelo controle do motor considerando agora o comando do sentido de rotação (frente e reverso) e a forma de conexão das bobinas (estrela-triangulo). A figura 7 mostra parcialmente o código para essa rotina.

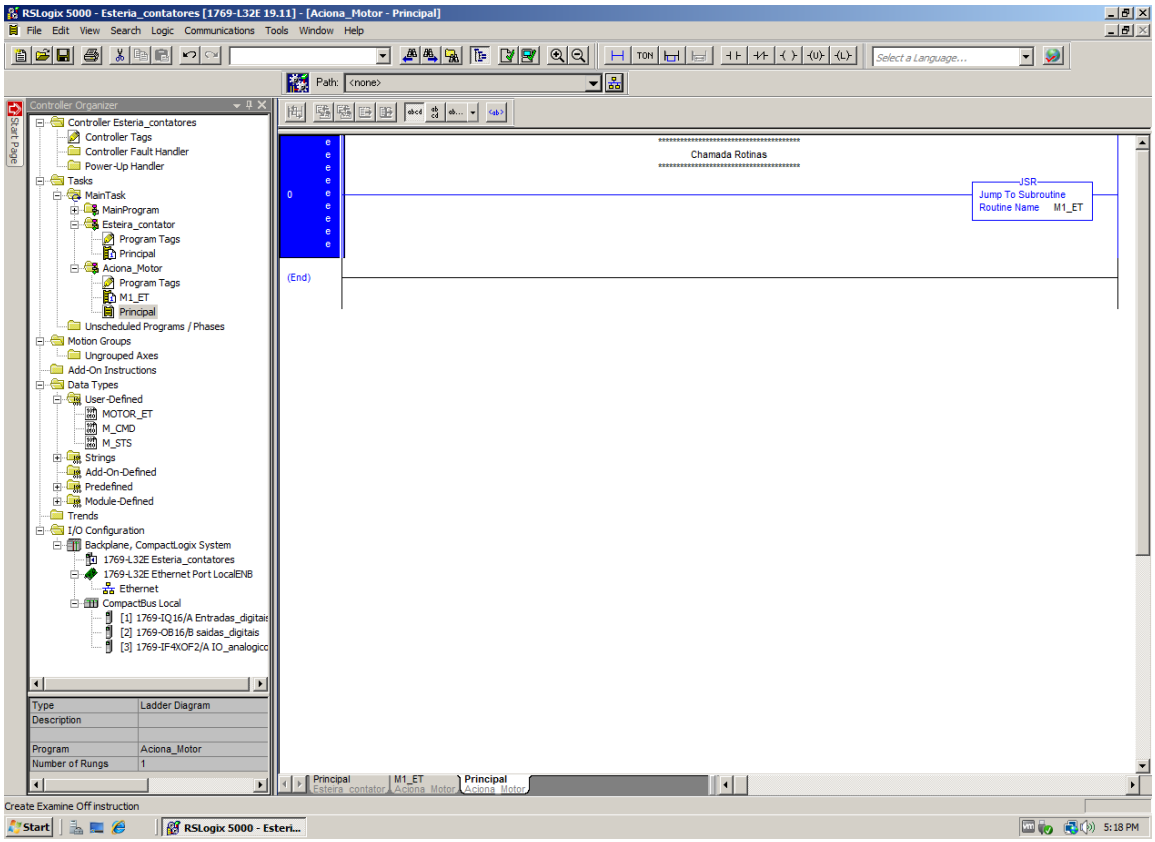


Figura 7. Rotina Principal do programa Aciona_Motor

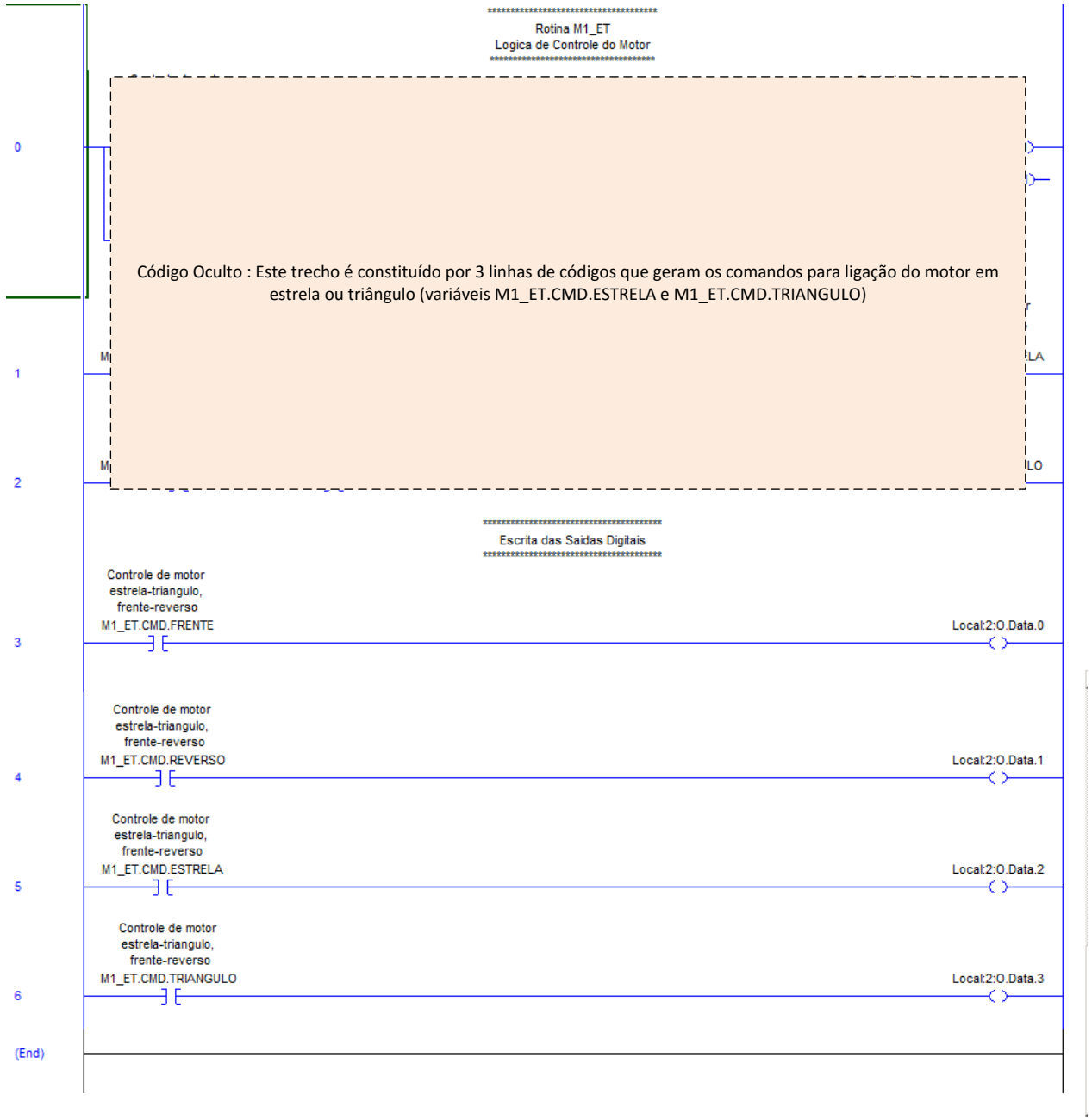


Figura 7. Rotina *M1_ET*

B. Programa: ***Esteira_contator***

Este programa possui uma única rotina, denominada *Principal*, escrita em linguagem *Ladder*, cujo código é parcialmente apresentado na figura 8. A quatro primeiras linhas desse código constituem o bloco de Leitura das Entradas digitais.

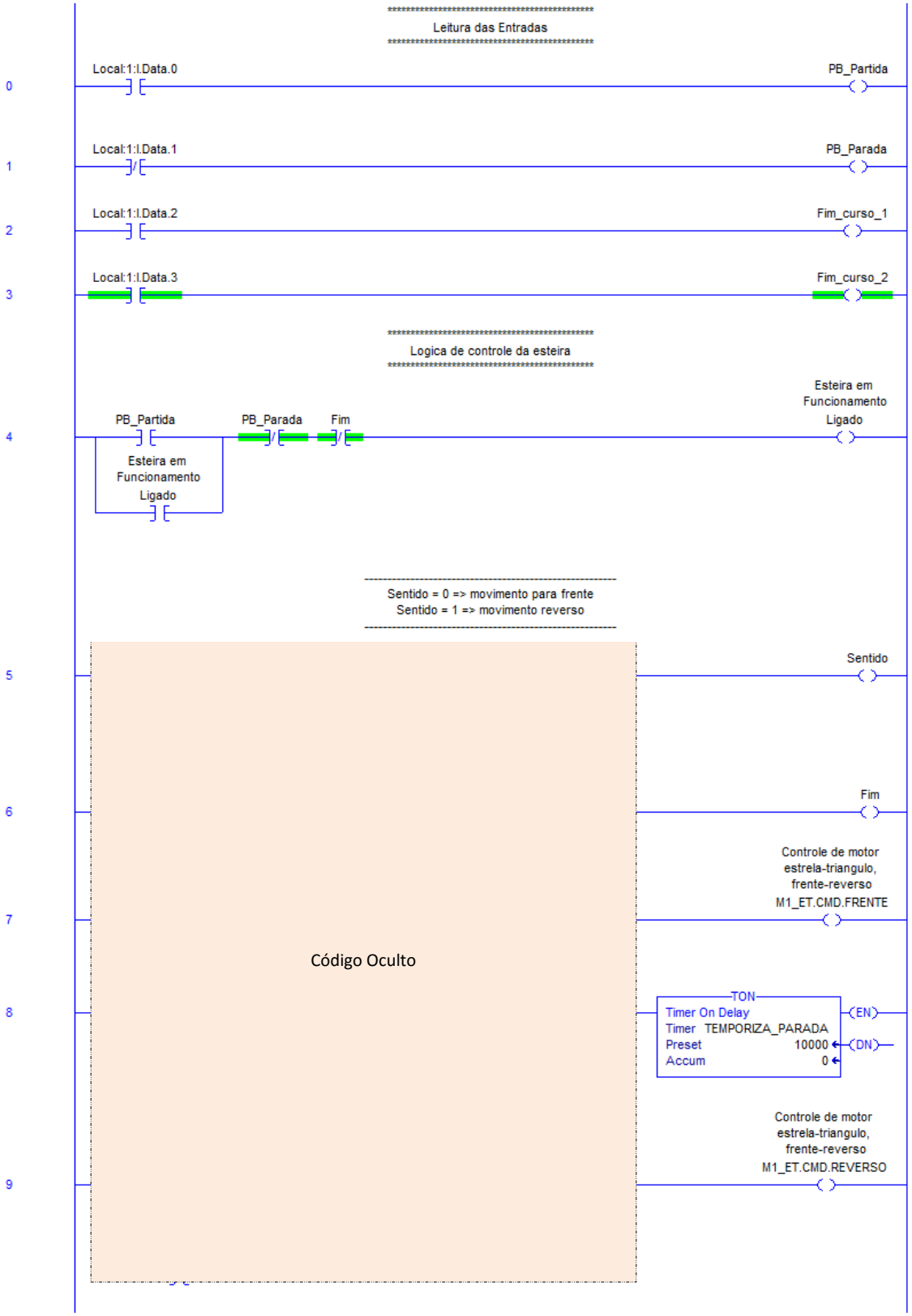


Figura 8. Rotina Principal do programa Esteira_contator

Na linha 4 é definida a variável *Ligado,* a qual torna-se ativa durante todo o ciclo de funcionamento da esteira, isto é, desde que a caçamba parte da posição 2, após o acionamento do *push-button* de Partida, até o seu completo retorno a essa posição. O encerramento de um ciclo completo de movimentação da esteira é indicado pela variável *Fim,* a qual é definida na linha 6 do diagrama *Ladder*.

A linha 5 define a variável *Sentido,* a qual assume o valor 0 quando a esteira faz o caminho de ida (da posição 2 para a posição 1), e o valor 1 durante o caminho de retorno (da posição 1 para 2). Quando o botão de Parada é acionado durante o deslocamento da esteira, essa variável memoriza o sentido que deverá ser seguido após o sistema ser colocado novamente em movimento através do acionamento do botão Partida.

As ultimas três linhas (7 a 9) são utilizadas para gerar as variáveis de comando *M1_ET.CMD.FRENTE* e *M1_ET.CMD.REVERSO*, as quais irão controlar, através do acionamento conveniente dos contatores A e B, a partida do motor em uma das duas direções de movimentação da esteira.

1.3 Atividades didáticas para implementação da versão 1 do sistema de controle

a) Na semana anterior à realização da Experiência 2 o aluno deverá analisar o código das rotinas apresentadas no item 1.2, completar as partes que encontram-se ocultas nas figuras 6 e 8 e implementar e testar o aplicativo completo no software RSLogix 5000 que encontra-se instalado nos computadores da Sala Energia.

b) Após a implementação ser concluída deverá salvar todas as telas e elaborar uma documentação detalhada, a qual deverá ser apresentada no relatório da Experiência 1.

c) O código desenvolvido deverá ser trazido para a aula da Experiência 2, onde, como primeira atividade dessa aula, seu funcionamento deverá ser testado e avaliado em tempo real.

2. Versão 2: Controle da esteira utilizando inversor e painel IHM *Touch-screen*

Nesta segunda versão do sistema, a placa de contatores será substituída por um inversor de frequência (PowerFlex 40-E) para controle do motor e todo o comando e monitoramento do sistema será realizado através de um painel *touch-screen* (PanelView Plus 1000), como mostrado na figura 9. Esses dois equipamentos comunicam-se com o CP através de rede Ethernet e devem ser conectados ao switch existente na lateral traseira do CP. O endereço IP utilizado nos diversos equipamentos existentes em cada bancada é mostrado na tabela 1, onde o dígito X corresponde ao número da bancada. Nos itens a seguir são discutidos os detalhes para configuração e programação de cada um desses três equipamentos.

Tabela 1. Endereço IP dos equipamentos

Equipamento	IP
Computador	192.168.1.X5
Controlador Programável	192.168.1.X0
PanelView	192.168.1.X1
Inversor	192.168.1.X2

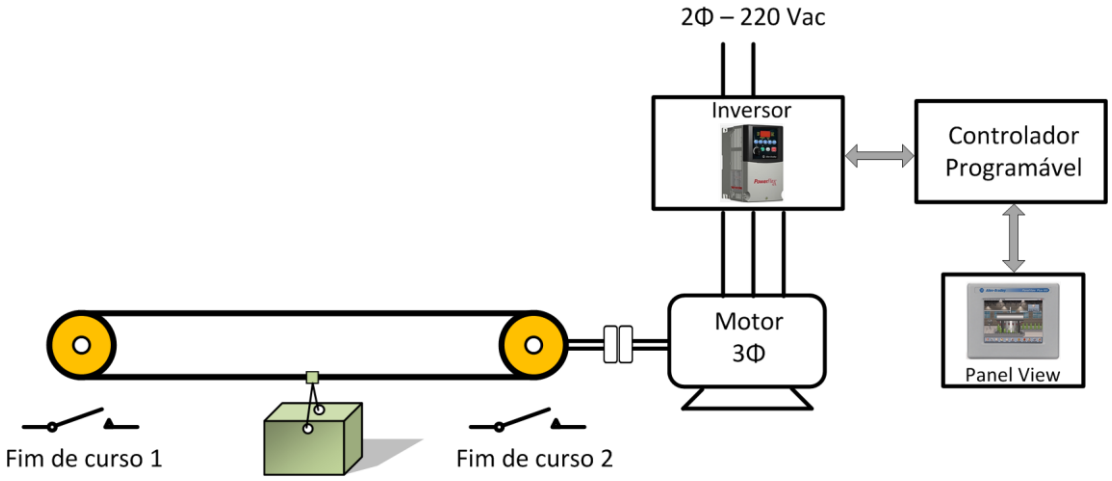


Figura 9. Controle da esteira utilizando Inversor de frequência e Painel IHM

Como mostrado na figura 9, o inversor é alimentado por duas fases da rede elétrica e na sua saída gera o sistema trifásico para alimentar as bobinas do motor que devem estar na ligação delta. Caso os três equipamentos ligados em rede encontrem-se corretamente configurados, eles devem aparecer na tela do RSLinx Classin como mostrado na figura 10 (exemplo de configuração para a bancada 3).

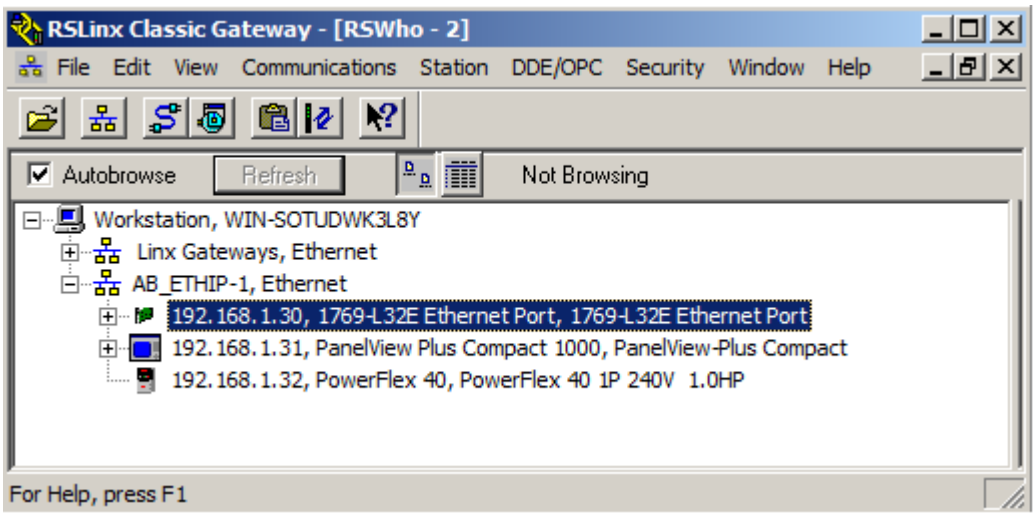


Figura 10. Configuração do CP, IHM e inversor na tela do RSLinx

2.1 Configuração do Inversor de Frequência

O inversor deverá ser configurado através do aplicativo (Start – All programs – DriveTools – DriveExecutive). Após iniciar o aplicativo, clique no menu Drive – Connect to Drive e escolha o inversor Powerflex40. Após a comunicação com o inversor ser estabelecida, clicar em Tools – Wizard – Powerflex40 Startup Wizard que irá iniciar um assistente de configuração. Avançar até as telas de configuração mostradas nos itens abaixo onde as opções indicadas em cada item deverão ser selecionadas.

OBS: Antes de iniciar a configuração do Inversor coloque o CLP no modo *off-line* (ou retire o cabo de rede do CLP)

a) Tela: **Controle do Motor**

Opções:
- *Modo Perf Torq* = Sensrls Vect
- *Escor Hertz em I nom* = 2.0 Hz

b) Tela: **Dados do Motor**

Opções:

- *Corrente Sobrec Motor* = 0.5 Amps
- *Volts Mtr ID* = 220 Volt
- *Hertz Mtr ID* = 120 Hz

c) Tela: **Modo de Parada/Tipo de Frenagem**

Opções:
- *Sel Resistor FD* = Disabled
- *Modo Parada* = Coast CF

d) Tela: **Teste de Direção**

Opções:
- *Referência de Jog* = 30.0 Hz

Obs: acionar a tecla *Jog* e informar se a direção de rotação do motor está correta

e) Tela: **Auto-ajuste**

Obs: Acionar a tecla *Ajuste Estát* de forma que o motor possa aquisitar os parâmetros característicos do motor.

f) Tela: **Taxas de Rampa/Limites de Velocidade**

Obs: Os valores recomendados para os parâmetros dessa tela são mostrados na figura 10.

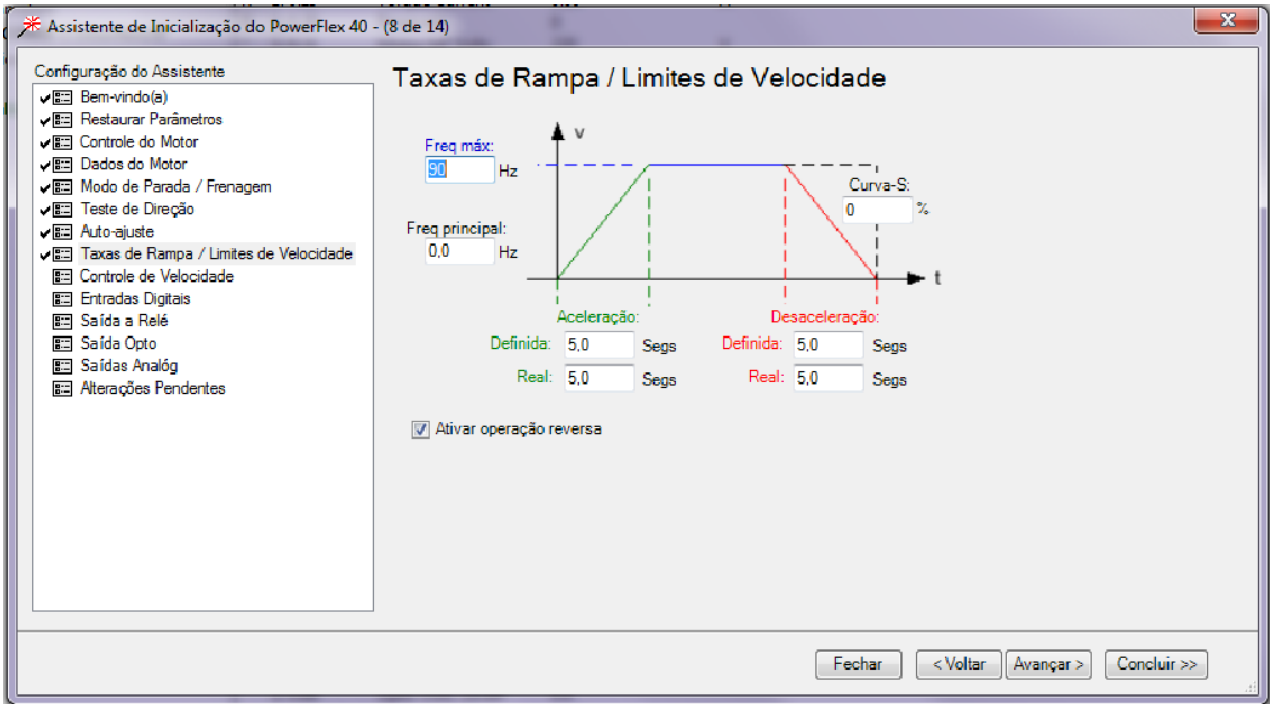


Figura 10. Ajuste para Taxas de Rampa/Limites de Velocidade

Como pode ser observado da figura 10, esta tela define a velocidade máxima de operação, a rampa de aceleração e a possibilidade de operação reversa do motor. A rampa de aceleração substitui o

controle do motor através das chaves estrela-triangulo utilizadas na implementação da versão 1 do sistema. O controle da rampa de desaceleração não será efetivado, dado que no item c (tela **Modo de Parada/Tipo de Frenagem)** selecionou-se a opção *Coast* para o modo de parada.

g) Tela: **Entradas Digitais**

Nesta versão do controle será considerado que a esteira pode operar em duas velocidades predefinidas: velocidade alta (inversor operando com 90 Hz) e velocidade baixa (inversor operando com 30 Hz). A velocidade de operação será selecionada através da IHM touch-screen. Como mostrado na figura 11, esses dois valores são inseridos nessa tela como as frequências predefinidas 0 e 2 (o inversor permite predefinir até 8 frequências de operação).

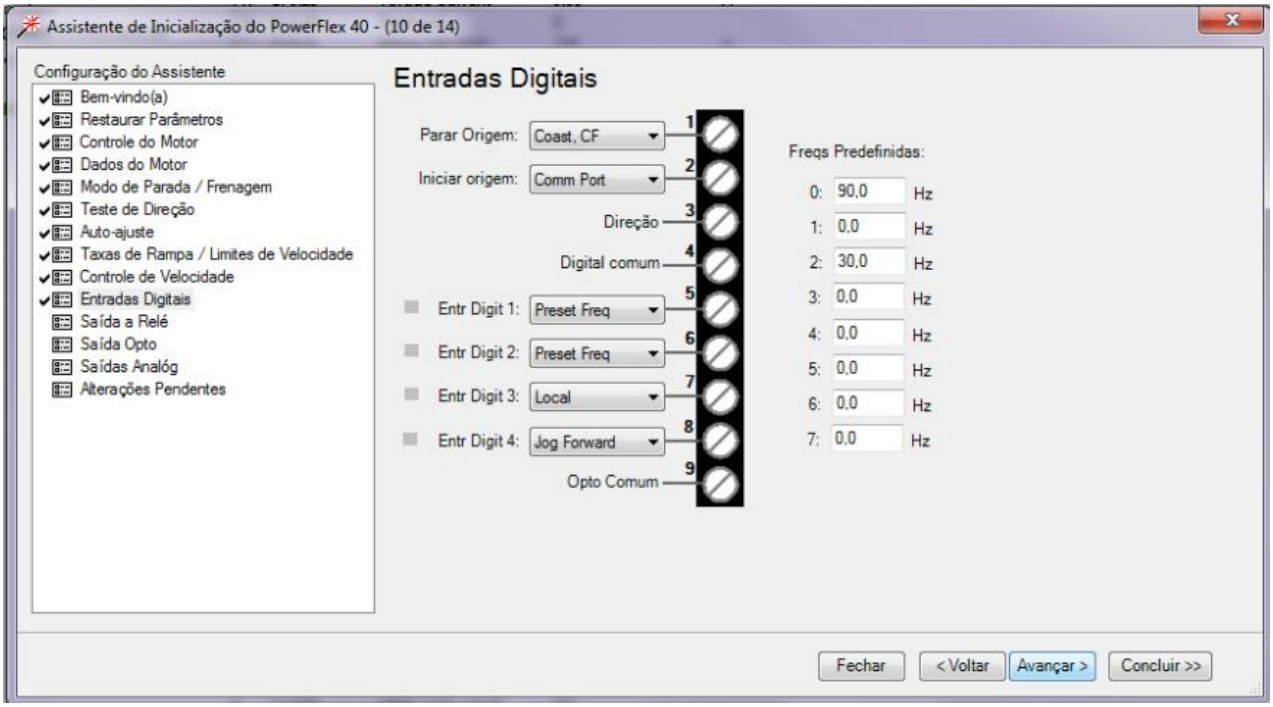


Figura 11. Definição das frequências de operação predefinidas.

Clicar em *Concluir,* em seguida em *Sim* e fechar o programa *DriveExecutive.*

2.2 Programação do Controlador Programável

Antes de iniciar a configuração do CP e a criação de um novo projeto (aplicativo) verifique a existência da conexão AB_ETHIP-1 (abra o RSLinx Classic, clique em Communications/RSWho). Em seguida configure os módulos de hardware utilizados no CP do laboratório e inicie a criação de um novo programa aplicativo através dos passos a seguir.

a) Abra o programa RSLogix5000 localizado em Menu Iniciar/Todos os programas/Rockwell Software/RS Logix 5000 Enterprise Series/ **RSLogix5000**

b) Clique em file/new

c) Abrirá a tela New Controller. Dê um nome para o seu programa em Name, selecione o CP a ser utilizado; no caso, o 1769-L32E CompactLogix5332E Controller e a revisão do Firmware (Revision: 19).

d) Clique em OK. A tela do seu programa irá abrir.

e) Clique com o botão direito sobre CompactBus Local localizado à direita na tela

f) Clique em New Module..., abrirá a tela Select Module.

g) Clique sobre a categoria Digital e em seguida sobre o módulo 1769-IQ16.

h) Confirme a inclusão desse módulo clicando OK

i) A tela de propriedade do módulo deverá abrir, configure o Slot como 1 e clique OK

j) Repita os passos de e) a i) para adicionar os seguintes módulos:

- Saídas Digitais: 1769-OB16 (Slot 2)
- I/O Analógico: 1769IF4XOF2 (Slot 3)

k) Na tela de propriedades de cada módulo é possível adicionar um nome e uma descrição para os mesmos, além, é claro, de configurar algumas particularidades do módulo. Verifique as propriedades de cada módulo que você acabou de adicionar.

l) Clique novamente em New Module..., abrirá a tela Select Module.

m) Clique em I_O Configuration /Backplane, CompactLogix System/1769-L32E Ethernet Port LocalENB/Ethernet e em seguida sobre o módulo PowerFlexc40-E para inclusão do inversor. Insira o endereço IP do inversor de acordo com a Tabela 1.

n) Clique em file/save e salve seu programa

Após concluir o desenvolvimento de seu aplicativo no software RSLogix5000, siga o procedimento abaixo para carregar o aplicativo do PC para o CP:

a) Clique em Communications/Who Active a tela Who Active abrirá.

b) Localize o CP de sua bancada (AB_ETHIP-1,Ethernet/“nº do IP do CP”) e clique em Download.

A estrutura do aplicativo a ser implementado no CP, como mostrado na tabela 2 e na figura 11, é constituído por uma única Tarefa (Tarefa Principal) do tipo contínua, constituída por 2

programas: ***Aciona_Motor*** e ***Esteira_Inversor.*** Cada programa é constituído por duas rotinas, como mostrado na tabela 2.

Tabela 2. Estrutura do Software para controle da Esteira – Versão 2

Tarefa	Programas	Rotinas	Linguagem
Tarefa Principal	*Aciona Motor*	*Principal*	Ladder
		M01_IN	Ladder
	Esteira_Inversor	*Principal*	Ladder
		Sequencia_do_Controle	SFC

As variáveis de dados utilizadas no software são mostradas na figura 12. Observe que, alem das variáveis booleanas, foram utilizadas duas estruturas conforme detalhado nas tabelas 3 e 4. A primeira estrutura armazena os comandos que serão enviados ao inversor. Já a segunda estrutura, de forma semelhante ao utilizado na experiência 1, armazena a informação de status e de comando associado ao motor agora controlado por inversor (vide figura 13 e tabela 4).

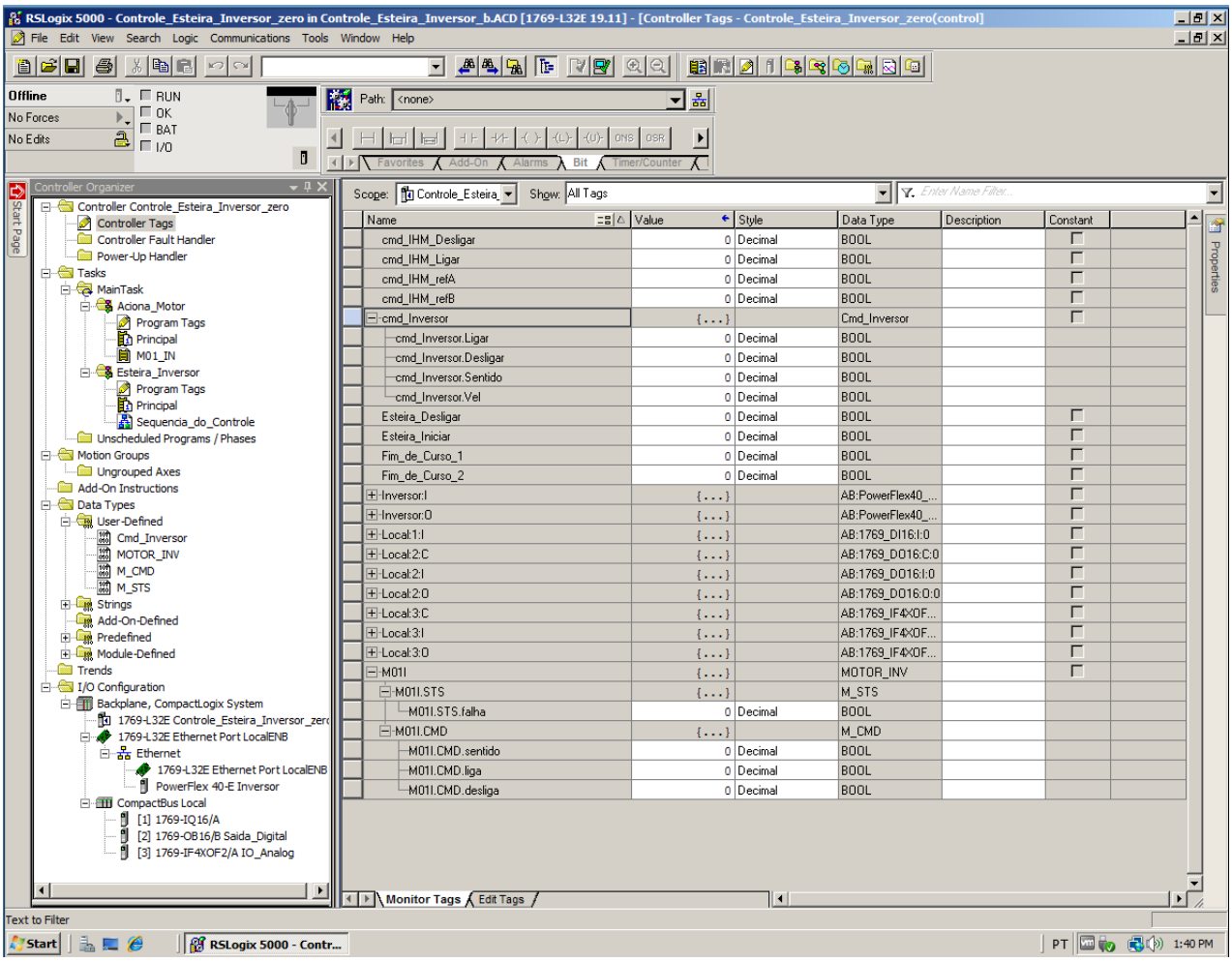


Figura 12. Estrutura e Base de dados da versão 2 do software para controle da esteira

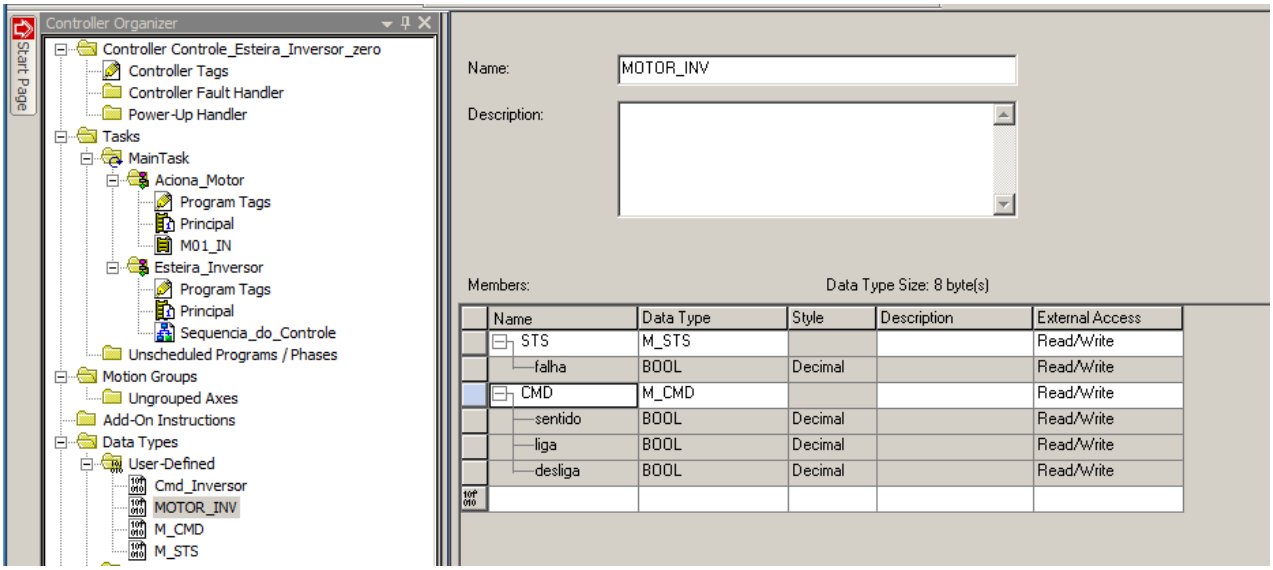


Figura 13. Tipo de estrutura criada para armazenar a informação de status e comando para o motor.

Tabela 3. Descrição da Estrutura ***cmd_Inversor***

Campo	Descrição
Cmd_Inversor.Ligar	Comando para o inversor partir o motor
Cmd_Inversor.Desligar	Comando para o inversor desligar o motor
Cmd_Inversor.Sentido	Comando de sentido de rotação para o inversor
Cmd_Inversor.Velocidade	Comando de frequência de rotação (90 ou 30 Hz) para o inversor

Tabela 4. Descrição da Estrutura ***M01I***

Tipo de informação	Campo	Descrição
status	M01I.STS.falha	Variável que indica falha detectada pelo inversor no acionamento do motor
comando	M01I.CMD.sentido	Comando que define o sentido de rotação do motor
	M01I.CMD.liga	Comando para ligar o motor
	M01I.CMD.desliga	Comando para desligar o motor

A seguir são apresentadas as principais características das rotinas utilizadas no controle da esteira.

A)- Programa: ***Aciona_Motor***

Esse programa é constituído por duas rotinas, como discutido a seguir.

A1)- Rotina 1: ***Principal***

Como indicado na figura 14, essa rotina é constituída por uma única linha que faz a chamada da rotina M01_IN.

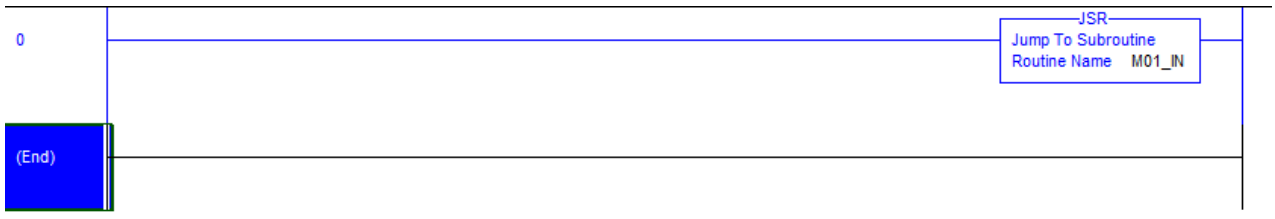


Figura 14. Rotina ***Principal*** do programa ***Aciona_Motor***

A2)- Rotina 2: ***M01_IN***

Essa rotina, escrita em linguagem Ladder e mostrada na figura 15, transfere, de forma adequada, os comandos de Ligar, Desligar, Sentido e Velocidade, para os bits da estrutura responsável pelo controle do inversor.

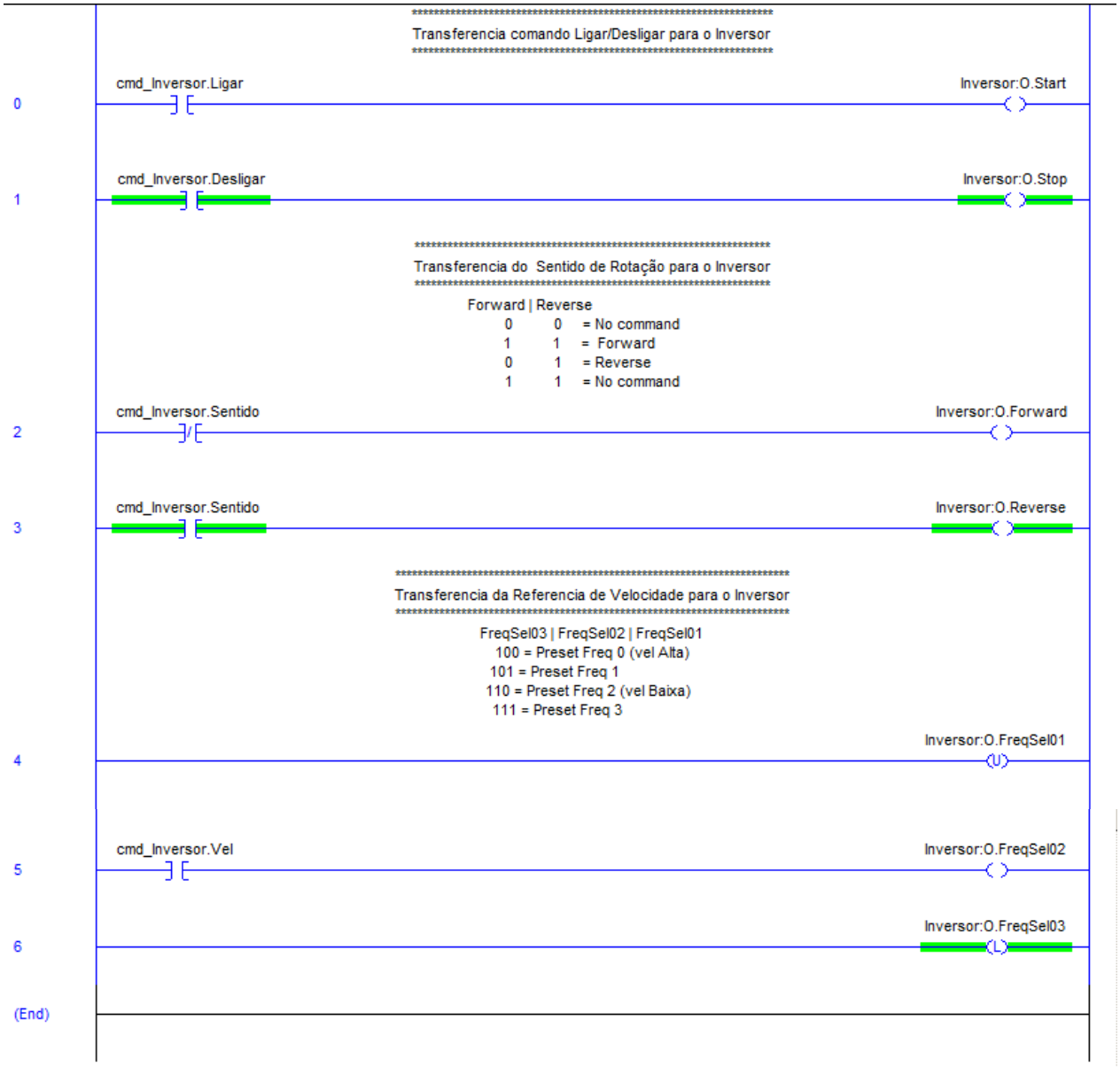


Figura 15. Rotina ***M01_INV*** do programa ***Aciona_Motor***

B)- Programa: ***Esteira_Inversor***

Esse programa também é constituído por duas rotinas (***Principal*** e ***Sequencia_do_Controle***), como discutido a seguir.

Como pode ser observado na figura 16, essa rotina é constituída por três blocos. O primeiro realiza a leitura das entradas (entradas digitais, comandos enviados pela IHM (PanelView) e status do inversor). O segundo bloco faz a chamada da rotina 2 (***Sequencia_do_Controle***), escrita em SFC, responsável por realizar o sequenciamento da movimentação da esteira.

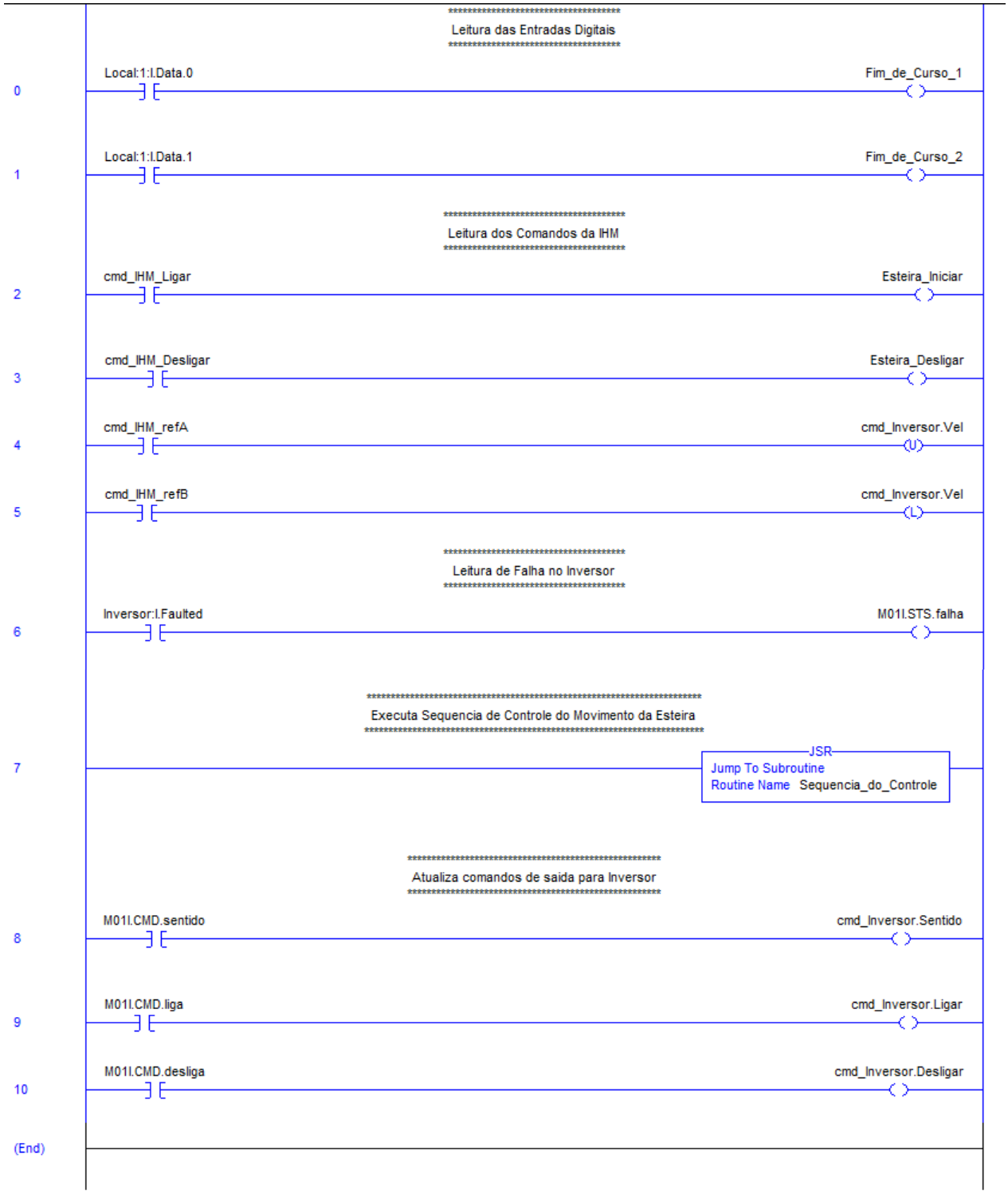


Figura 16. Rotina ***Principal*** do programa ***Esteira_Inversor***

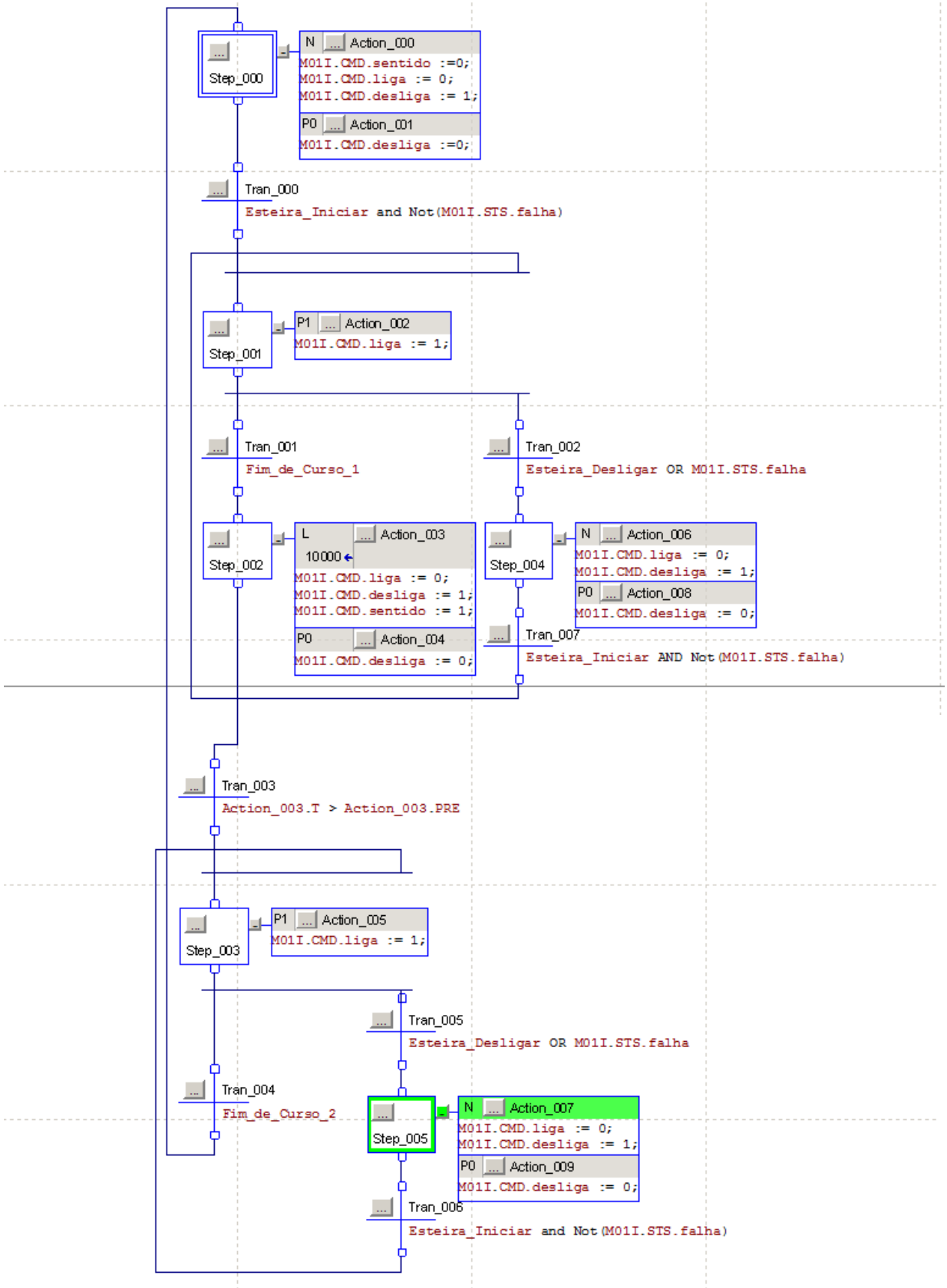


Figura 17. Rotina ***Sequencia_do_Controle*** do programa ***Esteira_Inversor***

2.3 Programação do Panel View

A figura 18 mostra a interface que deverá ser desenvolvida para monitorar e controlar a esteira transportadora através do Panel View. Essa interface será constituída dos seguintes objetos:

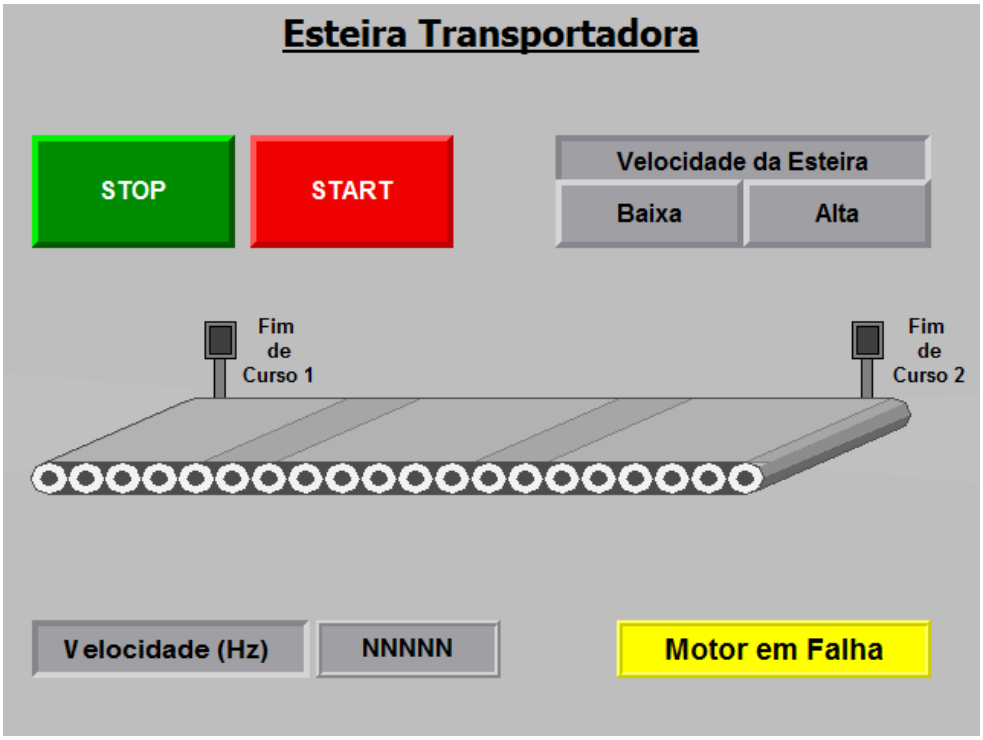


Figura 18. IHM implementada no PanelView para controle da Esteira.

Tabela 5. Objetos da tela de IHM para supervisão e controle da esteira transportadora

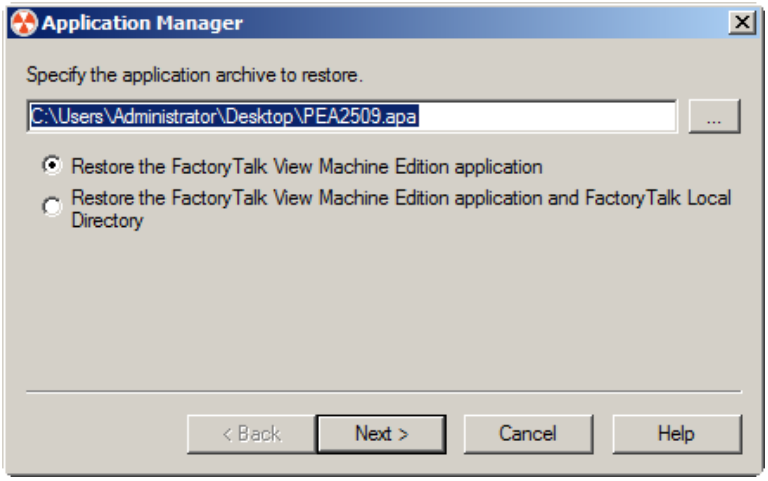
	Função	Objeto do Factory Talk	Tag da aplicação associada
Botão STOP	Parada esteira	Botão Ação Instantânea	Cmd_IHM_Desliga
Botão START	Partida esteira	Botão Ação Instantânea	Cmd_IHM_Liga
Botão Baixa	Aciona o motor em velocidade baixa	Botão Ação Instantânea	cmd_IHM_refA
Botão Alta	Aciona o motor em velocidade alta	Botão Ação Instantânea	cmd_IHM_refB
Fim de Curso 1	Indicação de estado do fim de curso 1	Indicador Multiestado	Fim_de_Curso_1
Fim de Curso 2	Indicação de estado do fim de curso 2	Indicador Multiestado	Fim_de_Curso_2
Mostrador de Velocidade	Mostra a velocidade (em Hz) de acionamento do motor	Mostrador Numérico	PowerFlex40:I.OutputFreq
Falha do Motor	Indicador do bit de falha do inversor	Indicador Multiestado	PowerFlex40:I.Fault (?)

Obs: os objetos de desenho da esteira possuem animação (deslocamento horizontal) para representar a movimentação da esteira.

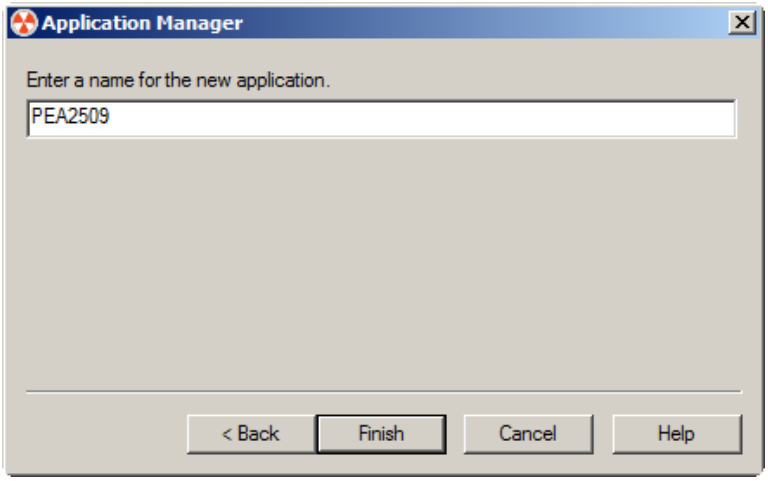
O procedimento para desenvolvimento das telas para as IHM de supervisão e controle de processos, utilizando a ferramenta *FactoryTalk View Studio*, será vista na experiência 4. Um guia rápido com os principais comandos para desenvolvimento de aplicativos nesse software é fornecido na apostila *Tutorial Configuração do* PanelView, disponível no Moodle. No momento, o arquivo de backup do aplicativo a ser utilizado para esta finalidade já se encontra pronto e armazenado na pasta *Desktop\Arquivos de alunos\Exp2\IHM\Tela_Esteira* (arquivo *Tela Esteira.apa*). Para restaurar e carregar esse arquivo siga o procedimento abaixo.

- **Restaurando a aplicação de um arquivo de backup (.apa)**

Para restaurar sua aplicação basta pressionar duas vezes o arquivo de backup que o A*pplication Manager* abrirá. A Tela a seguir deverá aparecer.



Verifique se o caminho mostrado aponta para seu arquivo de backup e se a opção “Restore the FactoryTalk Machine Edition Aplication” está selecionada. Pressione Next. Na próxima tela entre com um nome para sua aplicação e pressione *Finish*.



O A*pplication Manager* fará a restauração de sua Aplicação e fechará. Sua Aplicação, então, estará disponivel no FactoryTalk View Studio (link disponível no desktop da maquina virtual). Na janela *Application Type Selection* deste aplicativo, escolher a opção *Machine Edition* e na janela a seguir, *New/open Application Selection* selecionar o nome da aplicação desejada. A tela inicial do *FactoryTalk View Studio* é mostrada na figura 19.

Obs.: Caso já exista alguma aplicação com o mesmo nome o *Application Manager* não permitirá que o backup seja restaurado, ou seja, deverá ser escolhido um novo nome para sua aplicação.

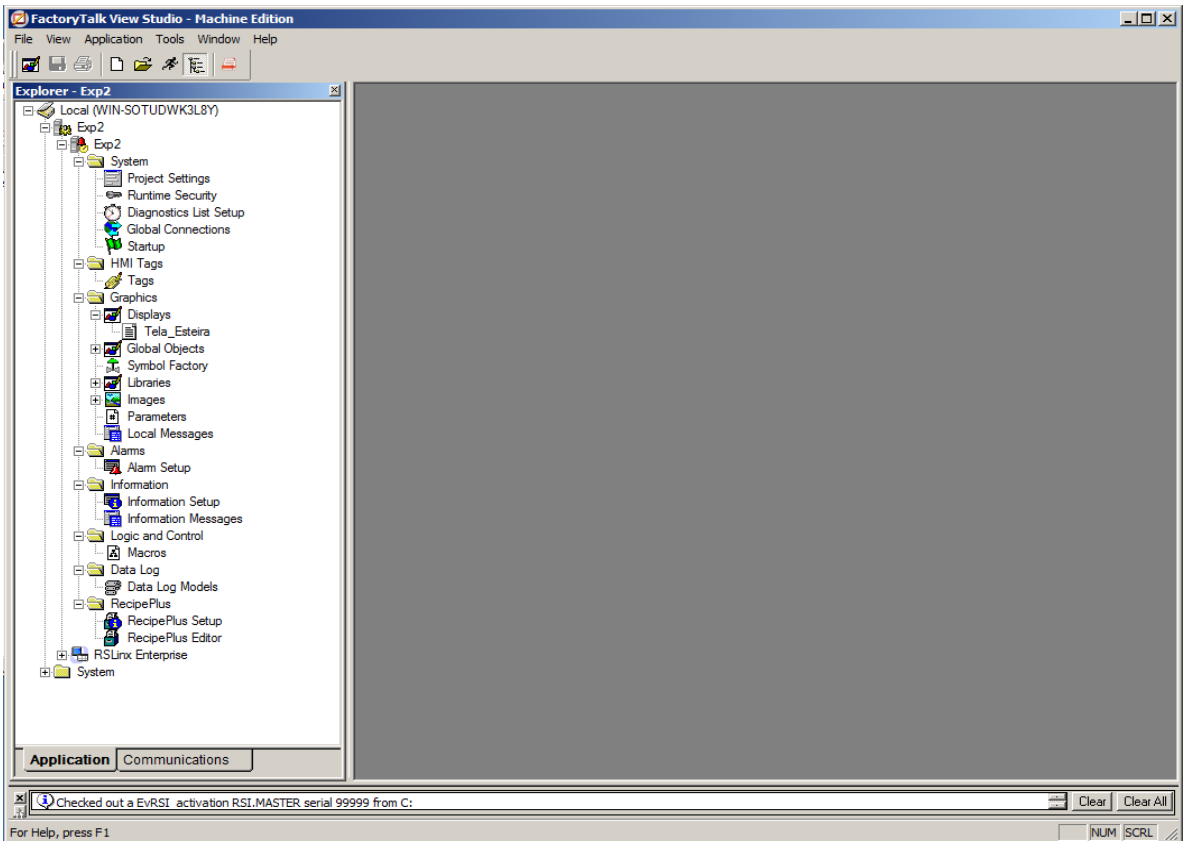


Figura 19. Tela Inicial do *FactoryTalk View Studio*

Na janela Explorer, clique em *Graphics – Displays – Tela Esteira* para abrir a IHM mostrada na figura 18. Explore as propriedades de alguns dos objetos existentes nessa tela. Para tanto, pressione duas vezes o objeto ou clique com o botão direito e em seguida em *Properties*. Verifique o conteúdo das diversas abas, particularmente a aba *Connections* que define a ligação com as tags do CLP. A figura 20 mostra, a título de exemplo, as propriedades do botão *Desligar* da IHM. Note que a tag associada com o botão corresponde ao comando de parada da esteira (*cmd_IHM_Desligar*) utilizado nos aplicativos do CP (vide figura 16).

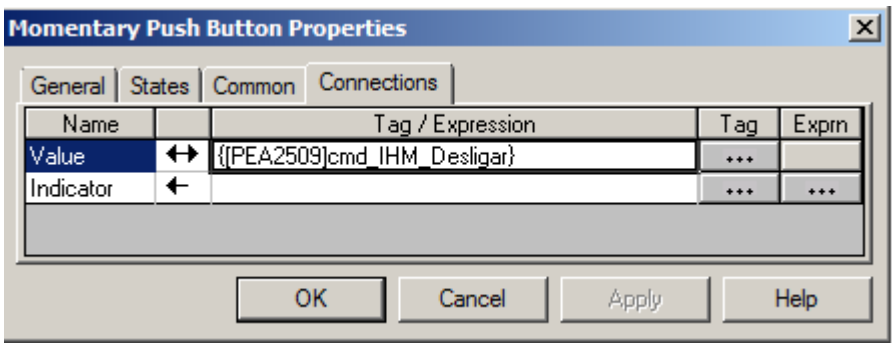


Figura 20. Propriedades do botão *Desligar* da IHM, guia *Connections.*

Clicando no botão "..." abaixo da coluna *Tag*, abriremos a janela *Tag Browser.* Quando corretamente configurado, na parte esquerda dessa janela, *Folders*, navegamos entre diversas pastas que contém os tags da aplicação rodando no CP que podem ser associadas com os elementos de tela. Na tela mostrada na figura 21, na parte *Folders*, clicando em PEA2509 e nas subpastas, tem-se acesso a todas as tags existentes na base de dados do aplicativo do CP que estavam disponíveis quando essa tela foi desenvolvida.

No momento você não consegue visualizar as tags atuais da sua aplicação, uma vez que sua aplicação ainda não foi associada com a tela. Para realizar essa associação, feche as janelas de propriedades e de interface gráfica, certifique-se no software RSLogix que sua **aplicação se encontra corretamente carregada e o CLP está em modo *RUN***. Volte à janela inicial do *Factory Talk View*, mostrada na figura 22, clicar em *RSLinx Enterprise – Comunications Setup*.

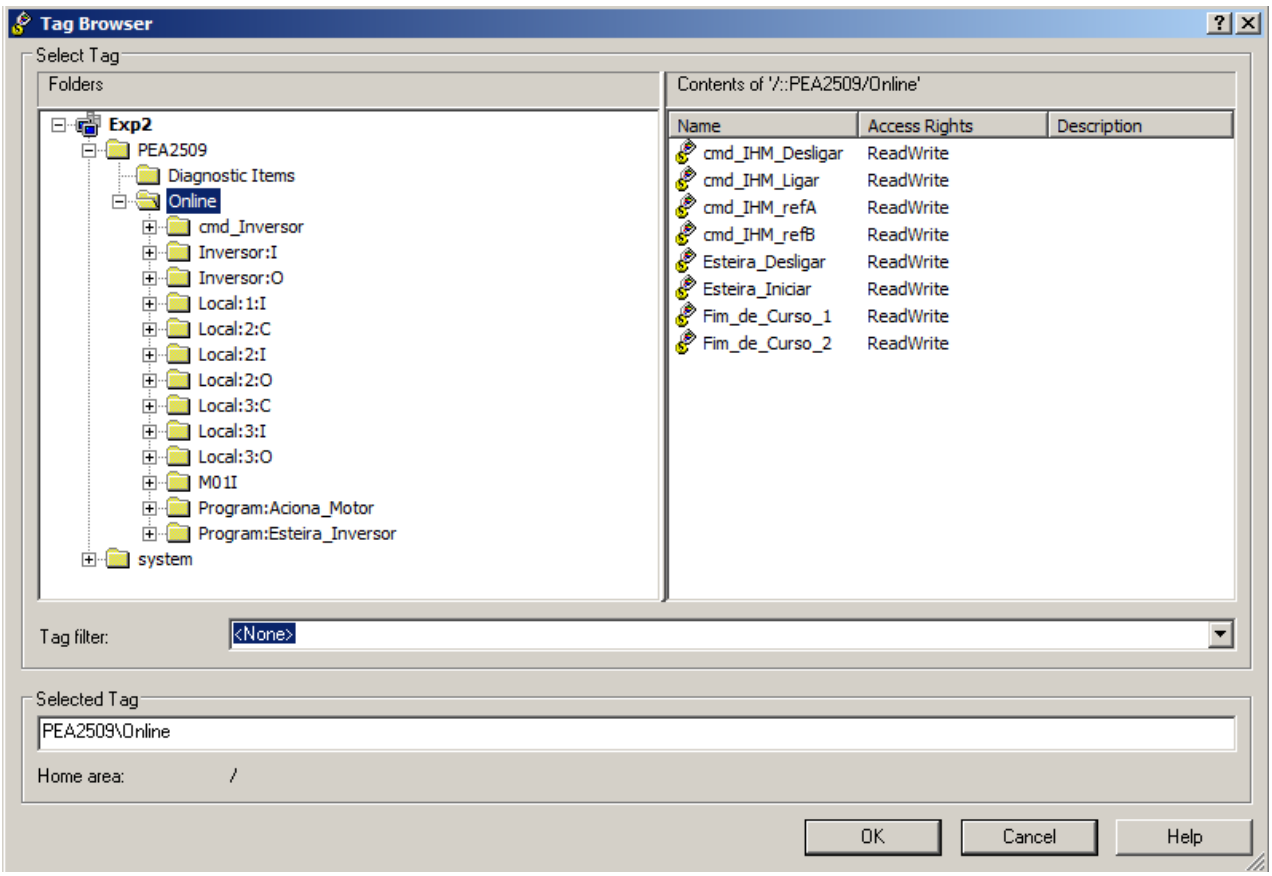


Figura 21. Janela *Tag Browser* do *FactoryTalk View Studio.*

Na janela mostrada na Figura 22, nas guias na parte superior do lado direito, selecionar a guia *Design (Local).* Na tela *Device Shortcuts*, remover todos os itens da janela através da opção *Remove* e

clicar em *Add*. Escolher um nome para o novo atalho, voltar à tela *Design (Local)* e ativar o cartão 1769-L32E/A (dois níveis abaixo do ícone do CP) e clicar no botão *Apply* logo abaixo da parte *Device Shortcuts*. Clicar no botão *Copy from Design to Runtime* e em seguida no botão *OK (*Para que esse procedimento funcione, a programação do CP deverá já estar carregada no mesmo e sendo executada (modo *Run*)).

Em seguida, na tela Folders, clicar com o botão direito no nome da aplicação carregada no CP e selecionar *Refresh All Foders.* Após seguir esse procedimento, de forma semelhante ao mostrado na figura 21, dentro da subpasta *Online* (tela Folders) você deve encontrar todas as variáveis da sua aplicação que podem ser associadas com os objetos da tela em desenvolvimento

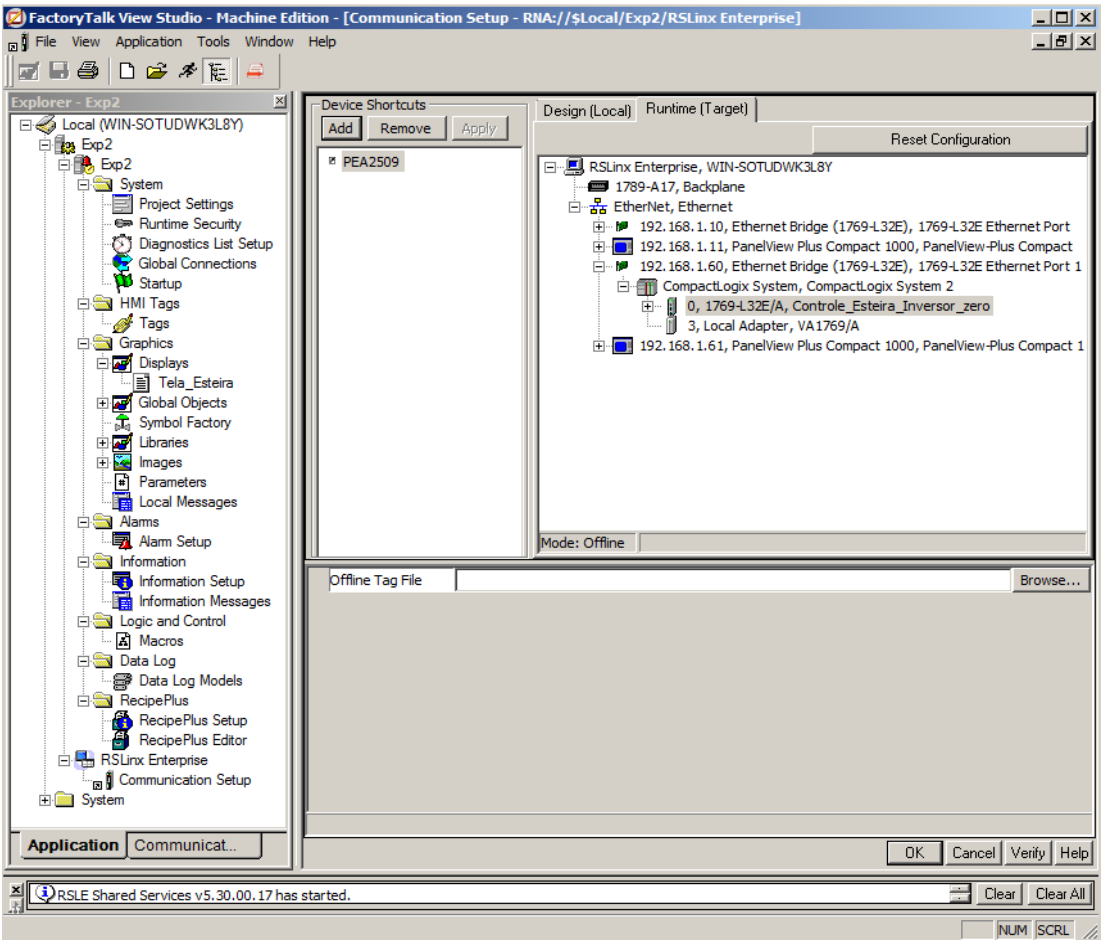


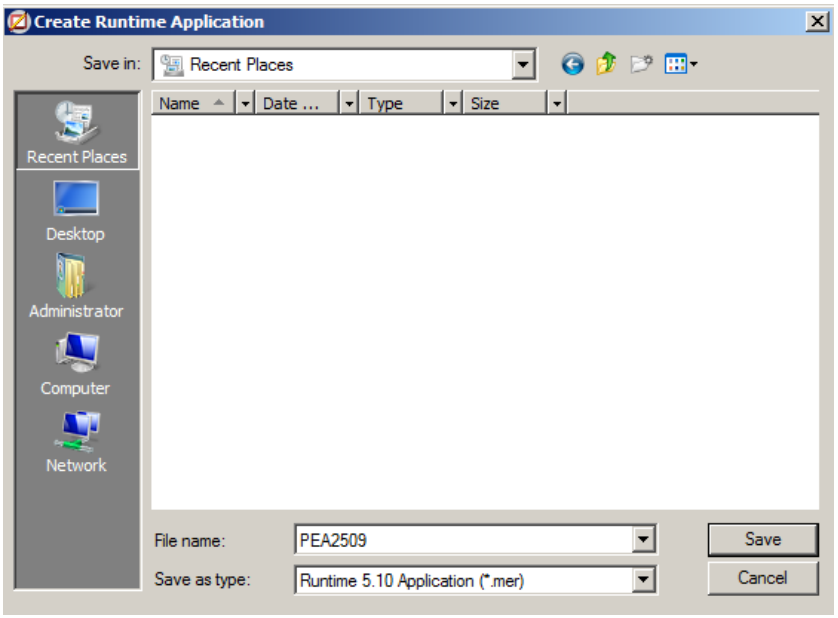
Figura 22. Communication Setup

Após o desenvolvimento da aplicação ser concluída, é necessário fazer o download para a IHM PanelView, segundo o procedimento descrito a seguir.

- **Download de uma Aplicação no PanelView Plus**

Para transferir as telas criadas para um terminal PanelView Plus é necessário criar uma arquivo do tipo .MER. Para tanto siga os seguintes passos.

a) Selecione *Application -> Create Runtime Application*



b) Indique onde o seu projeto será salvo e dê um nome à sua aplicação.

Após a geração do arquivo tipo .MER, basta fazer o download para o PanelView Plus. Para tanto, siga os seguintes passos:

a) Selecione *Tools -> Transfer Utility* (ou pelo ícone )

b) Na tela dessa ferramenta (vide figura23), na opção *Source File* indique onde a sua aplicação está salva e em *Select destination terminal* selecione o seu PanelView Plus. Selecione as opções *Run application at start-up e Replace communications*, para que a aplicação comece a funcionar automaticamente na IHM após o download. A seguir clique em *Download.*

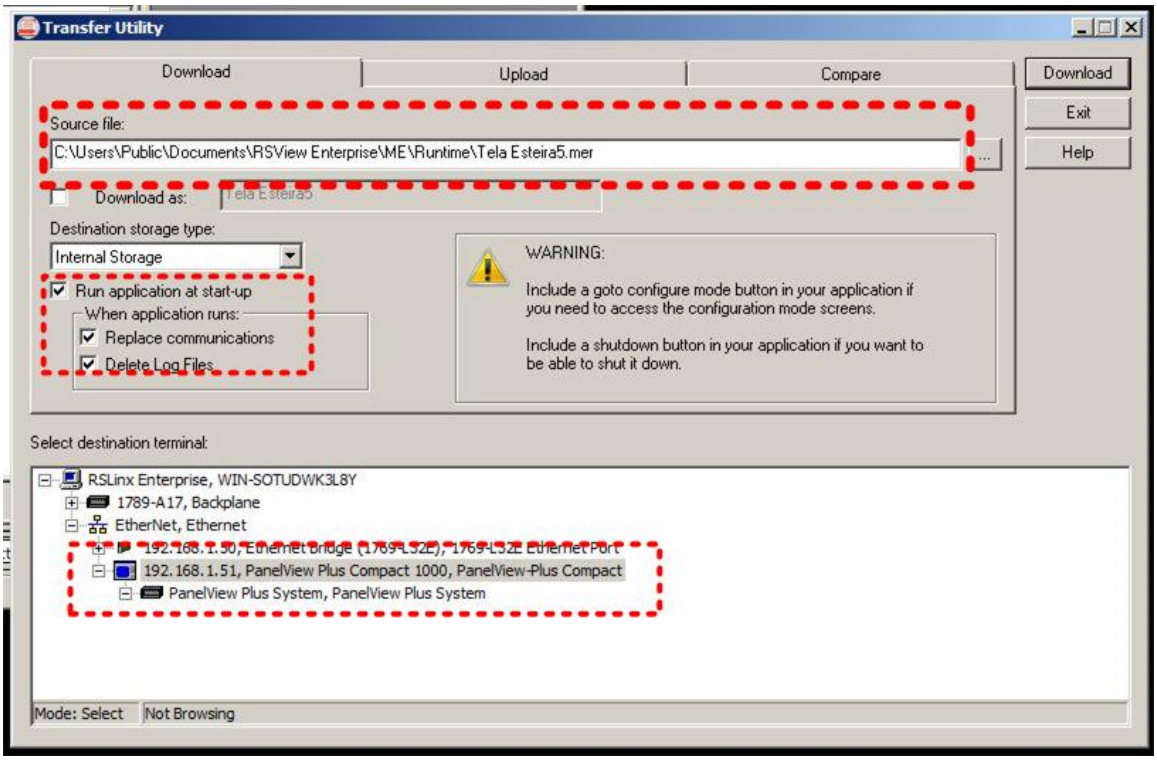


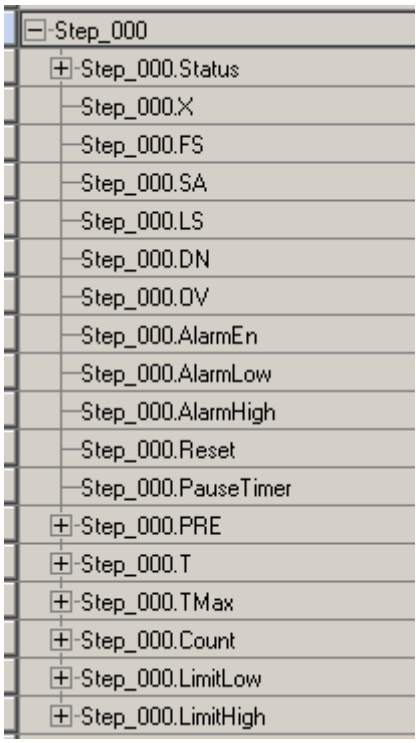
Figura 23. Tela da ferramenta *Transfer Utility*

- **Carregar e iniciar a aplicação no PanelView Plus**

Após concluir o download, na tela *touch-screen* do PanelView, apertar o botão *Load Application[F1]*. Escolher a aplicação carregada através das barras de rolagem e clicar em *Load[F2]* e em *Yes[F7],* caso necessário. A IHM vai voltar para a tela principal, só que desta vez o nome da aplicação estará indicado no campo *Current Application*. Apertar então o botão *Run Application[F2]*. Lembre-se que, para comandar o motor a partir da IHM, deve-se colocar o CP em modo *Remote Run*.

Anexo

- **Linguagem SFC: TAGs associadas com um STEP**



SFC_STEP Structure

Each step uses a tag to provide information about the step. Access this information via either the Step Properties dialog box or the Monitor Tags tab of the Tags window.

If you want to	Then check or set this member	Data type	Details
Determine how long a step has been active (milliseconds)	T	DINT	When a step becomes active, the Timer (T) value resets and then starts to count up in milliseconds. The timer continues to count up until the step goes inactive, regardless of the Preset (PRE) value.
Flag when the step has been active for a specific length of time (milliseconds)	PRE	DINT	Enter the time in the Preset (PRE) member. When the Timer (T) reaches the Preset value, the Done (DN) bit turns on and stays on until the step becomes active again. As an option, enter a numeric expression that calculates the time at runtime.
	DN	BOOL	When the Timer (T) reaches the Preset (PRE) value, the Done (DN) bit turns on and stays on until the step becomes active again.
Flag if a step did not execute long enough	LimitLow	DINT	Enter the time in the LimitLow member (milliseconds). • If the step goes inactive before the Timer (T) reaches the LimitLow value, the AlarmLow bit turns on. • The AlarmLow bit stays on until you reset it. • To use this alarm function, turn on (check) the AlarmEnable (AlarmEn) bit. As an option, enter a numeric expression that calculates the time at runtime.
	AlarmEn	BOOL	To use the alarm bits, turn on (check) the AlarmEnable (AlarmEn) bit.
	AlarmLow	BOOL	If the step goes inactive before the Timer (T) reaches the LimitLow value, the AlarmLow bit turns on. • The bit stays on until you reset it. • To use this alarm function, turn on (check) the AlarmEnable (AlarmEn) bit.

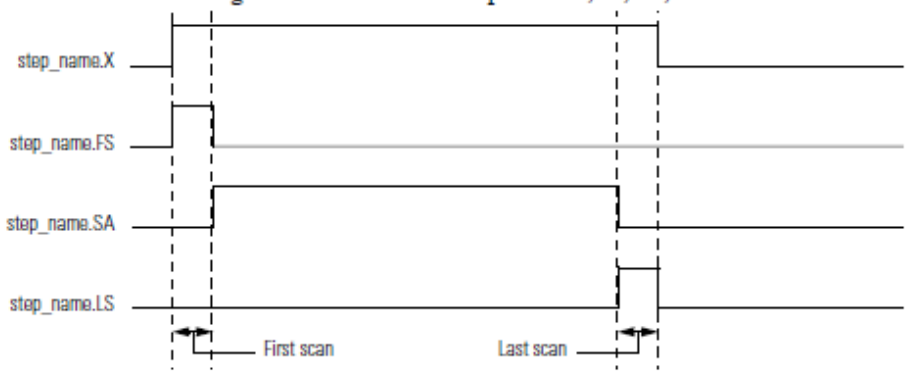
If you want to	Then check or set this member	Data type	Details
Flag if a step is executing too long	LimitHigh	DINT	Enter the time in the LimitHigh member (milliseconds). • If the Timer (T) reaches the LimitHigh value, the AlarmHigh bit turns on. • The AlarmHigh bit stays on until you reset it. • To use this alarm function, turn on (check) the AlarmEnable (AlarmEn) bit. As an option, enter a numeric expression that calculates the time at runtime.
	AlarmEn	BOOL	To use the alarm bits, turn on (check) the AlarmEnable (AlarmEn) bit.
	AlarmHigh	BOOL	If the Timer (T) reaches the LimitHigh value, the AlarmHigh bit turns on. • The bit stays on until you reset it. • To use this alarm function, turn on (check) the AlarmEnable (AlarmEn) bit.
Do something while the step is active (including first and last scan)	X	BOOL	The X bit is on the entire time the step is active (executing). Typically, we recommend that you use an action with a N Non-Stored qualifier to accomplish this.
Do something one time when the step becomes active	FS[1]	BOOL	The FS bit is on during the first scan of the step. Typically, we recommend that you use an action with a P1 Pulse (Rising Edge) qualifier to accomplish this.
Do something while the step is active, *except* on the first and last scan	SA	BOOL	The SA bit is on when the step is active except during the first and last scan of the step.
Do something one time on the last scan of the step	LS[1]	BOOL	The LS bit is on during the last scan of the step. Use this bit only if on the Controller Properties dialog box, SFC Execution tab, you set the Last Scan of Active Step to Don't Scan or Programmatic reset. Typically, we recommend that you use an action with a P0 Pulse (Falling Edge) qualifier to accomplish this.
Determine the target of an SFC Reset (SFR) instruction	Reset	BOOL	An SFC Reset (SFR) instruction resets the SFC to a step or stop that the instruction specifies. • The Reset bit indicates to which step or stop the SFC will go to begin executing again. • Once the SFC executes, the Reset bit clears.
Determine the maximum time that a step has been active during any of its executions	TMax	DINT	Use this for diagnostic purposes. The controller clears this value only when you select the Restart Position of Restart at initial step and the controller changes modes or experiences a power cycle.
Determine if the Timer (T) value rolls over to a negative value	OV	BOOL	Use this for diagnostic purposes.

If you want to	Then check or set this member	Data type	Details
Determine how many times a step has become active	Count	DINT	This is not a count of scans of the step. • The count increments each time the step becomes active. • It increments again only after the step goes inactive and then active again. • The count resets only if you configure the SFC to restart at the initial step. With that configuration, it resets when the controller changes from program mode to run mode.
Use one tag for the various status bits of this step	Status	DINT	**For this member** / **Use this bit**

For this member	Use this bit
Reset	22
AlarmHigh	23
AlarmLow	24
AlarmEn	25
OV	26
DN	27
LS	28
SA	29
FS	30
X	31

(1) The FS and LS bits are only active during a step's execution. Once a step finishes executing the code within its actions, the FS and/or LS bits are reset. If you reference either of these bits in code outside of the SFC routine in a different part of the project, the bits are always cleared (0).

This diagram shows the relationship of the X, FS, SA, and LS bits.



- **Linguagem SFC: Qualificador para uma Action**

To change when an action starts or stops, assign a different qualifier.

Choose a Qualifier for an Action

If you want the action to	And	Then assign this qualifier	Which means
Start when the step is activated	Stop when the step is deactivated	N	Non-Stored
	Execute only once	P1	Pulse (Rising Edge)
	Stop before the step is deactivated or when the step is deactivated	L	Time Limited
	Stay active until a Reset action turns off this action	S	Stored
	Stay active until a Reset action turns off this action Or a specific time expires, even if the step is deactivated	SL	Stored and Time Limited
Start a specific time after the step is activated and the step is still active	Stop when the step is deactivated	D	Time Delayed
	Stay active until a Reset action turns off this action	DS	Delayed and Stored
Start a specific time after the step is activated, even if the step is deactivated before this time	Stay active until a Reset action turns off this action	SD	Stored and Time Delayed
Execute once when the step is activated	Execute once when the step is deactivated	P	Pulse
Start when the step is deactivated	Execute only once	P0	Pulse (Falling Edge)
Turn off (reset) a stored action • S Stored • SL Stored and Time Limited • DS Delayed and Stored • SD Stored and Time Delayed	⟶	R	Reset