

Programação em Ladder

Composição de um programa em *Ladder*

Um programa escrito em *Ladder* é constituído por um conjunto de sequências (*rungs*) que são executados sequencialmente pelo autómato. Uma sequência é composta por um conjunto de elementos gráficos limitados à esquerda e à direita por linhas de energia (*power rails*). Os elementos gráficos representam:

- ? I/O do autómato (interruptores, sensores, indicadores, relés, etc.).
- ? Blocos funcionais (temporizadores, contadores, etc.).
- ? Operações aritméticas e lógicas.
- ? Variáveis internas do autómato.

Cada sequência contém no máximo 7 linhas e 11 colunas que se encontram divididas em na zona de teste, onde se encontram as condições necessárias para a execução das acções, e na zona de actuação, onde se encontram as acções que são executadas dependendo do resultado da zona de teste.

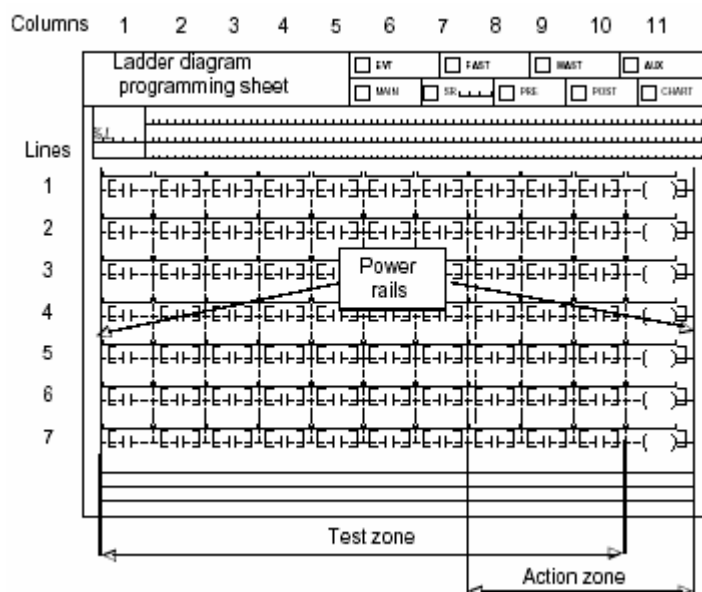


figura 9 – estrutura de uma sequência

Os objectos gráficos estão divididos em três categorias – básicos, blocos funcionais e blocos de operação – e encontram-se representados e descritos nas tabelas seguintes.

Designation		Symbol	Function
Test elements	• Normally open contact		Contact closed when the bit object which controls it is at 1.
	• Normally closed contact		Contact closed when the bit object which controls it is at 0.
	• Edge detection contacts		Rising edge : contact closed when the bit object which controls it changes from 0 to 1.
			Falling edge : contact closed when the bit object which controls it changes from 1 to 0.
Link elements	• Horizontal links		Used to link test and action graphic elements between the two power rails in series.
	• Vertical links		Used to link test and action graphic elements in parallel.
Action elements	• Direct coil		Sets the associated bit object to the value of the result of the test zone.
	• Negated coil		Sets the associated bit object to the inverse value of the result of the test zone.
	• Latch coil		Sets the associated bit object to 1 when the result of the test zone is at 1.
	• Unlatch coil		Resets the associated bit object to 0 when the result of the test zone is at 1.
	• Conditional JUMP to another rung		Allows connection to a labelled rung, either upstream or downstream. Jumps are only effective within the same programming entity (main program, subroutine, etc). If a jump is activated : <ul style="list-style-type: none"> • Scanning of the current rung is interrupted. • The requested labelled rung is executed. • The part of the program between the jump action and the designated rung is not executed.
	• Transition condition coil		Offered in Grafcet language, used when programming conditions associated with transitions, to move to the next step.

figura 10 – elementos gráficos

Uma sequência pode ainda conter uma etiqueta e comentários. Uma etiqueta (%L) é utilizada para identificar uma sequência no programa ou rotina mas não é obrigatória. As etiquetas são também utilizadas para permitir saltos entre sequências. Os comentários são integrados nas sequências e permitem uma melhor compreensão mas não são obrigatórios.






Designation		Symbol	Function
Action elements (continued)	• Subroutine call coil (CALL)		Allows connection at the beginning of subroutines when the result of the test zone is at 1. If a subroutine is called : • Scanning of the current rung is interrupted. • The subroutine is executed. • Scanning of the interrupted rung is resumed.
	• Subroutine return	<RETURN>	Reserved for subroutines SR, allows return to the calling module when the result of the test zone is at 1.
	• Stop program	<HALT>	Stops execution of the program when the result of the test zone is at 1.
Function blocks			
Designation		Symbol	Function
Test elements	• Blocks : Timer Counter Monostable Register Drum controller		Each of the standard function blocks uses I/O which allow them to be linked to other graphic elements. The functions of each block are described in part B. Size : see section 2.2-5.
Operation blocks			
Designation		Symbol	Function
Test elements	• Vertical comparison block		Allows comparison of 2 operands. Depending on the result, the corresponding output changes to 1. Size : 2 columns/4 lines.
	• Horizontal comparison block		Allows comparison of 2 operands. The output changes to 1 when the result is checked. (A block can contain up to 4096 characters). Size : 2 columns/1 line.
Action elements	• Operation block	 -	Performs arithmetic, logic operations etc and uses Structured text language syntax. (A block can contain up to 4096 characters). Size : 4 columns/1 line.

figura 11 – elementos gráficos (continuação)

Exemplo de criação e edição de um programa em *Ladder*

Pretende-se desenvolver o sistema de controlo para o depósito (apresentado para o caso da programação em IL) baseado num autómato programável cuja programação deve ser efectuada através da linguagem *Ladder*.

Implementação do Sistema de Controlo

Comece por criar uma nova aplicação. Arranque o *software PL7 Junior* e seleccione *File/New*. Identifique o autómato programável com que esta a trabalhar – TSX Micro 37-21/22 V2.0 – e seleccione *No* na opção de Grafcet.

Configure a aplicação para o seu autómato, defina as variáveis e a tabela de animação. Na janela *Application Browser* seleccione *STATION/ Configuration/ Hardware Configuration*:

? Adicione o módulo TSX DMZ 28 DR na posição 1.

? Confirme a alteração no botão .

Na janela *Application Browser* seleccione *STATION/ Variables/ I/O*:

? No modulo 1: TSX DMZ 28 DR defina as entradas.

? No modulo 2: TSX DMZ 28 DR defina as saídas.

Entradas	Saídas
ARRANQUE (%I1.1)	BOMBA (%Q2.1)
PARAGEM(%I1.2)	LUZ (%Q2.2)
SUPERIOR (%I1.3)	
INFERIOR (%I1.4)	

tabela 2 – entradas e saídas do controlador

Na janela *Application Browser* seleccione *STATION/ Variables/ Memory Objects*:


? Defina uma variável interna para guardar o estado do sistema - ESTADO (%M0).

Para este exemplo vamos utilizar um bloco funcional pré definido do autómato: o temporizador. A sua utilização passa por uma definição prévia de um conjunto de valores.

No caso do temporizador (%TM0) é necessário definir o valor da temporização, o modo de funcionamento (TP, TON ou TOF) e a base de tempo associada. Na janela *Application Browser* seleccione *STATION/ Variables/ PredefinedFB*:

- ? Seleccione a opção *Parameters* e o tipo TM.
- ? Defina o valor *Preset* para a temporização pretendida (5 segundos), seleccione o modo *TP* e escolha para base de tempo (*TB*) 1 sec.

Na janela *Application Browser* seleccione *STATION/ Animation Table/ Create*:

- ? Adicione as variáveis de entrada e de saída à tabela de animação.
- ? Adicione as variáveis internas à tabela de animação.
- ? Adicione o valor actual (.v) do bloco pré-definido à tabela de animação.
- ? Confirme a alteração no botão .

Edição do Programa

Abra o editor em *Station/ Program/ MAST Task/ Main*. Seleccione a linguagem de programação LD. No editor, introduza o programa, dividido em diversas sequências, correspondente ao esquema de funcionamento do sistema. Para o exemplo actual, uma solução possível para o programa seria a apresentada na figura 6.

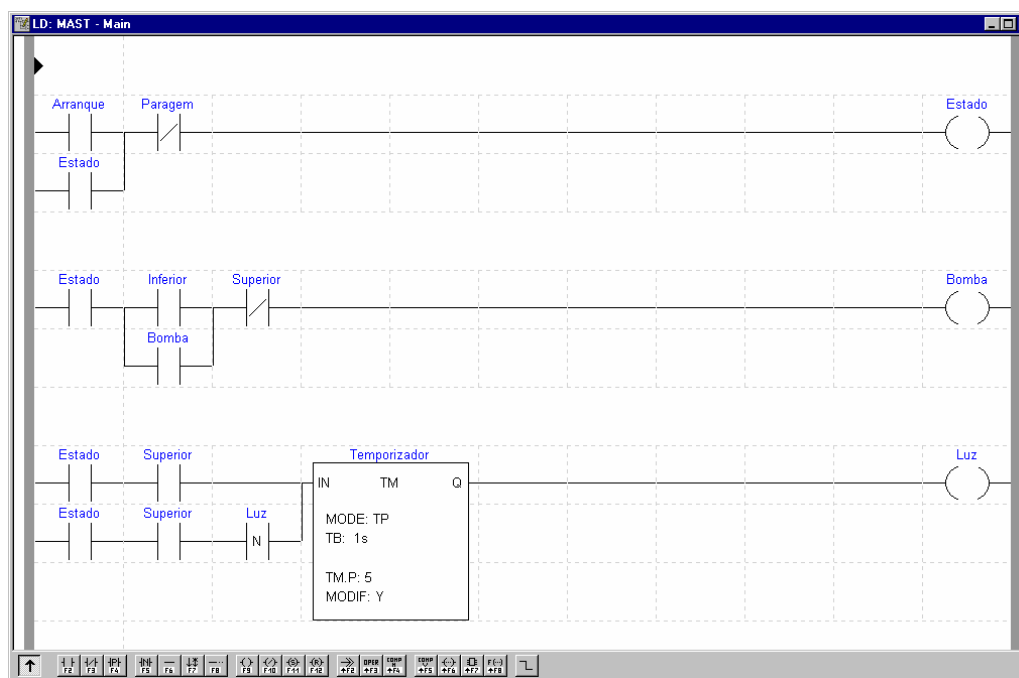



figura 12 – programa em LD

Após terminar a introdução de uma sequência do programa deve validá-la no botão  ou através do menu *Edit/Confirm*. Quando uma sequência é confirmada o seu aspecto altera-se passando o texto de vermelho para preto.

Teste e experimentação do sistema de controlo

Utilize o kit “Parque Automóvel” para simular o funcionamento do depósito. Considere que o botão BE representa o botão ARRANQUE e o botão BS representa o botão PARAGEM. Inclua dois botões exteriores para representar os sensores SUPERIOR e INFERIOR. Utilize o conjunto de *leds* CANCELA ABERTA para representar a bomba (BOMBA), o *led* SEMÁFORO VERDE para representar o sinal luminoso (LUZ).

Depois de realizar as ligações indicadas entre o PLC e o kit de simulação estabeleça a ligação entre o computador e o PLC – menu *PLC/ Connect* (ou CTRL + K). Seleccione a opção de transferir do PC para o PLC. Uma vez efectuada a transferência (indicação ON LINE na barra de estados) dê ordem de arranque ao autómato – menu *PLC/ Run* (ou CTRL + SHIFT + R).

Animação

Quando o autómato se encontra em modo **Run** é possível visualizar no editor de LD a evolução do estado das diferentes variáveis. Em simultâneo pode utilizar a tabela de animação que definiu inicialmente para observar a evolução das variáveis de entrada e saída.

Programação em Grafcet

Composição de um programa em Grafcet (GR7)

O Grafcet é uma linguagem que permite descrever sistemas de controlo sequenciais de forma gráfica e estruturada. Esta descrição é efectuada utilizando objectos gráficos que representam:

- ? Etapas – às quais podem ser associadas acções³.
- ? Transições – às quais podem ser associadas condições de transição.
- ? Arcos direccionados – ligam uma etapa a uma transição ou uma transição a uma etapa.

Editor gráfico

O diagrama é construído utilizando o editor gráfico que se encontra na figura seguinte.

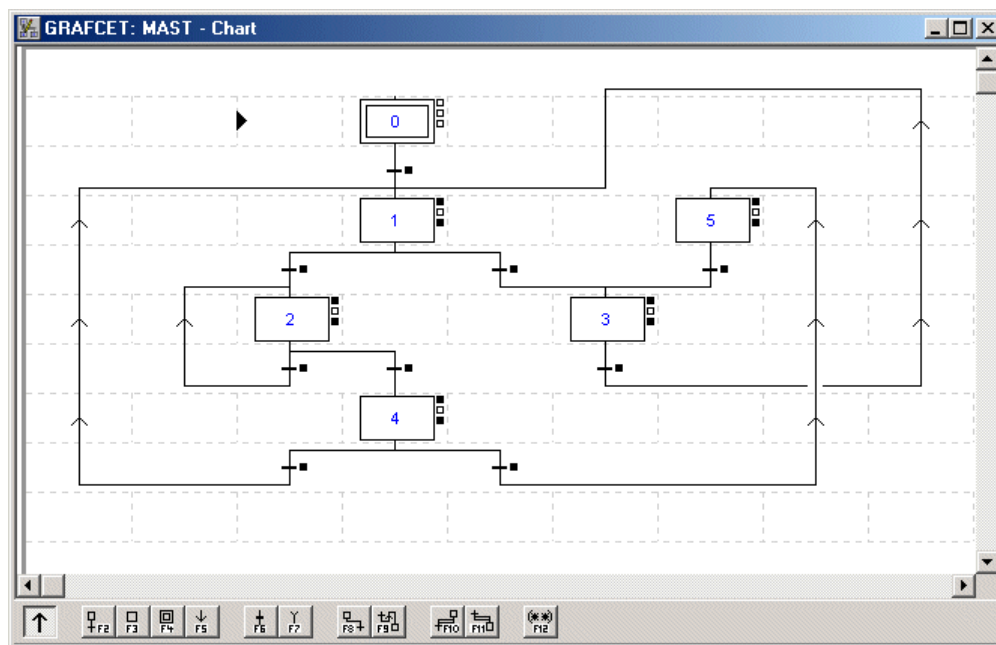


figura 13 – Editor de GR7 do PL7 Junior

³ Indicam o que deve ser realizado quando a etapa associada está activa (em particular descrevem os comandos que devem ser enviados para o sistema físico ou para outros sistemas de controlo).

O editor suporta 8 páginas, referenciadas de 0 a 7 na barra de estado. Cada página apresenta a forma de uma matriz com 14 linhas e 11 colunas que definem 154 células. Uma mesma página pode conter vários diagramas. Apenas pode ser colocado um objecto – etapa, transição, etc. – por célula.

Cada página do editor tem dois tipos de linhas:

- ? Linhas de etapas – onde podem ser colocadas etapas, macro-etapas⁴ e conectores.
- ? Linhas de transição – onde podem ser colocadas transições e conectores fonte.

Os comentários são objectos independentes, que não estão associados a etapas ou transições, que podem ser introduzidos em qualquer um dos tipos de linha.

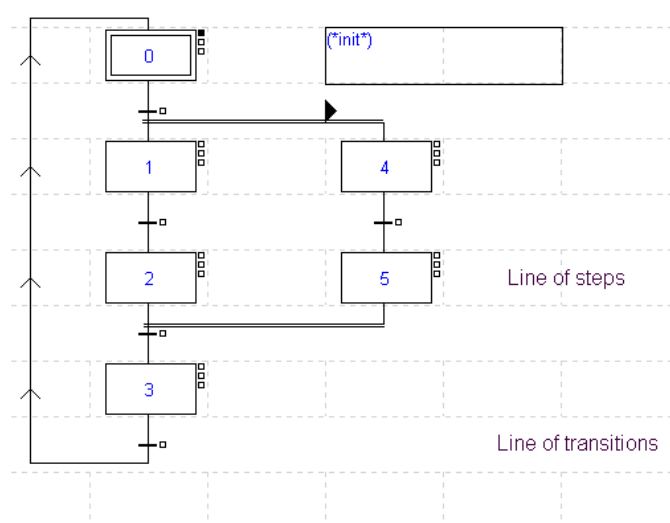


figura 14 – Página de GR7

Objectos gráficos

Os objectos gráficos que podem ser utilizados para construir os diagramas (apresentados da figura 15 à figura 20) são os seguintes:

- ? Etapas (inicial ou simples) – podem-se associar acções (expressas em LD, ST ou IL).
- ? Etapa + transição.
- ? Transições – são associadas condições de transição (receptividades) às transições (expressas em LD, ST ou IL).
- ? Arcos orientados.
- ? Conectores.
- ? Comentários.

⁴ Não suportado pelo TSX 37-21/22

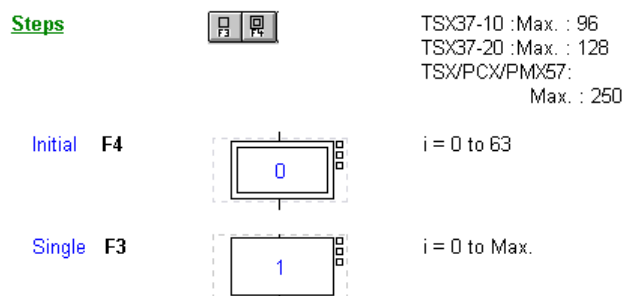


figura 15 - etapas

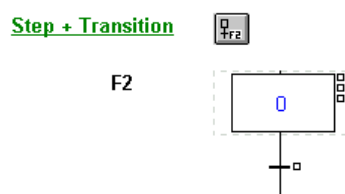


figura 16 – etapa + transição

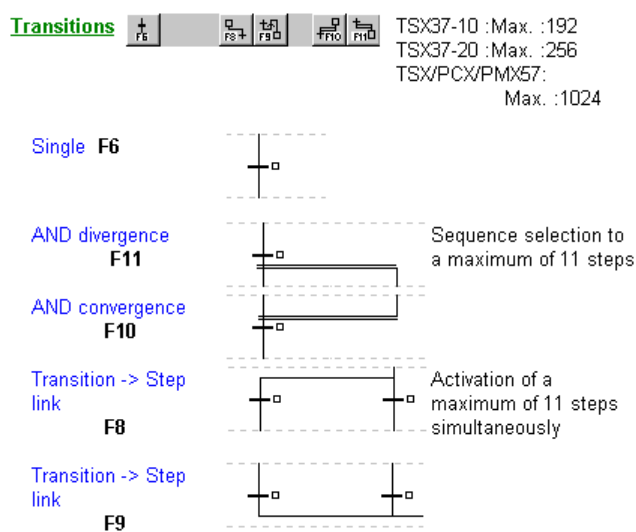


figura 17 - transições

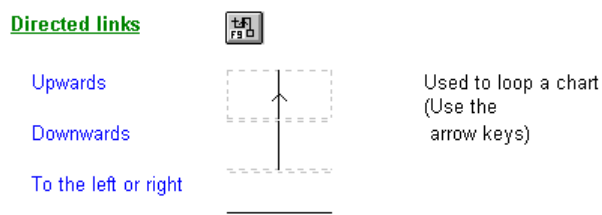


figura 18 – arcos orientados

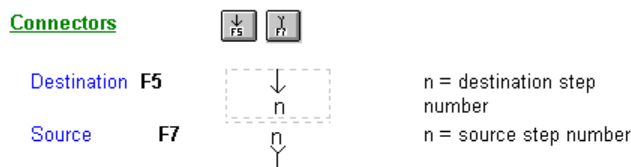


figura 19 - conectores



figura 20 – comentários

Objectos específicos do GR7

Existe um conjunto de objectos que são específicos da programação em GR7:

- ? Objectos (*bits*) associados às etapas - %Xi.
- ? Objectos (*bits*) do sistema associados ao GR7 - %S21, %S22, %S23 e %S26.
- ? Objectos (*words*) associados às etapas e que indicam o tempo de actividade das mesmas - %Xi.T.
- ? Objectos (*words*) do sistema associadas ao GR7 - %SW20 e %SW21.

Os objectos associados às etapas - %Xi - tomam o valor 1 quando a respectiva etapa está activa. O seu valor pode ser testado em todas as tarefas de processamento mas apenas pode ser modificado no pré processamento da tarefa principal.

Os objectos que indicam o tempo de actividade das etapas - %Xi.T - são incrementados de 100 em 100 *ms* e apresentam à quanto tempo a etapa está activa em (*ms*). Quando uma etapa é activada, o conteúdo deste objecto é colocado a zero e passa a ser incrementado enquanto esta se mantiver activa. Quando uma etapa é desactivada o seu conteúdo é mantido.

Acções associadas com etapas

Cada etapa pode ter acções associadas (programadas em LD, ST ou IL) que apenas são executadas enquanto a etapa a que se encontram associadas estiver activa. São admissíveis três tipos de acções que podem ser utilizadas em simultâneo numa mesma etapa:

- ? Acções na activação – executadas assim que a etapa a que estão associadas se torna activa (e apenas nessa altura).
- ? Acções contínuas – executadas continuamente enquanto a etapa a que estão associadas se encontra activa.

- ? Acções na desactivação – executadas quando a etapa a que estão associadas é desactivada (e apenas nessa altura).

As regras de programação das etapas são as seguintes:

- ? Todas as acções são tratadas como acções memorizadas, consequentemente:
 - ? Uma acção que é controlada pela duração de uma etapa X_n deve ser desactivada quando da desactivação da etapa X_n ou da activação da etapa X_{n+1} .
 - ? Uma acção que afecta várias etapas é activada na activação da etapa X_n e desactivada na desactivação da etapa X_{n+m} .
- ? Todas as acções podem ser controladas por condições lógicas.
- ? As acções que são controladas por esquemas de segurança ou modos de funcionamento devem ser programadas no pós processamento.

Condições de transição (receptividade)

Cada transição tem associada condições de transição (programada em LD, ST ou IL) que apenas são avaliadas quando a transição a que estão associadas está validada (receptiva). No **PL7 Junior** uma condição de transição não programada é sempre avaliada como falsa.

Quando as condições de transição são programadas em IL é necessário considerar algumas diferenças para a utilização normal da linguagem:

- ? Não é utilizada a etiqueta %L.
- ? Não podem ser utilizadas instruções de acção.
- ? Não são permitidos saltos nem chamadas a subrotinas.

Estrutura de um programa em Grafcet (*single task*)

O programa numa aplicação single task está associado a uma única tarefa principal: MAST. O programa associado com esta tarefa principal pode ser estruturado em diversos módulos. Dependendo ou não de se estar a utilizar GR7, existem duas alternativas.

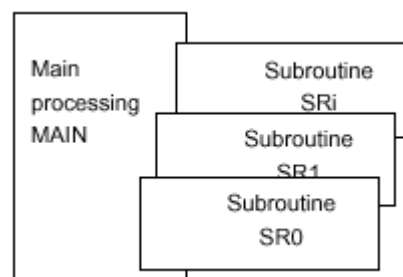


figura 21

Sem Grafcet

- ? Processo principal (MAIN).
- ? Subrotina SRi (i=0 até 253): as subrotinas são módulos programados independentemente, podendo ser chamadas do processo principal ou de outras subrotinas (é possível ter até 8 níveis encadeados de subrotinas).

Com Grafcet

- ? Pré processamento (PRL): executado antes do Grafcef, utilizado para processar a lógica de entrada e inicializar o Grafcet.
- ? Grafcet (CHART): condições de transição associadas às transições e acções associadas às etapas são programadas nas páginas do Grafcet.
- ? Pós processamento (POST): executado depois do Grafcet, utilizado para processar a lógica de saída, monitorizar e definir esquemas de segurança.
- ? Subrotina SRi (i=0 até 253): as subrotinas são módulos programados independentemente, podendo ser chamadas do processo principal ou de outras subrotinas (é possível ter até 8 níveis encadeados de subrotinas).

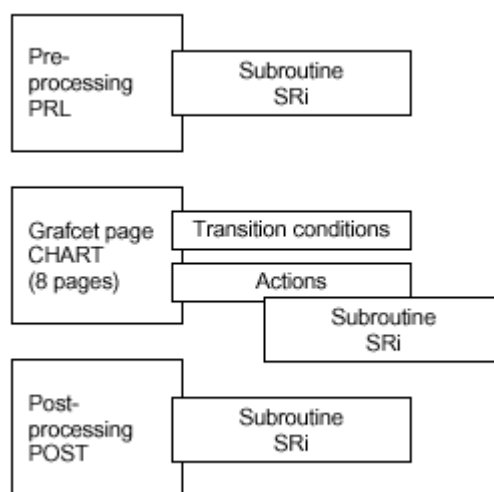


figura 22

Exemplo de criação e edição de um programa em GR7

Descrição do sistema - Portão de Garagem

O portão de uma garagem é accionado por um motor que pode ser comandado remotamente. O sistema de comando dispõe de 2 sensores de fim de curso, do tipo normalmente aberto, que indicam se o portão está completamente fechado (FECHADO) ou completamente aberto (ABERTO). O motor pode ser controlado quanto ao sentido de abertura (ABRIR) ou fecho (FECHAR) do portão. O comando à distância dispõe de um botão de pressão (COMANDO) do tipo normalmente aberto. O funcionamento do sistema é o seguinte:

- ? Quando é pressionado o botão do comando e o portão está fechado é accionado o motor no sentido ABRIR. O motor manterá este sentido de funcionamento até ser detectado que o portão está aberto ou voltar a ser pressionado o botão do comando.
- ? Quando é pressionado o botão do comando e o portão está aberto é accionado o motor no sentido FECHAR. O motor manterá este sentido de funcionamento até ser detectado que o portão está fechado ou voltar a ser pressionado o botão do comando.
- ? Se, enquanto o portão se encontra em movimento (em qualquer um dos sentidos), voltar a ser pressionado o botão do comando o motor para imediatamente. Voltando a pressionar o botão do comando o motor é accionado no sentido contrário ao que se deslocava anteriormente (se estava a FECHAR passa a ABRIR; se estava a ABRIR passa a FECHAR).

Pretende-se desenvolver o sistema de controlo para este portão baseado num autómato programável cuja programação deve ser efectuada através de Grafcet.

Implementação do sistema de controlo

Comece por criar uma nova aplicação. Arranque o *software* **PL7 Junior** e seleccione *File/New*. Identifique o autómato programável com que esta a trabalhar – TSX Micro 37-21/22 V2.0 – e seleccione *Yes* na opção de Grafcet.

Configure a aplicação para o seu autómato, defina as variáveis e a tabela de animação. Na janela *Application Browser* seleccione *STATION/Configuration/Hardware Configuration*:

- ? Adicione o módulo TSX DMZ 28 DR na posição 1.

? Confirme a alteração no botão .

Na janela *Application Browser* seleccione *STATION/ Variables/ I/O*:

? No modulo 1: TSX DMZ 28 DR defina as entradas.

? No modulo 2: TSX DMZ 28 DR defina as saídas.

Entradas	Saídas
ABERTO (%I1.1)	ABRIR (%Q2.1)
FECHADO (%I1.2)	FECHAR (%Q2.2)
COMANDO (%I1.5)	

tabela 3 – entradas e saídas do controlador

Na janela *Application Browser* seleccione *STATION/ Animation Table/ Create* :

? Adicione as variáveis de entrada e de saída à tabela de animação

? Confirme a alteração no botão .


Edição do programa

Abra o editor em *Station/Program/MAST Task/Chart*. Utilizando a barra de ferramentas que se encontra no fundo da janela de edição construa o diagrama correspondente ao esquema de funcionamento sequencial do sistema.



figura 23 – barra de ferramentas do editor

Para o exemplo actual, o GR7 correspondente deve apresentar o seguinte aspecto.

Após terminar a construção do diagrama deve validá-lo no botão  ou através do menu *Edit/Confirm*. Quando um diagrama é confirmado o seu aspecto altera-se: os objectos passam de vermelho para preto e a delimitação das páginas torna-se cinzenta.

Neste altura podem ser definidas as condições de transição (para as diferentes transições) e programadas as acções (associadas às diferentes etapas).

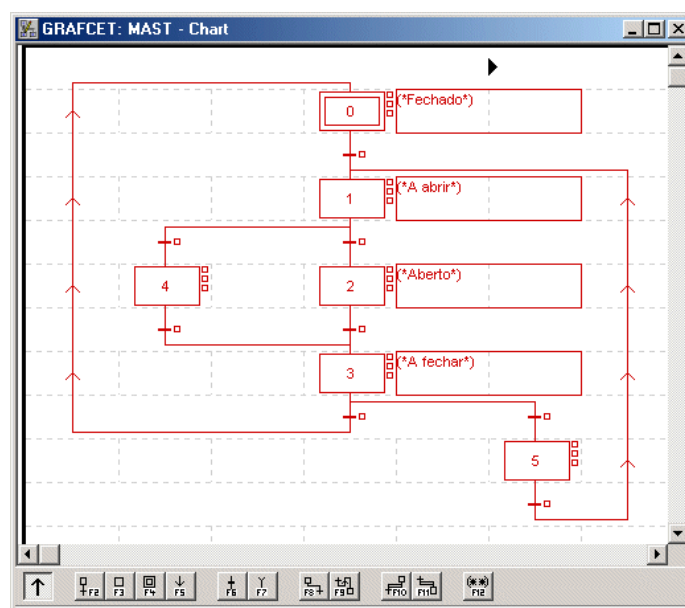


figura 24 – diagrama do exemplo

Para efectuar a programação das acções e das condições de transição o diagrama deve estar validado. Sempre que efectuar uma alteração no diagrama deve voltar a validá-lo.

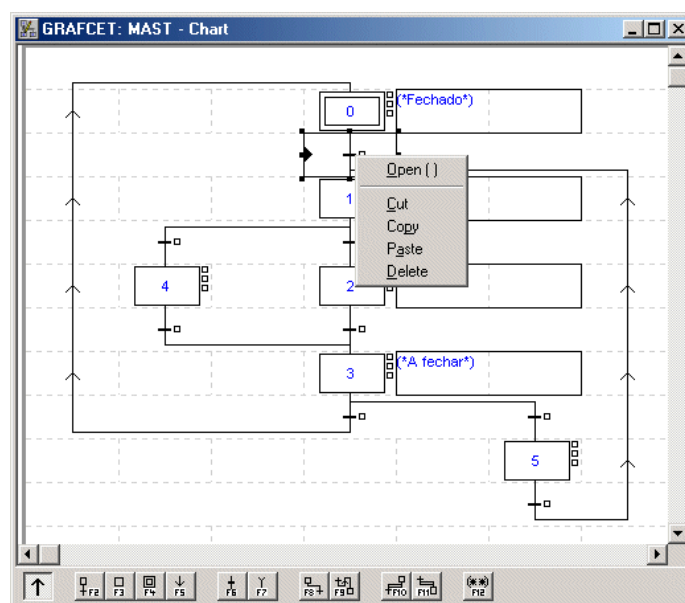


figura 25 – programação de condições de transição

A programação das transições é acessível seleccionando a transição com o botão do lado direito do rato. Abra a transição e seleccione a linguagem para a programação (LD, ST ou IL).

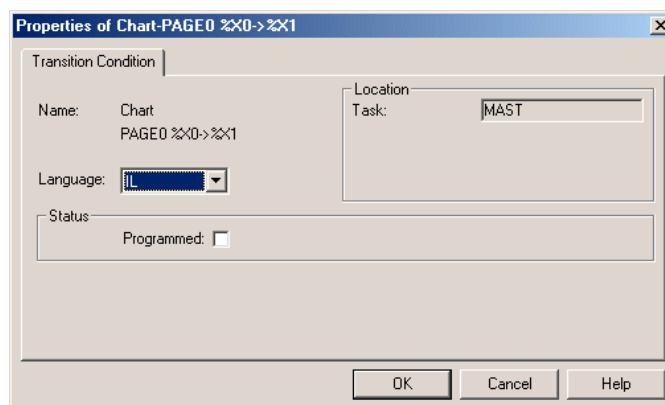



figura 26 – escolha da linguagem de programação

Neste caso vamos seleccionar a linguagem IL e programar a condição de transição para a transição entre as etapas **0** e **1** (figura 27). Após programar a condição deve validá-la através do botão . As condições para as restantes transições são programadas de modo semelhante.

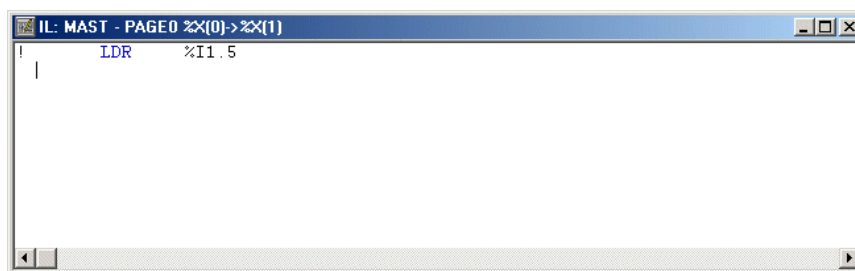


figura 27 – definição da condição de transição 0/1

A tabela seguinte apresenta as condições de transição para as restantes transições no GR7.

0/1	1/2	1/4	2/3	4/3	3/0	3/5	5/1
! LDR %I1.5	! LDR %I1.1	! LDR %I1.5	! LDR %I1.5	! LDR %I1.5	! LDR %I1.2	! LDR %I1.5	! LDR %I1.5

tabela 4 – condições de transição

A programação das acções é efectua seleccionando a etapa pretendida com o botão direito do rato (ou com o botão esquerdo e de seguida premindo Shift+F10). Comece por seleccionar o tipo de acção associado com a etapa (acção na activação, acção contínua ou acção na desactivação) e de seguida a linguagem em que pretende programar a acção (LD, ST ou IL).

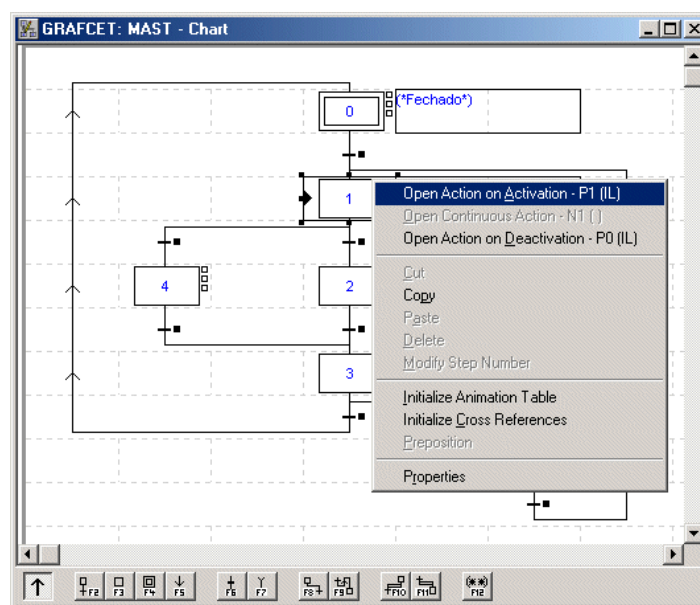



figura 28 – programação das acções

Para este caso vão ser utilizadas acções na activação e acções na desactivação, isto é, que são executadas enquanto a etapa a que estão associadas é activada e quando é desactivada, programadas em IL. A figura seguinte apresenta a programação da acção na activação associada à etapa **1**. Após programar a acção deve validá-la através do botão . As restantes acções são programadas de modo semelhante.

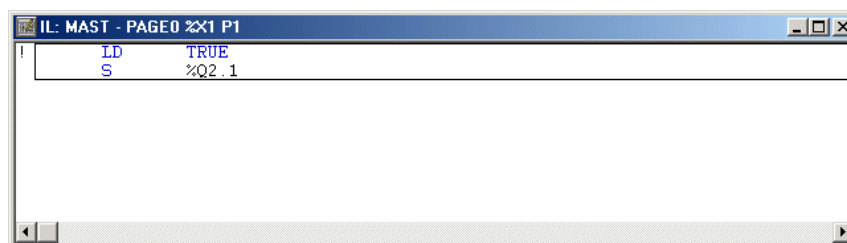


figura 29 – acção na activação da etapa 1

A tabela seguinte apresenta as acções a programar no GR7. Apenas é necessário associar acções às etapas **1** (a abrir) e **3** (a fechar).

1		3	
activação	desactivação	activação	desactivação
! LD TRUE S %Q2.1	! LD TRUE R %Q2.1	! LD TRUE S %Q2.2	! LD TRUE R %Q2.2

tabela 5 – acções

O aspecto final do GR7 deverá ser semelhante ao da figura seguinte. Os quadrados a cheio junto aos objectos (transições e etapas) indicam quais são os que têm as respectivas condições de transição (para as transições) ou acções (para as etapas) definidas.

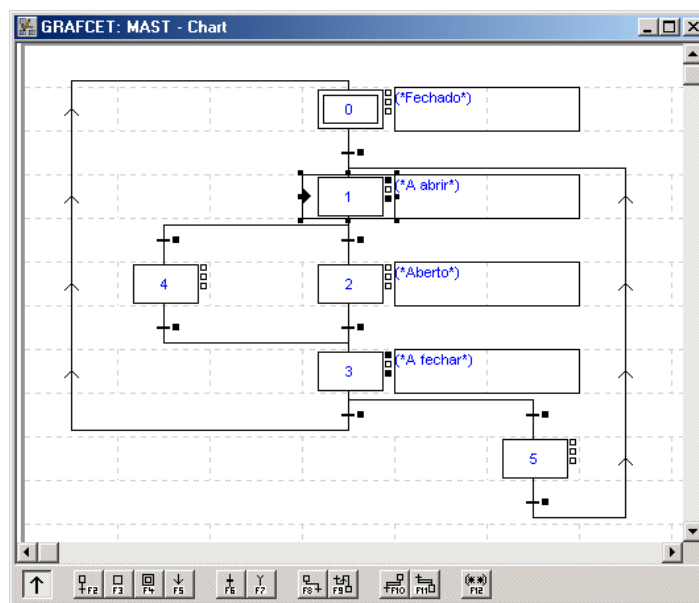


figura 30 – aspecto final do GR7

Teste e experimentação do sistema de controlo

Utilize o kit “Parque Automóvel” para simular o funcionamento do portão. Considere que o botão BE representa o sensor ABERTO e o botão BS representa o sensor FECHADO. Inclua um botão exterior para representar o botão COMANDO. Utilize o *led* SEMÁFORO VERDE para representar o portão a abrir (ABRIR) e o *led* SEMÁFORO VERMELHO para representar o portão a fechar (FECHAR).

Depois de realizar as ligações indicadas entre o PLC e o kit de simulação estabeleça a ligação entre o computador e o PLC – menu *PLC/ Connect* (ou CTRL + K). Seleccione a opção de transferir do PC para o PLC. Uma vez efectuada a transferência (indicação ON LINE na barra de estados) dê ordem de arranque ao autómato – menu *PLC/ Run* (ou CTRL + SHIFT + R).

Animação

Quando o autómato se encontra em modo **Run** é possível visualizar no editor de GR7 a evolução do estado do autómato. Uma etapa activa é indicada a preto enquanto que uma etapa não activa é indicada a branco. Para tal, a função de animação deve estar activa (quando o PLC está no modo **Run**).

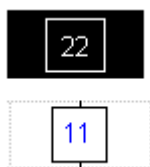


figura 31 – animação do GR7

Em simultâneo pode utilizar a tabela de animação que definiu inicialmente para observar a evolução das variáveis de entrada e saída.

Address	Symbol / Name	Current value	Kind	Type
%I1.1	Aberto			
%I1.2	Fechado			
%I1.5	Comando			
%Q2.1	Abre			
%Q2.2	Fecha			

figura 32 – tabela de animação

Tópicos avançados

Temporização de paragem

Considere agora um modo de funcionamento diferente. Ao contrário do que foi especificado inicialmente, quando o portão é parado antes de alcançar o extremo (ABERTO, quando o portão está a ABRIR, ou FECHADO, quando o portão está a FECHAR) e ficar nessa posição até que volte a ser pressionado o botão COMANDO (instante em que se começa a mover no sentido contrário), este deve permanecer imóvel por 30 segundos e, após esse período de tempo, deve começar a mover-se no sentido contrário ao que se movia inicialmente.

Para satisfazer este requisito de funcionamento deve utilizar um objecto específico do Grafset - %Xi.T - associado às etapas. Substitua as condições de transição das transições 4/3 e 5/0 pelas apresentadas na tabela seguinte.

4/3	5/0
! LD [%X4.T>30]	! LD [%X5.T>30]

tabela 6 – condições de transição para temporização de paragem

Célula fotoelétrica

Considere agora que o sistema, retomando o modo de funcionamento inicial, dispõe de um outro sensor. Existe uma célula fotoelétrica⁵ (CELULA), um sensor do tipo normalmente aberto, que detecta objectos atravessados no portão. Considere dois modos de funcionamento distintos.

Paragem

Considere que se, quando o portão está a fechar, a célula detecta um objecto atravessado no portão este pára imediatamente e mantém-se imóvel, sem responder ao botão COMANDO, enquanto se mantiver um objecto atravessado o portão. Quando o objecto deixa de ser detectado retoma o fecho do portão. Este requisito é implementado através de acções condicionais. Para isso acrescente uma acção contínua associado à etapa **3** (A fechar) incluindo um teste ao valor da célula. Também é necessário alterar as condições de transição das transições **3/0** e **3/5** pelas apresentadas na tabela seguinte.

3
! LD %X3 ANDN %I1.3 ST %Q2.2

3/0	3/5
! LDR %I1.2 ANDN %I1.3	! LDR %I1.5 ANDN %I1.3

tabela 8 – condições de transição para paragem condicionada

Abertura

Considere que se, quando o portão está a fechar, a célula detecta um objecto atravessado no portão este pára imediatamente e após 5 segundos move-se no sentido contrário até estar completamente aberto. Enquanto se mantiver um objecto atravessado o portão mantém-se imóvel sem responder ao botão COMANDO.

Este requisito é implementado recorrendo a macro acções. Para tal deve construir um novo diagrama que representa o funcionamento sequencial do esquema de segurança baseado na célula fotoelétrica. Os diagramas resultantes são apresentados na figura 33. As condições de transição para o novo GR7 são apresentadas na tabela 9.

⁵ Inclua uma nova variável de entrada – CELULA (%I1.3).

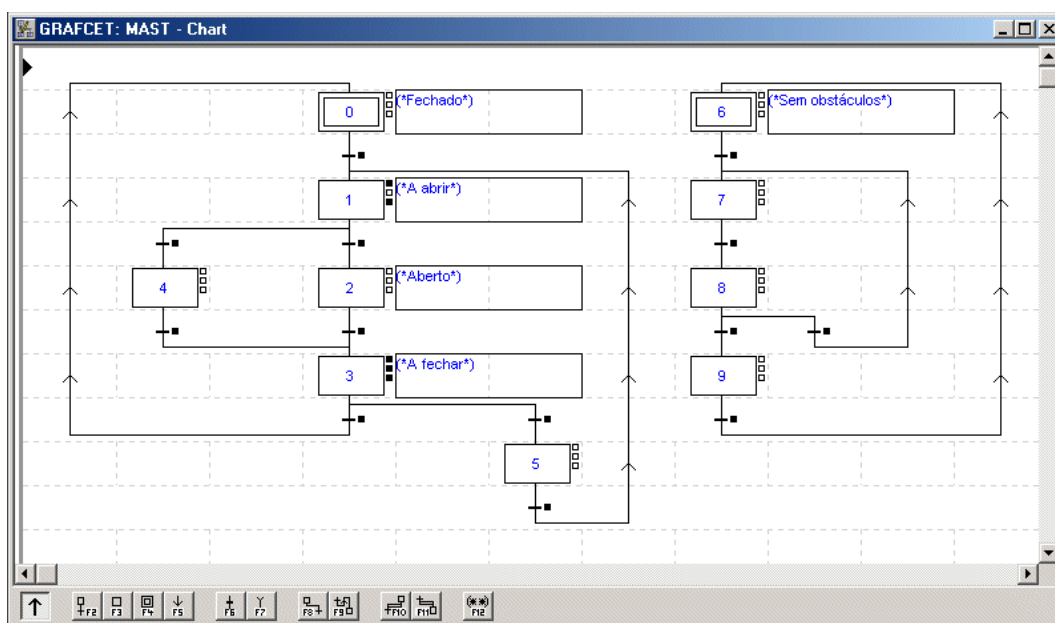


figura 33 – GR7 com macro acções

O GR7 que representa o funcionamento do esquema de segurança da célula define dois estados de funcionamento para o portão: o estado de funcionamento *normal* (sem obstáculos - etapa 6) e *bloqueio* (com obstáculos – etapas 7, 8 e 9). A transição do estado *normal* para *bloqueio* só é possível se o portão estiver a fechar.

6/7	7/8	8/9	8/7	9/6
! LDR %I1.3 AND %X3	! LDN %I1.3	! LD [%X8.T>50]	! LDR %I1.3	! LD TRUE

tabela 9 – condições de transição para o GR7 da CELULA

O pré processamento (programado em LD, ST ou IL) é utilizado para alterar o estado actual do GR7 (ou seja “forçar” determinados estados no GR7) quando ocorre uma alteração no modo de operação (por exemplo quando ocorre uma falha que provoca uma degradação do desempenho). As macro acções são programadas no pré processamento, pois é aqui que podem ser alterados os bits associados às etapas do GR7 (%Xi: instruções de Set e Reset⁶). Para esta situação a macro acção estaria localizada na etapa 9 que alteraria o GR7 do portão, desactivando a etapa 3 e activando a etapa 1. Esta macro acção é

3
! LD %X3 ANDN %X7 ANDN %X8 ANDN %X9

⁶ Quando se efectua o Reset de uma etapa, as acções associadas à sua desactivação não são executadas.

utilizada em conjunto com a acção condicionada que vimos no caso anterior que neste caso teria que ser alterada de modo ficar relacionada com o estado do sistema.

Para programar a macro acção abra o editor no módulo de pré processamento – (*Application Browser*) *Station/ Program/ MAST Task/ Prl* – e introduza a sequência da tabela seguinte.

Também é necessário voltar a alterar as condições de transição das transições **3/0** e **3/5** pelas apresentadas na tabela seguinte, de modo a relacionar o disparo da transição com o estado em que o sistema se encontra.

Pré Processamento (IL)	
! LD %X9	
R %X3	
S %X1	

3/0	3/5
! LDR %I1.2	! LDR %I1.5
ANDN %X7	ANDN %X7
ANDN %X8	ANDN %X8
ANDN %X9	ANDN %X9

tabela 12 – condições de transição para temporização de paragem

Sistema de iluminação

Considere agora um requisito adicional para o modo de funcionamento inicial. Existe um sistema de iluminação (LUZ) que também deve ser controlado pelo autómato. Pretende-se que, assim que o portão seja accionado (seja para abrir ou fechar) a luz se acenda e, após a imobilização do portão, se mantenha acesa por um período de 5 minutos. A iluminação da garagem pode também ser controlada por um interruptor (INTERRUPTOR) de pressão, do tipo normalmente aberto. Quando o INTERRUPTOR é accionado, e caso esteja apagada, a iluminação deve acender imediatamente e manter-se acesa por 5 minutos. Se entretanto o portão for accionado o sistema de iluminação passa ao primeiro modo de funcionamento (apaga passados 5 minutos da imobilização do portão).

Sistema de iluminação com sensor de luminosidade

Uma vez que a garagem tem iluminação natural é desnecessário que o sistema de iluminação funcione durante o dia. Para isso foi colocado um sensor luminosidade (LUMINOSIDADE). Considere que o sensor tem toda a lógica associada que permite ter à entrada do autómato um sinal digital (**0** / **1**). Assim um **0** significa que existe luz natural suficiente (isto é, não é necessário accionar o sistema de iluminação) e um **1** significa luz natural insuficiente.

Pretende-se alterar o sistema de controlo, incorporando este sensor, de modo a garantir que sempre que exista luz natural suficiente o sistema de iluminação não é accionado.

Segurança anti-stress

Por razões de segurança do material, pretende-se introduzir no sistema de controlo um esquema de segurança anti-stress. Este esquema pretende evitar possíveis danos ao sistema causados por utilizadores que accionam o botão COMANDO quatro ou mais vezes seguidas (ou quando é accionado por vários utilizadores em simultâneo). Pretende-se que o sistema pare de responder, por um período de 30 segundos, a partir do momento que o botão COMANDO é accionado mais de três vezes num espaço de 20 segundos. Ao fim dos 30 segundos de paragem deve regressar ao modo de funcionamento em que se encontrava.