Parte 06 - Técnicas de programação (máquina de estados)

MÁQUINA DE ESTADOS FINITO

Pode-se definir máquina de estado como sendo um modelo de comportamento de um determinado processo, em nosso caso industrial. Uma máquina de estado é composta por estados, transições e saídas.

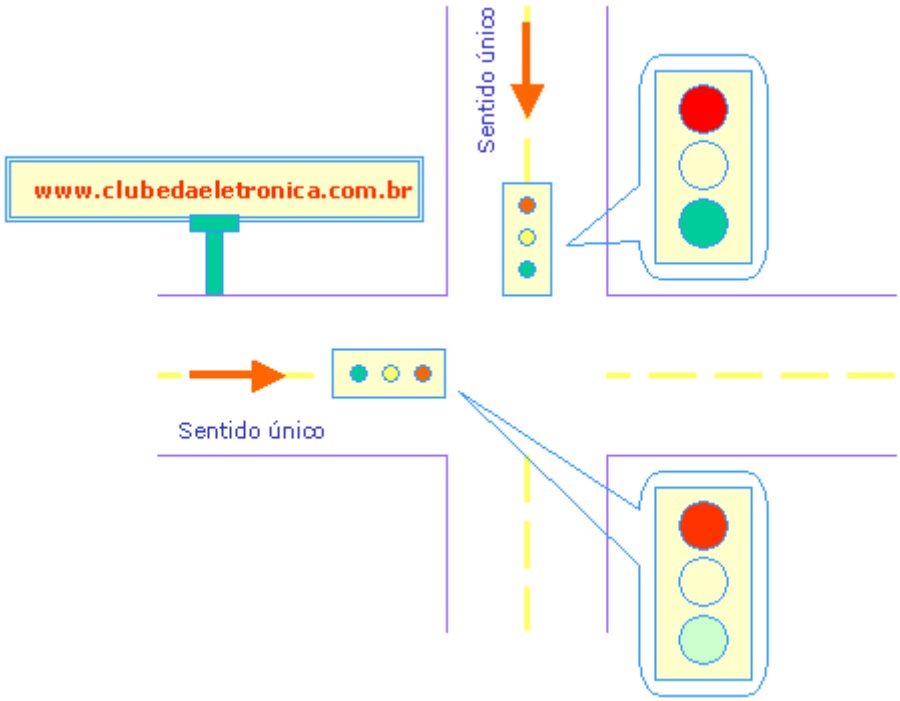
- **Estado** → comporta-se como uma memória, ou seja, armazena todas as informações sobre as saídas em um determinado momento.
- **Transição** → é a condição para que ocorra a mudança de um estado para outro.
- **Saída** → descreve a atividade que deve ser realizada num determinado estado.

A máquina de estado é representada por um diagrama bastante simplificado, conhecido como diagrama de transição de estado, que tem como objetivo facilitar o entendimento de qualquer pessoa interessada no processo.

O objetivo deste material é demonstrar com exemplos como a máquina de estado reproduz fielmente todas as etapas idealizadas pelo projetista.

Sistema seqüencial simples – controle de tráfego (Resolvido)

Para que um programa faça o que você quer, você deve dizer o que ele deve fazer, sem omitir qualquer passo. Para explicar a máquina de estado primeiramente faremos um sistema lógico simples e conhecido por todos, trata-se de um controle de trafego. A figura abaixo ilustra a idéia proposta.



Descrição de funcionamento

O semáforo terá início quando um botão (BL) tipo “push button” for acionado, dar-se-à então início ao ciclo. O tempo (T) ficará a critério do operador e o sistema poderá ser desligado em qualquer momento através de um botão (BD), também push button.

Definição de entradas e saídas (I/O)

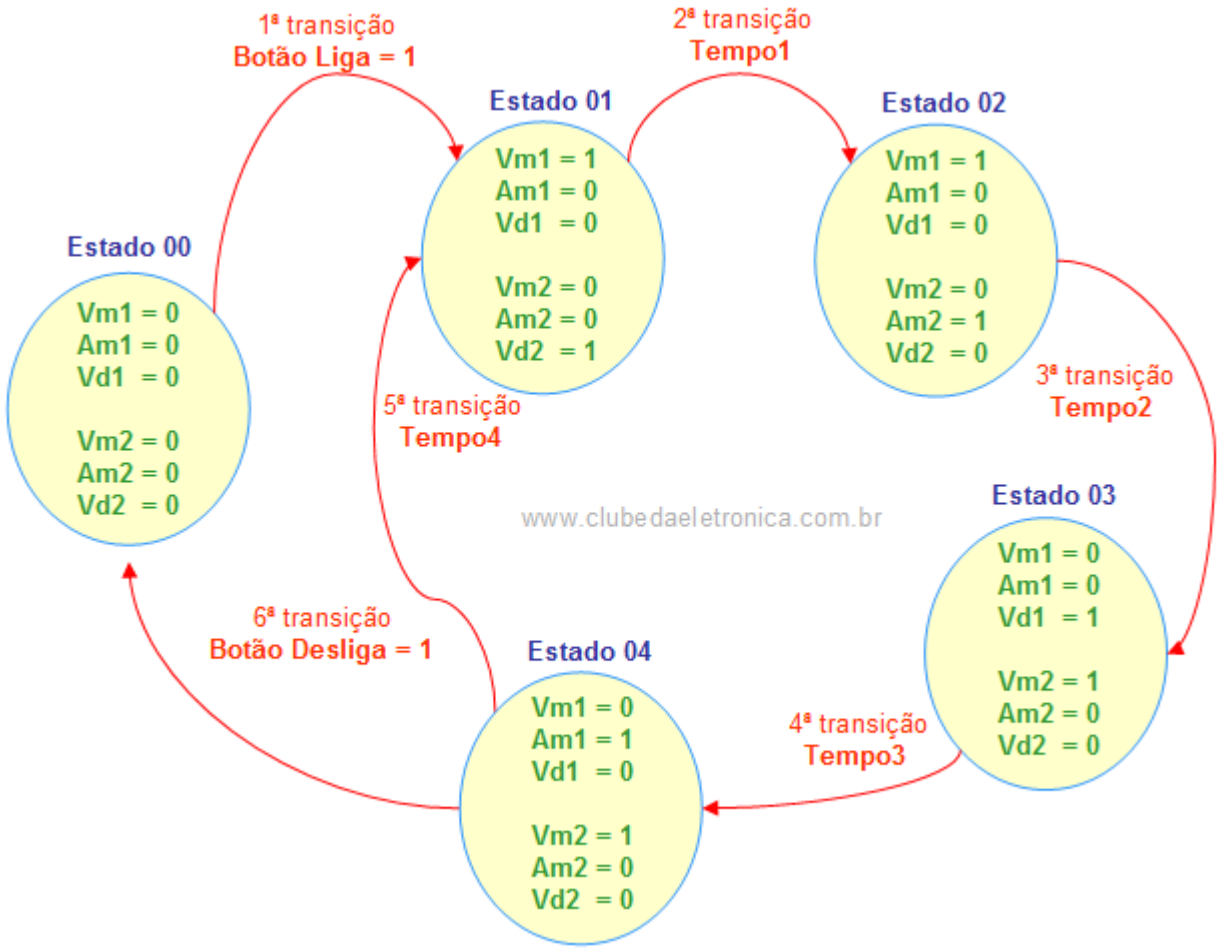
Entradas e saídas (E/S) ou input/output (I/O) são termos comuns na automação, isso porque, em todo sistema há necessidade da inserção de informações (I), que devem se processadas através de uma unidade central de processamento (CPU) e enviadas às saídas (O), para que uma determinada ação, previamente programada, seja tomada.

Lista de entradas e saídas (I/O) para o semáforo

Entradas	Saídas Semáforo 1	Saídas Semáforo 2
Botão liga = I0	Vermelho 1 = O0	Vermelho 2 = O3
Botão desliga = I1	Amarelo 1 = O1	Amarelo 2 = O4
	Verde 1 = O2	Verde 2 = O5

Elaboração da máquina de estado

O objetivo da máquina de estados é modelar através do diagrama de estado o comportamento do projeto, no caso, o controle de trafego. Sua representação é feita com detalhes a fim de facilitar o entendimento. Abaixo, sua representação já aplicada ao semáforo.



Explicação detalhada

Estado 00 ⇒ Representa o estado inicial da máquina, ou seja, todas as saídas estão desligadas.

1ª Transição ⇒ Para que o sistema mude de um estado para outro, é necessário que haja a incersão de informações, em nosso caso, estado 01 só será ligado se BL for acionado.

Estado 01 ⇒ Representa o estado após o recebimento da informação, neste caso, as lâmpadas vermelho 01 e Verde 02 estarão em nível alto, ou seja, acesas enquanto que as outras estarão apagadas.

2ª Transição ⇒ O estado 01 permanecerá até que termine o tempo previamente estabelecido pelo projetista, neste caso, o tempo representa uma transição e por conseqüência uma entrada.

Estado 02 ⇒ Ao terminar o tempo estabelecido na 2ª transição, haverá mudança do estado 01 para o estado 02 e assim, suas saídas serão acionadas, neste caso, as lâmpadas vermelho 01 e amarelo 02 estarão em nível alto, ou seja, acesas enquanto que as outras estarão apagadas.

3ª Transição ⇒ Mais um tempo é necessário para que ocorra a mudança de estado, ou seja, só mudará do estado 02 para o estado 03 no final do se o podendo , O estado 01 permanecerá até que termine o tempo estabelecido no programa.

Estado 03 ⇒ Ao final da 2º transição, o estado 03 é acionado, e com ele as lâmpadas verde 01 e vermelho 02 estarão em nível alto, ou seja, acesas enquanto que as outras estarão apagadas.

4ª Transição ⇒ Mais um tempo é necessário para que ocorra a mudança de estado, ou seja, só mudará do estado 03 para o estado 04 no final do tempo estabelecido no programa.

Estado 04 ⇒ Ao final da 3º transição, o estado 04 é acionado, e com ele as lâmpadas amarelo 01 e vermelho 02 estarão em nível alto, ou seja, acesas enquanto que as outras estarão apagadas.

5ª Transição ⇒ Um último tempo é necessário para que o processo reinicie, ou seja, volta ao estado 01 e o ciclo recomeça.

6ª Transição ⇒ A 6ª transição tem função de parar o sistema. Note que todos os estados serão desligados simultaneamente.

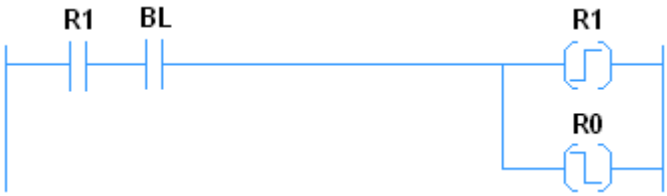
O Programa ladder

O programa ladder deve ser fiel a máquina de estados e se esta última for perfeita o programa também será.

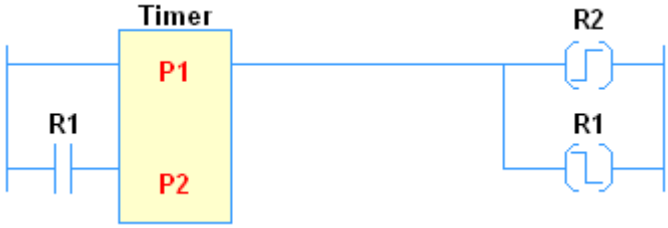
Detalhes da lógica Ladder

Lógica 01 ⇒ Cada estado de seu sistema deve ser associado a um contato auxiliar **(R)**. Como na máquina há quatros estados, temos assim, quatro contatos auxiliares, ligando o estado inicial (tudo desligado).

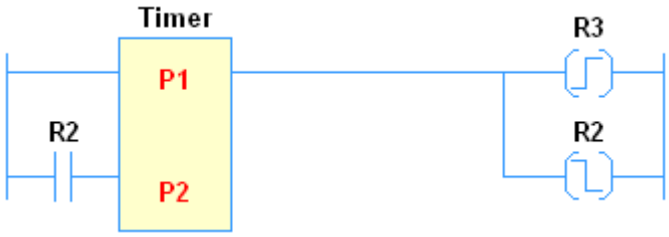
Lógica 02 ⇒ Se o estado inicial **(R0)** estiver acionado e se o Botão (push button) liga **(BL)** for pressionado é setado o estado **(R1)** e resetado o estado **(R0)**.



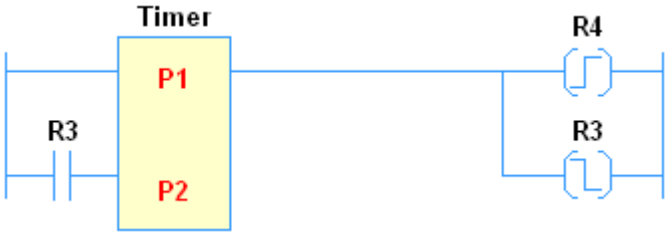
Lógica 03 ⇒ Se o estado **(R1)** estiver setado dar-se-á inicio a contagem de tempo **(T1)** e ao final o estado **(R2)** será setado e **(R1)** ressetado.



Lógica 04⇒ Se o estado **(R2)** estiver setado dar-se-á inicio a contagem de tempo **(T2)** e ao final o estado **(R3)** será setado e **(R2)** ressetado.

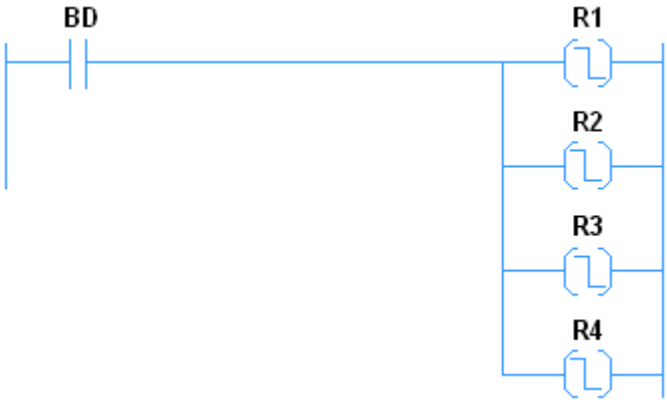


Lógica 05 ⇒ Se o estado **(R3)** estiver setado dar-se-á inicio a contagem de tempo **(T3)** e ao final o estado **(R4)** será setado e **(R3)** ressetado.

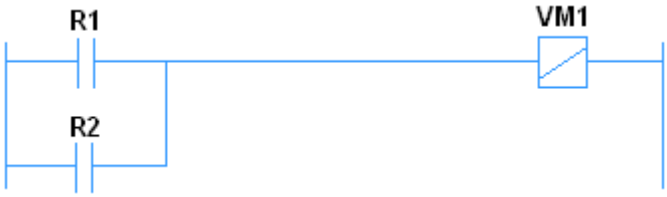


Lógica 06 ⇒ Se o estado **(R4)** estiver setado dar-se-á inicio a contagem de tempo **(T4)** e ao final o estado **(R1)** será setado e **(R4)** ressetado. O ciclo recomeça.

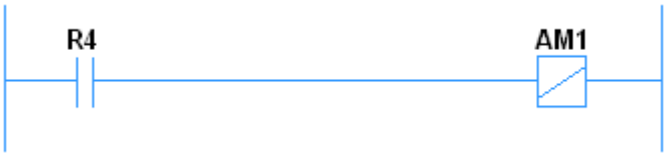
Lógica 07 ⇒ Se o botão (push button) for pressionado todos os estados **(R1)**, **(R2)**, **(R3)** e **(R4)** serão ressetados.



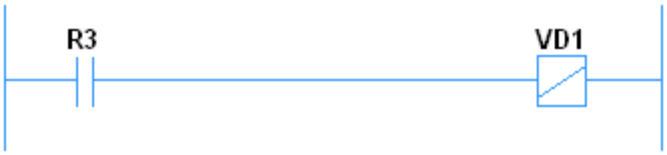
Lógica 08 ⇒ Acionando a saída vermelha do semáforo 01 **(VM1)**. Observando a máquina de estado note que a saída vermelho 01 está em nível lógico alto nos estados **(R1)** e **(R2)**. Assim, os estados **(R1)** ou **(R2)** devem acionar a saída **(VM1)**.



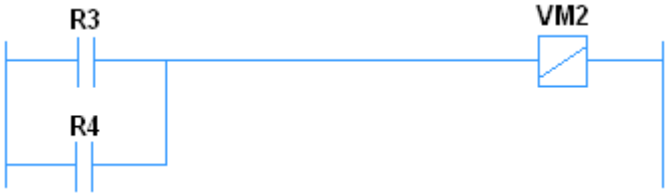
Lógica 09 ⇒ Acionando a saída amarela do semáforo 01 **(AM1)**. Observando a máquina de estado note que a saída amarela 01 está em nível lógico alto somente no estado **(R4)**. Assim, o estado R4 deve acionar a saída **(AM1)**.



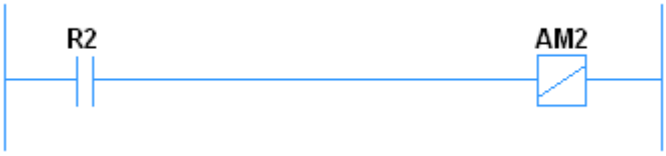
Lógica 10 ⇒ Acionando a saída verde do semáforo 01 **(VD1)**. Observando a máquina de estado note que a saída verde 01 está em nível lógico alto somente no estado **(R3)**. Assim, o estado R3 deve acionar a saída **(VD1)**.



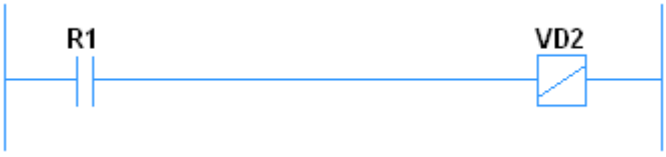
Lógica 11 ⇒ Acionando a saída vermelha do semáforo 02 **(VM2)**. Observando a máquina de estado note que a saída vermelha 02 está em nível lógico alto nos estados **(R3)** e **(R4)**. Assim, o estado os estados **(R3)** ou **(R4)** devem acionar a saída **(VM2)**.



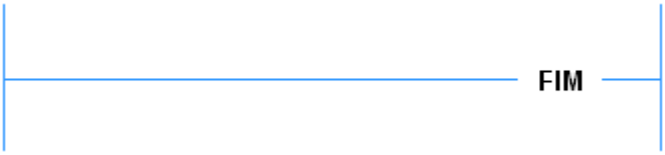
Lógica 12 ⇒ Acionando a saída amarela 02 do semáforo 02 **(AM2)**. Observando a máquina de estado note que a saída amarela 02 está em nível lógico alto somente no estado **(R2)**. Assim, o estado o estado **(R2)** deve acionar a saída **(AM2)**.



Lógica 13 ⇒ Acionando a saída verde 02 do semáforo 02 **(VD2)**. Observando a máquina de estado note que a saída amarela 02 está em nível lógico alto somente no estado **(R1)**. Assim, o estado o estado **(R1)** deve acionar a saída **(VD2)**.

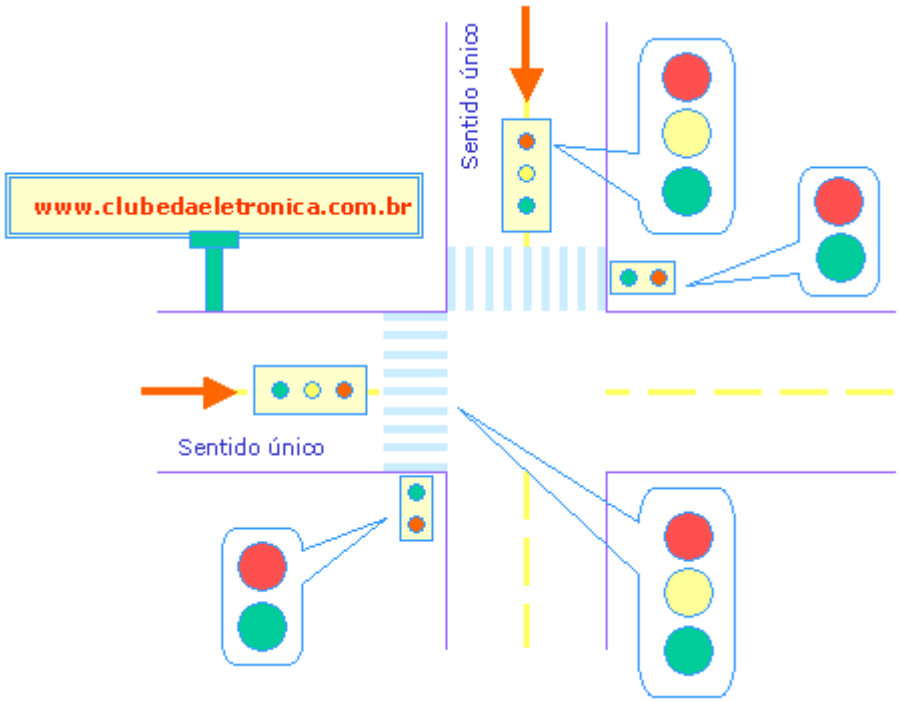


Lógica 14 ⇒ Fim de programa.



Praticando...

Deseja-se implementar um sistema de controle para semáforos para veículos e pedestres, como mostrado no esquema:



Descrição de funcionamento:

O circuito é iniciado quando o **botão de Liga (BL)** tipo “push button” é acionado, dar-se-á então, o início ao ciclo. O tempo fica a critério do projetista tomando cuidado para que não ocorram choques ou atropelamentos. Um **botão de desliga (BD)** tipo “push button” também deve ser implementado.

Etapas para relatório do projeto

- Descrição funcional (apontar possíveis falhas no processo e soluciona-las);
- Identificar as variáveis e criar a lista de entradas e saídas;
- Fazer o diagrama de estado da solução proposta;
- Implementar no Kit do Zap500 (não esquecer dos comentários); e

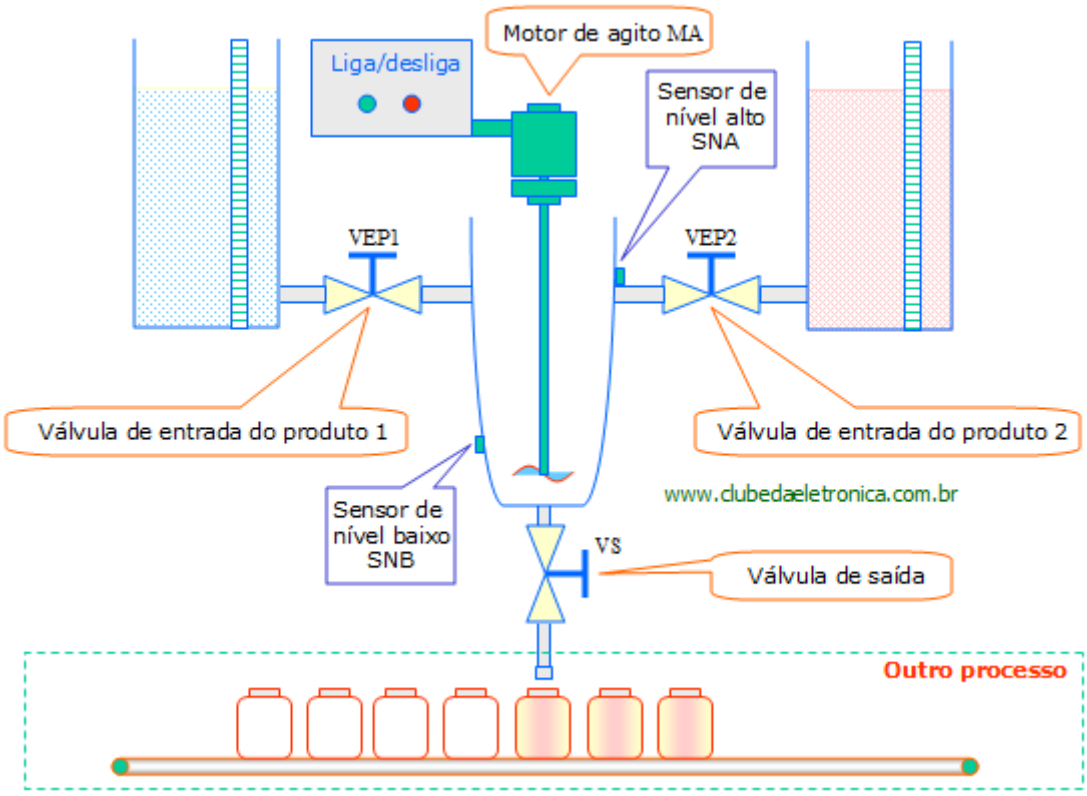
Mensagens na IHM do CLP

Hoje, praticamente todos os fabricantes de CLP (Controladores lógicos programáveis) disponibilizam a IHM (Interface Homem Máquina) como acessório opcional que, diga-se de passagem, agrega bastante valor ao CLP.

Normalmente a IHM consiste de um teclado para entrada de dados e uma tela (display) para visualização das informações pertinentes ao sistema. Abaixo, um exemplo resolvido ilustra o uso das mensagens na IHM.

Agitador de produtos (Resolvido)

Deseja-se implementar o sistema de controle do esquema abaixo:



Descrição de funcionamento

Ao pressionar o botão liga (**BL**), dar-se-á início ao processo abrindo simultaneamente as válvulas de entrada **VEP1** e **VEP2**; quando o sensor de nível alto **(SNA)** for atingido, automaticamente as válvulas de entrada se fecham e o motor agitador funciona por 10 segundos; A válvula de saída **(VS)** será aberta e o

líquido escoará até que o sensor de nível baixo **(SNB)** seja atingido, então o ciclo recomeça. Pressionando o botão desliga (**BD**) o processo será interrompido.

Etapas para elaboração do projeto

- Definir o problema corrigindo possíveis falhas no processo.
- Identificar as variáveis e criar a lista de entradas e saídas.
- Montar o diagrama de estado da solução proposta.
- Implementar no Kit do Zap500 (não esquecer dos comentários e identificação das variáveis)

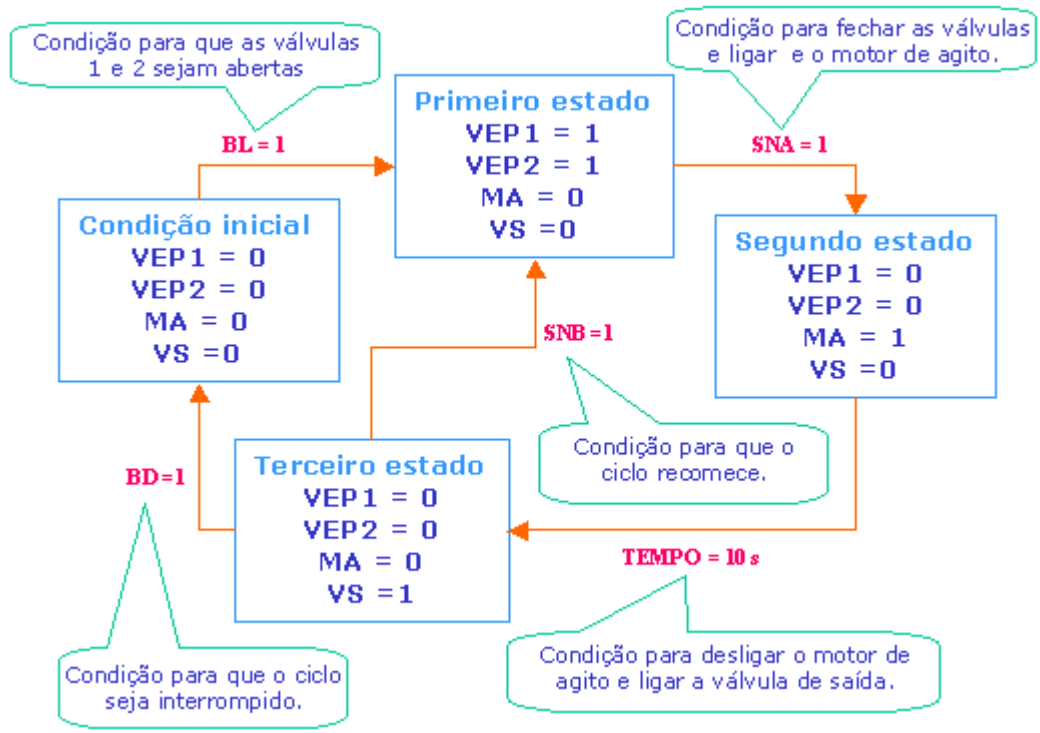
Solução:

Definição de entradas e saídas (I/O)

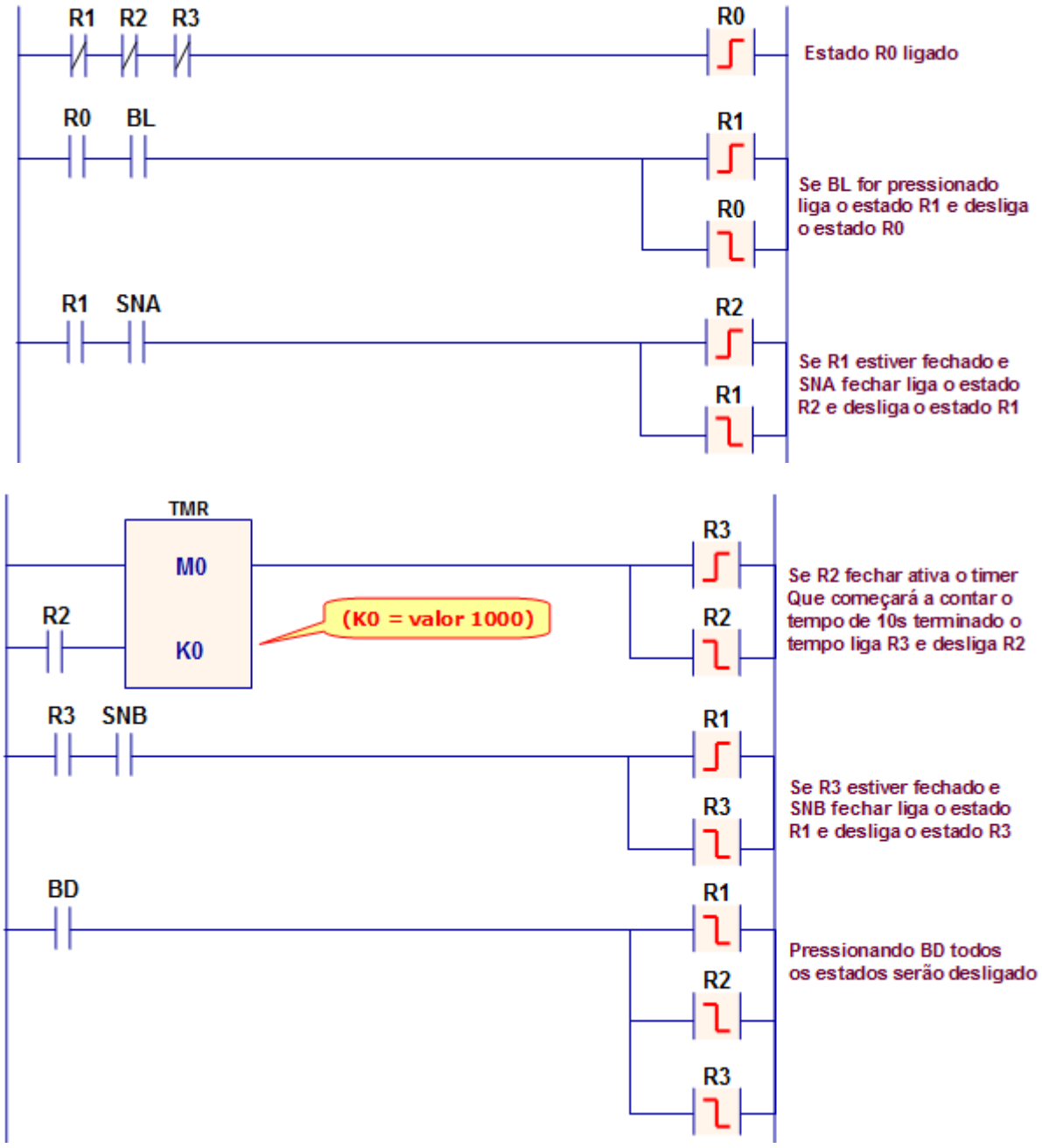
As entradas são as condições (transições) impostas pelo projetista para ocorra uma mudança de estado, e as saídas são os atuadores, ou seja, os que executarão algo se uma determinada condição for atingida. No projeto a ser implementado as entradas e as saídas, são:

Entradas	Saídas
✓ **BL** = Botão de liga	✓ **VEP1** = Válvula de entrada do produto 1
✓ **BD** = Botão de desliga	✓ **VEP2** = Válvula de entrada do produto 2
✓ **SNA** = Sensor de nível alto	✓ **MA** = Motor agitador
✓ **SNB** = Sensor de nível baixo	✓ **VS** = Válvula de saída.

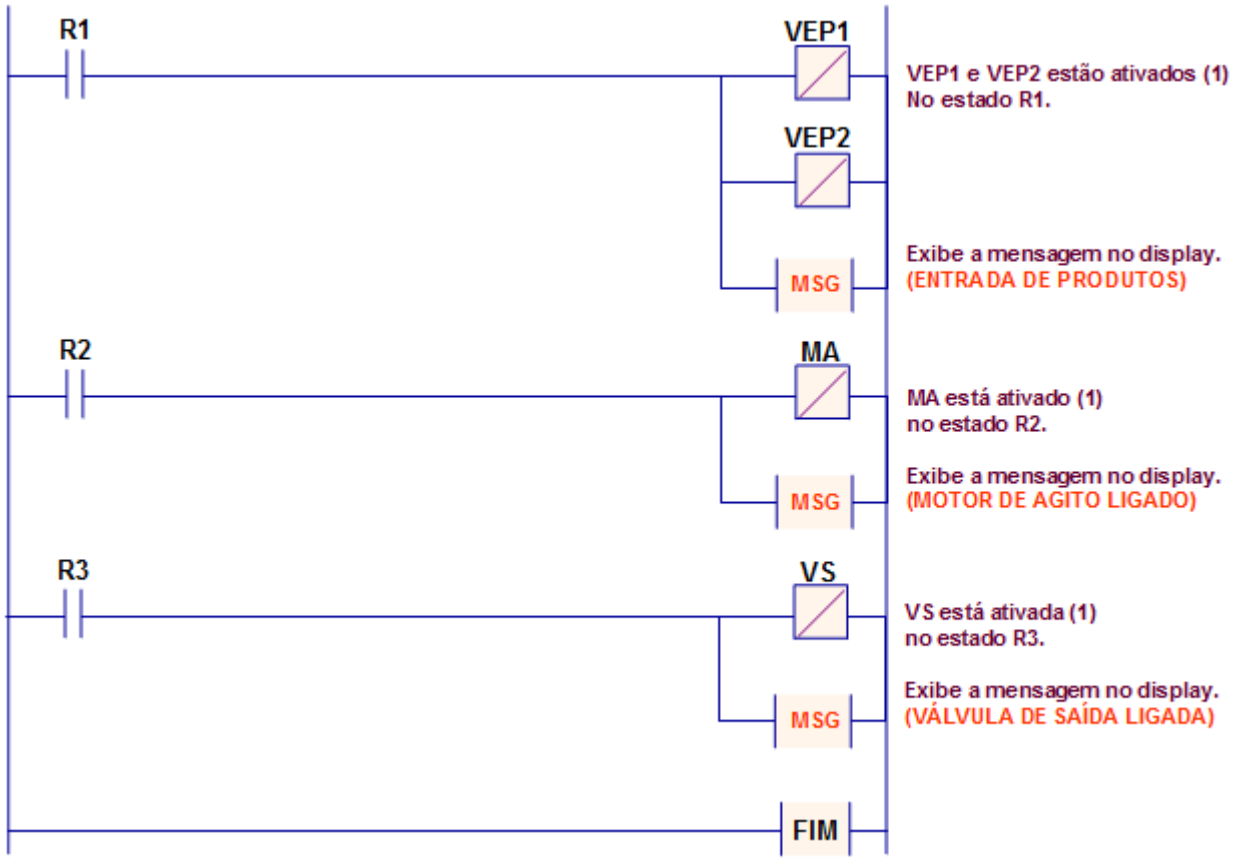
Máquina de estados



O programa ladder com comentários

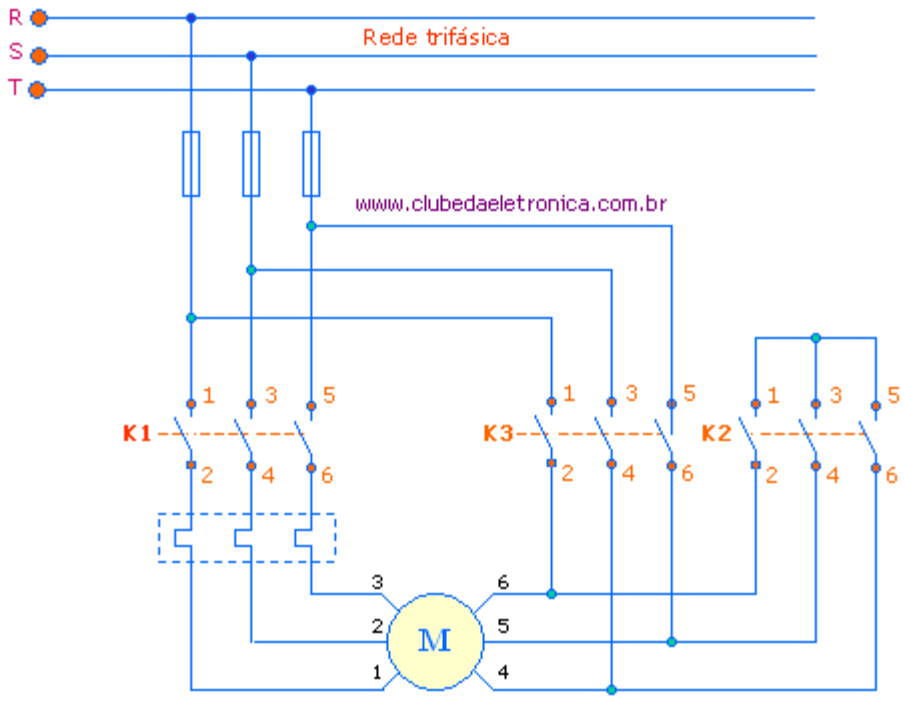


Energizando as saídas.

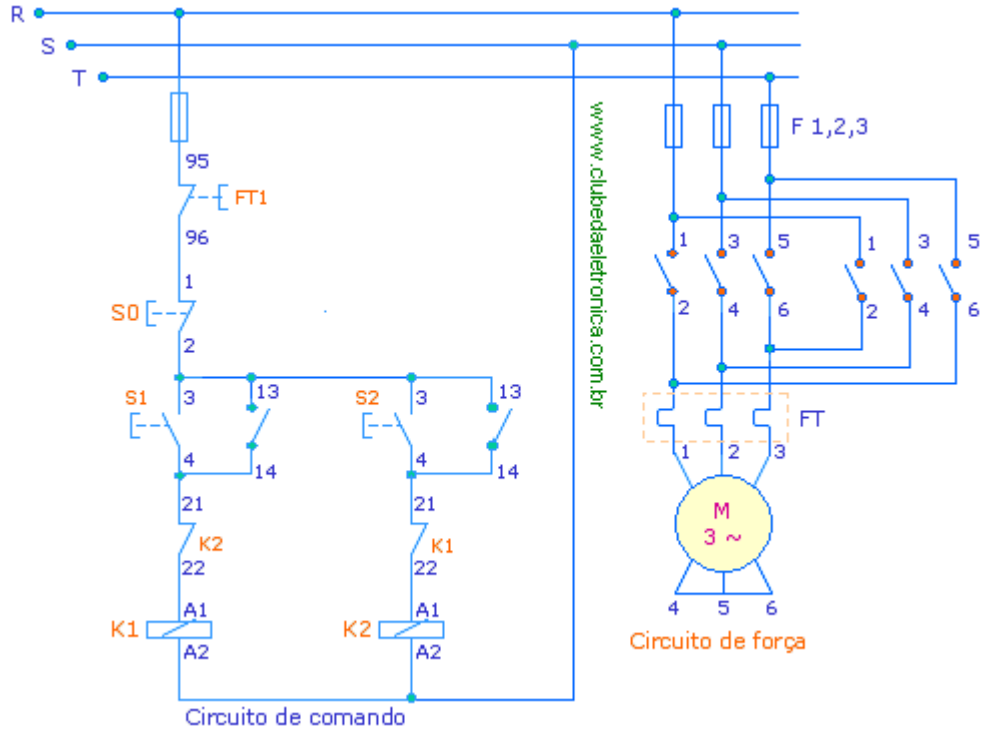


Praticando...

Elabore em ladder um programa capaz de partir o motor em Y (exibir mensagem TENSÃO 127V) e após 3 segundos comutá-lo automaticamente para Δ (exibir mensagem TENSÃO 220V). O diagrama de força esta representado abaixo.



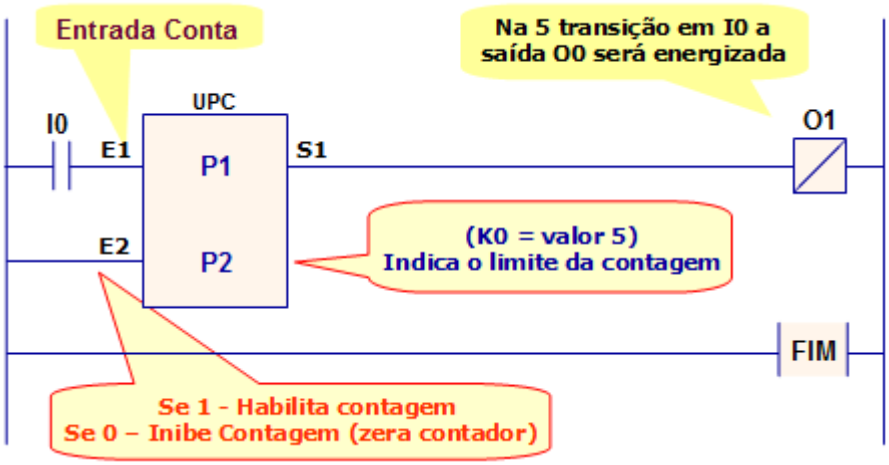
Elabore em ladder uma partida direta reversa, exibindo as seguintes mensagens (SENTIDO HORÁRIO e SENTIDO ANTI-HORÁRIO). O diagrama de comando e de força está representado abaixo.



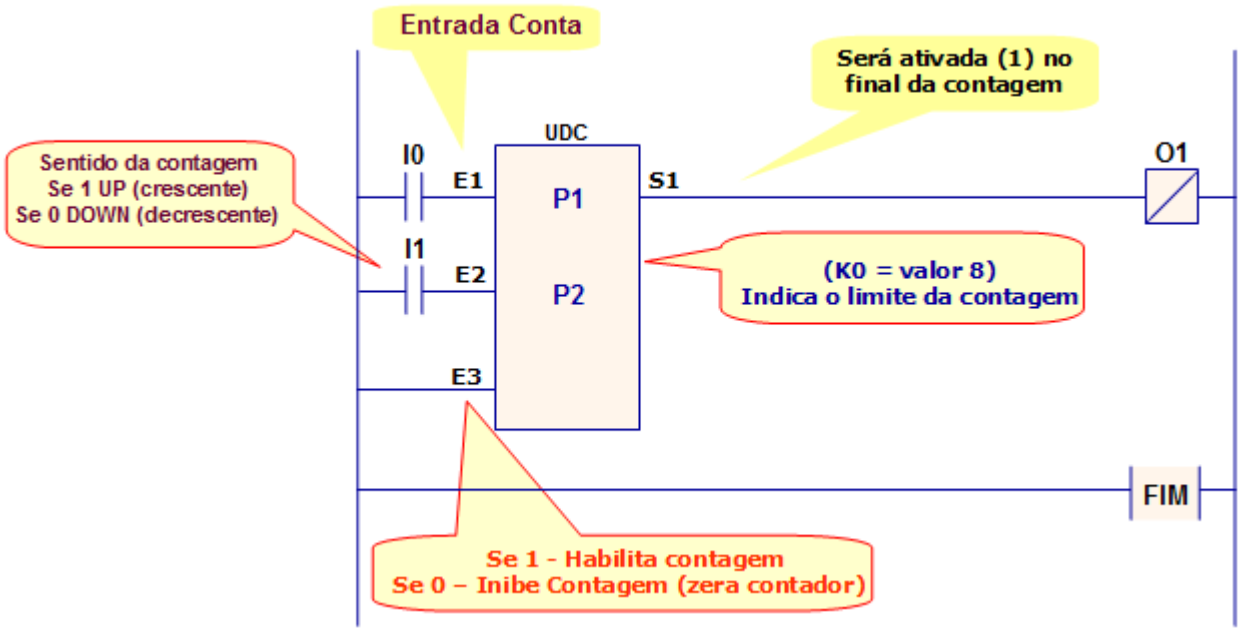
Blocos contadores UP (crescente) e UP/DOWN (crescente e decrescente)

São blocos destinados à contar um determinado número de transições ocorridas na entrada “conta”. Ele conta o número de transições até um valor fornecido pelo usuário como parâmetro. A saída indica o fim da contagem. Os contadores podem ser UP e UP/DOWN, vejamos:

Contador UP (crescente)



Contador UP (crescente)/DOWN(decrescente)



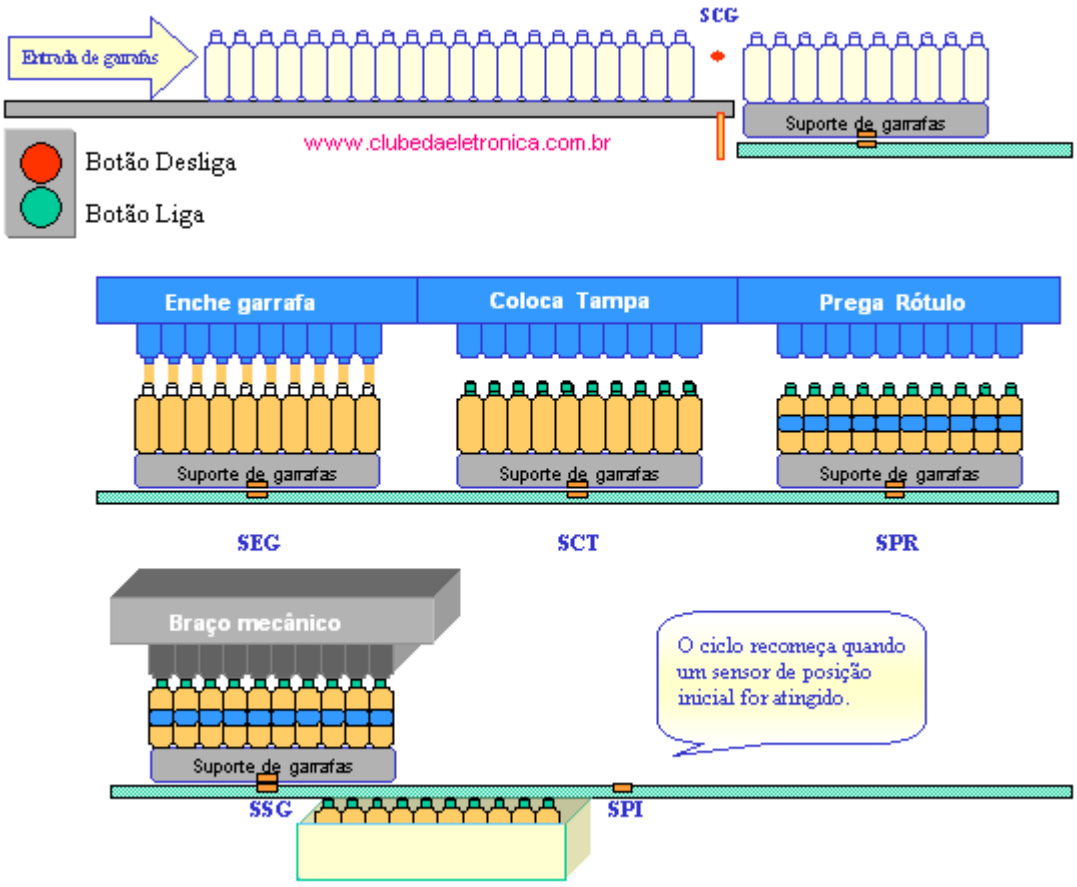
Os parâmetros P1 e P2 são iguais para ambos os contadores sendo:

P1 – representa o valor corrente da contagem e deve ser obrigatoriamente uma memória inteira (M).

P2 – representa o valor limite da contagem e pode ser memória inteira (M) ou constante inteira (K).

Praticando...

O projeto abaixo tem o objetivo didático, onde o aluno deverá entender o sistema descrito e automatizá-lo.



Descrição de funcionamento

- Pressionando o botão liga **(BL)** as garrafas começarão a entrar (processo não descrito) por uma esteira secundária **(EG) .**
- Haverá um controle de garrafas que será colocada sobre o suporte, em cada suporte haverá 10 garrafas, que serão contadas por intermédio de um sensor **(SCG)**.
- Assim que as dez garrafas estiverem no suporte, a esteira se movimentará através de um motor **(ME)**;
- O motor da esteira irá parar assim que o suporte atingir o sensor de enchimento **(SEG)**;
- Atingido o **SEG**, deverá acionar o motor de controle de descida da máquina que encherá as garrafas **(MEG)** que deverá ficar baixo por 5 segundos (tempo para enchimento da garrafa) e retornará a sua posição de origem aguardando próximo suporte.
- Assim o motor da esteira ligará novamente até chegar no sensor de tampa **(SCT)** que deverá acionar o motor de controle de descida da segunda máquina que será responsável por colocar a tampa esse processo levará 2 segundos e retornará a sua posição de origem aguardando próximo suporte.
- Novamente a esteira será ligada e chegará o sensor de prega de rótulo **(SPR)** que deverá acionar o motor de controle de descida da segunda máquina que será responsável pela colocação do rotulo esse processo levará 3 segundos e retornará a sua posição de origem aguardando próximo suporte.
- Assim que o rótulo for colocado a esteira se movimentará novamente até encontrar um sensor de saída de garrafas **(SSG)**, onde um braço mecânico irá retirar as garrafas do suporte, para coloca-las em uma caixa. O tempo de retirada será de aproximadamente 3s.
- A esteira liga novamente levando o suporte para a origem **(SPI)** que aguardará outras garrafas.

Importante:

- O sistema deverá ser ligado por intermédio de um botão de liga **(BL)** e desligado através de um botão desliga **(BD)**. **(Usar IHM)**
- Assim que o sistema for ligado as garrafas serão inseridas em um suporte, dando inicio a contagem.
- O processo acima é passivo pode ser melhorado, aceitando sugestões do projetista.
- Verifique se há erros no processo e aponte soluções.

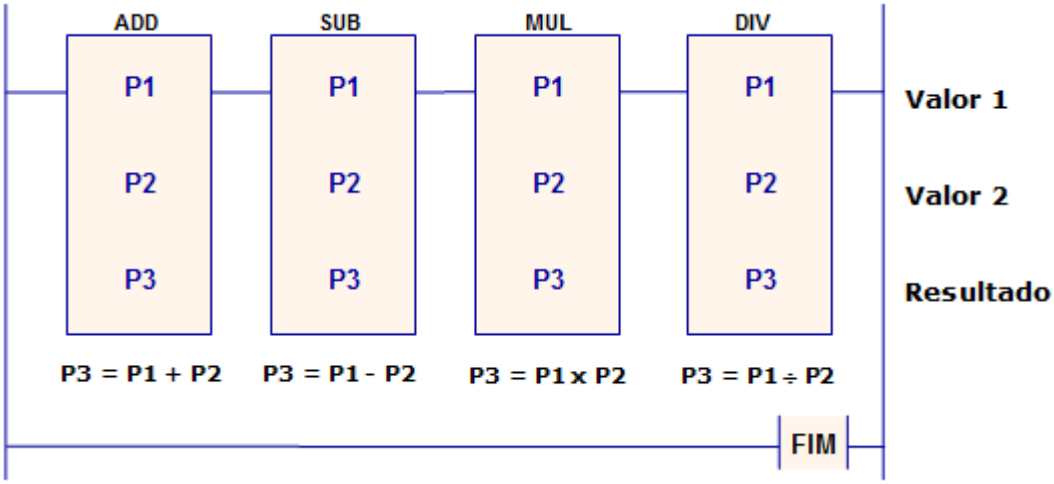
Etapas à seguir:

a) Identificar as variáveis e criar a lista de E/S.
b) Fazer os diagramas de estado, fluxograma, etc. da solução proposta.
c) Implementar no Kit do Zap500 (utilizar IHM para monitorar saídas)

Blocos matemáticos

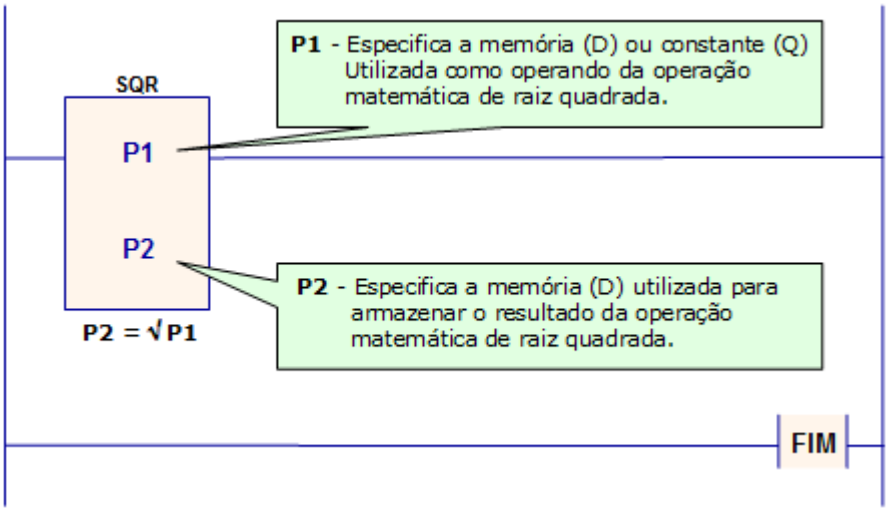
Como o próprio nome diz estes blocos realizam operações matemáticas como Adição (**ADD**), Subtração **(SUB)**, Multiplicação **(MUL)** e Divisão **(DIV)**.

As operações são realizadas entre os operandos P1 e P2 que podem ser Memória inteira **(M)**, Memória real **(D)**, Constante inteira **(K)**, ou constante real **(Q)** armazenando o resultado em P3 que pode ser uma memória Inteira **(M)** ou real **(D)**.

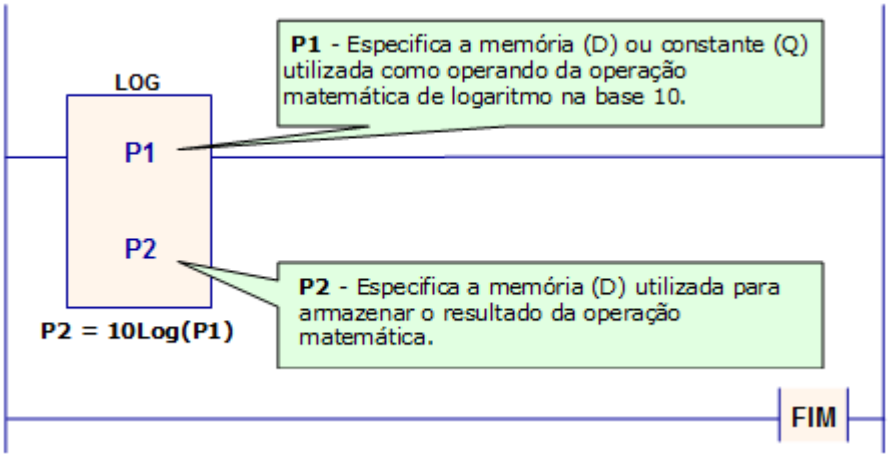


Além de realizar as operações básicas o SPDS-W também disponibiliza outros blocos que são:

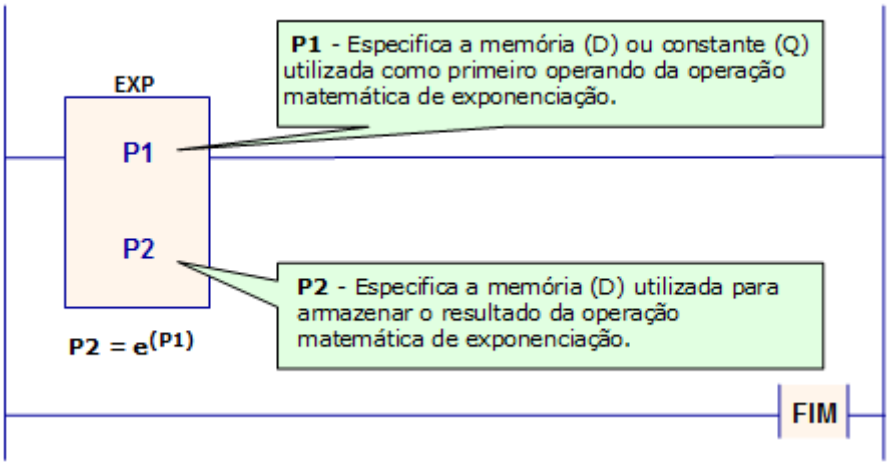
Extrator de raiz quadrada



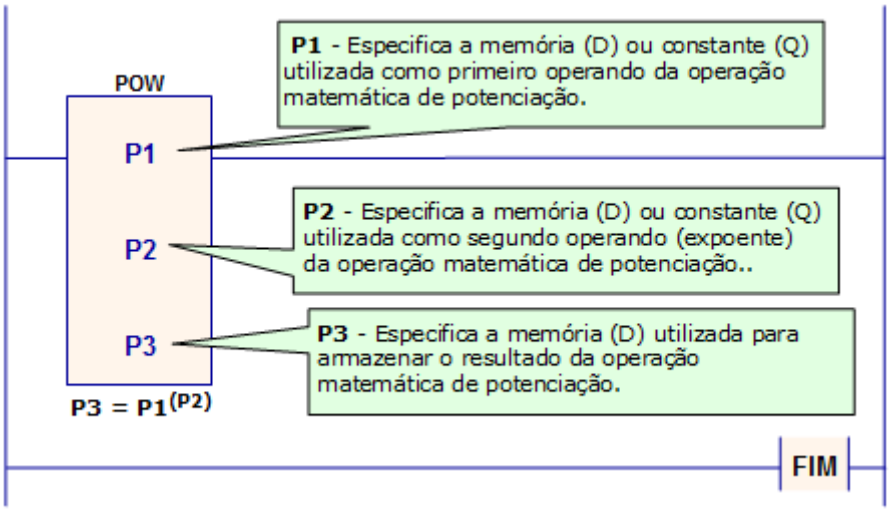
Log na base de 10



Exponenciação



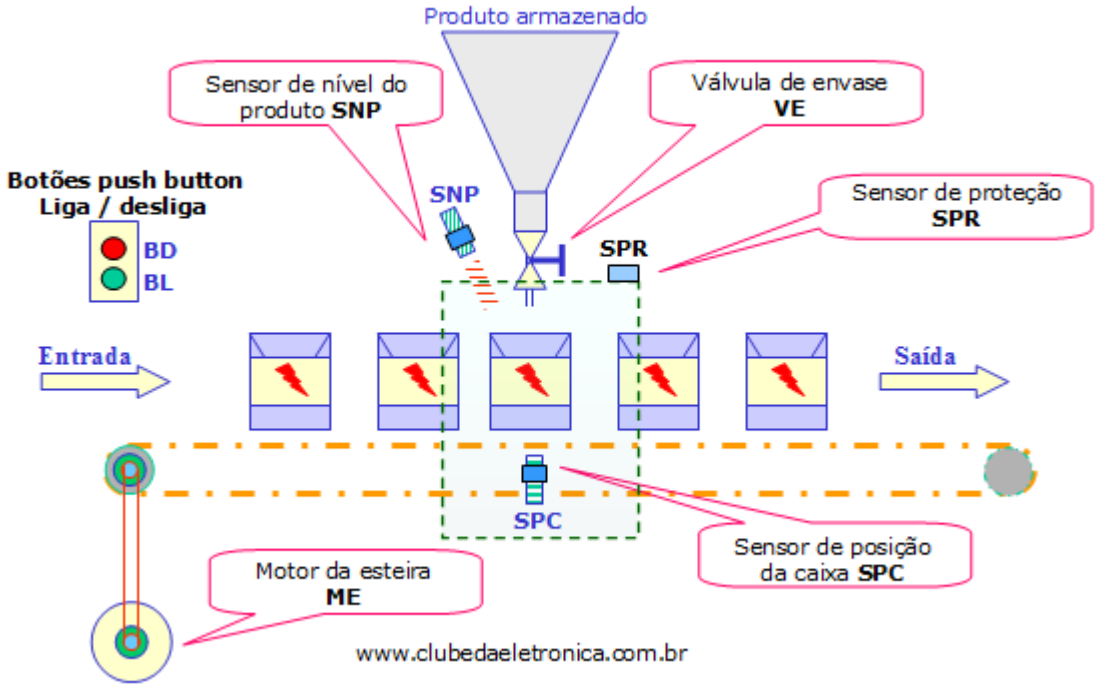
Potenciação



Praticando...

1- Sabendo-se que o timer possui uma base de tempo de 10ms. Elabore um programa capaz de converter o valor apresentado na memória do timer em segundos, e o apresente no display.

3- Deseja-se implementar um sistema para envase de produtos, conforme ilustrado na figura abaixo:



Descrição de funcionamento:

Ao pressionar o botão de Liga **(BL)**, dará início ao processo e a esteira **(ME)** será ligada e só pára quando a caixa chegar à posição de envase, dado pelo sensor de posição da caixa **(SPC)**, neste momento, abre-se a válvula de envase **(VE)** liberando o produto que terá seu nível controlado pelo sensor (**SNP)** e assim que este nível for alcançado liga-se novamente a esteira até que uma nova caixa chegue a posição de envase.

Condição de partida: O sistema só ligará se a tampa de proteção estiver fechada, ou seja, se o sensor de proteção **(SPR)** estiver acionado.

Pressionando o botão de desliga **(BD)** o ciclo será interrompido.

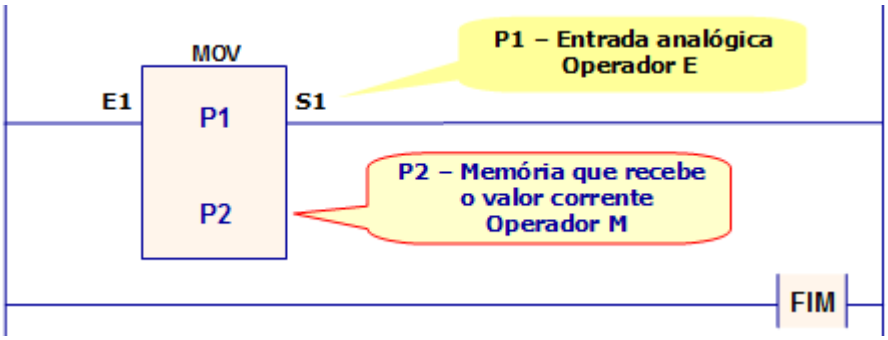
1- Defina as entradas e saídas (I/O) relacionando a sigla com o contato do CLP.
2- Fazer o diagrama de estado da solução proposta.
3- Fazer programa ladder da solução proposta.

Trabalhando com variáveis analógicas

Os Controladores Lógicos Programáveis são equipamentos digitais, ou seja, só entendem “0” e “1”, porém a grande maioria deles possui internamente conversores AD (entrada de sinais) e DA (saída de sinais) na maioria dos casos estes conversores são de 10 bits.

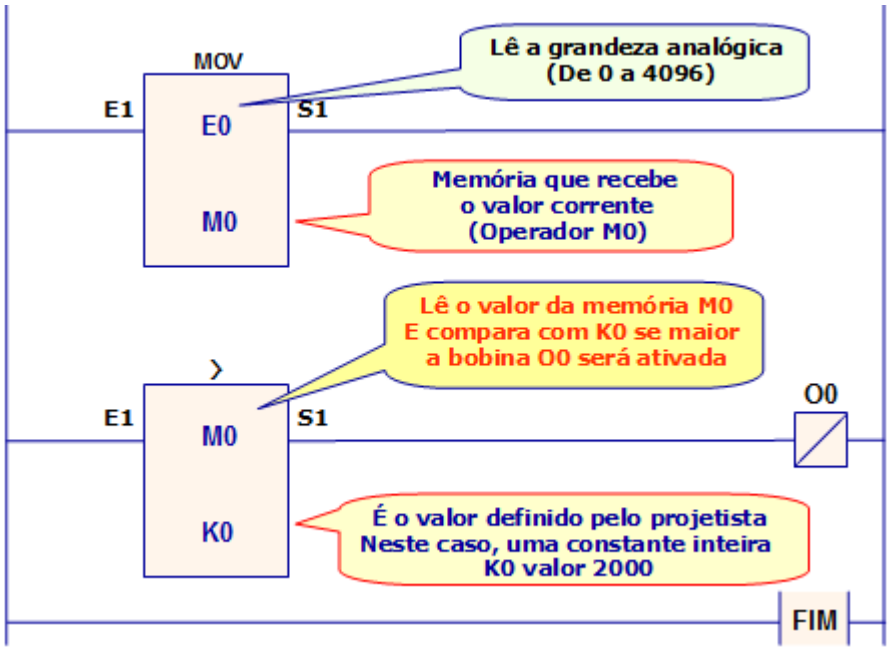
Os controladores lógicos são preparados para receber ou enviar sinais em tensão (tipicamente 0 a 10V) ou em corrente (tipicamente 4 a 20mA), cabendo ao usuário uma consulta ao manual do fabricante.

Segue um exemplo onde o valor corrente do operador E0 será transferido para a memória M0.



Blocos comparadores (=, ⟨⟩, ⟨, ⟨=, ⟩ , ⟩= e &)

Comparar duas variáveis e tomar decisão é sem dúvida de extrema importância para automação. Os CLPs disponibilizam vários blocos para este fim. Vejamos alguns exemplos:



Outros blocos de comparação

P1 Igual a P2 (P1 = P2)

O objetivo destes elementos é realizar a comparação entre operadores. Esta comparação é do tipo que verifica se o operando P1 é igual ao operando P2 Para inseri-lo no programa deve-se posicionar o cursor na posição desejada, selecionar no menu à esquerda o grupo "Comparação" e clicar no botão correspondente.

P1 diferente de P2 (P1 <> P2)

O objetivo destes elementos é realizar a comparação entre operadores. Esta comparação é do tipo que verifica se o operando P1 é diferente do operando P2. Para inseri-lo no programa deve-se posicionar o cursor na posição desejada, selecionar no menu à esquerda o grupo "Comparação" e clicar no botão correspondente.

P1 maior que P2 (P1 > P2)

O objetivo destes elementos é realizar a comparação entre operadores. Esta comparação é do tipo que verifica se o operando P1 é maior que o operando P2. Para inseri-lo no programa deve-se posicionar o cursor na posição desejada, selecionar no menu à esquerda o grupo "Comparação" e clicar no botão correspondente.

P1 maior ou igual a P2 (P1 >= P2)

O objetivo destes elementos é realizar a comparação entre operadores. Esta comparação é do tipo que verifica se o operando P1 é maior ou igual ao operando P2. Para inseri-lo no programa deve-se posicionar o cursor na posição desejada, selecionar no menu à esquerda o grupo "Comparação" e clicar no botão correspondente.

P1 menor que P2 (P1 < P2)

O objetivo destes elementos é realizar a comparação entre operadores. Esta comparação é do tipo que verifica se o operando P1 é menor que o operando P2. Para inseri-lo no programa deve-se posicionar o cursor na posição desejada, selecionar no menu à esquerda o grupo "Comparação" e clicar no botão correspondente.

P1 menor ou igual a P2 (P1 <=P2)

O objetivo destes elementos é realizar a comparação entre operadores. Esta comparação é do tipo que verifica se o operando P1 é menor ou igual ao operando P2. Para inseri-lo no programa deve-se posicionar o cursor na posição desejada, selecionar no menu à esquerda o grupo "Comparação" e clicar no botão correspondente.

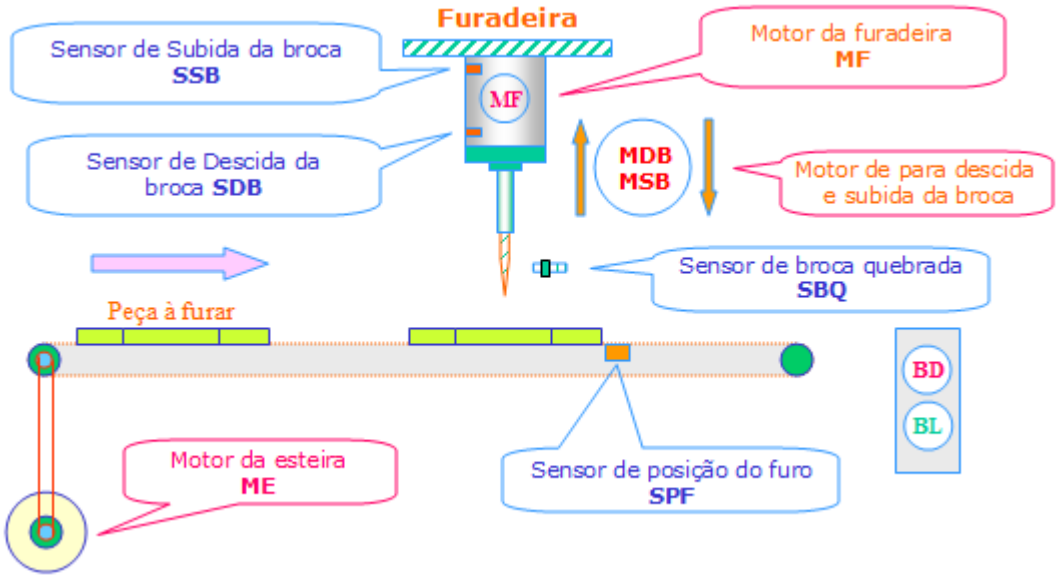
Teste lógico (P1 & P2)

O objetivo deste elemento é realizar a operação lógica AND (E) bit a bit entre dois operadores. Para inseri-lo no programa deve-se posicionar o cursor na posição desejada, selecionar no menu à esquerda o grupo "Comparação" e clicar no botão correspondente.

Praticando...

1- Elabore um programa capaz de:
   a. Ler um sinal analógico de 0 (0°C) a 5V (100°C).
   b. Apresentar o valor da temperatura no display.
   c. O circuito deverá acionar ligar uma ventoinha quando o set point (40°C) for alcançado
   d. A ventoinha deverá ficar ligada por 5 segundos.
   e. Implemente um set point ajustável via IHM

2- Deseja-se programar um sistema para furação de peças, conforme ilustrado na figura abaixo:



Descrição de funcionamento:

Ao pressionar o botão de Liga **(BL)**, dará inicio ao processo e a esteira **(ME)** será ligada e só para quando a peça chegar à posição de furo, dada pelo sensor de furo **(SPF)**, neste momento, o motor de furo **(MF)** e o motor de descida da broca **(MDB)** serão acionados até que o furo esteja completo, dado pelo sensor de descida da broca **(SDB)**, neste momento, é acionado o motor de subida da broca **(MSB)**. Deve-se manter o motor de furo acionado para que a broca não quebre. Quando o sensor de subida da broca **(SSB)** for alcançado o sistema recomeça.

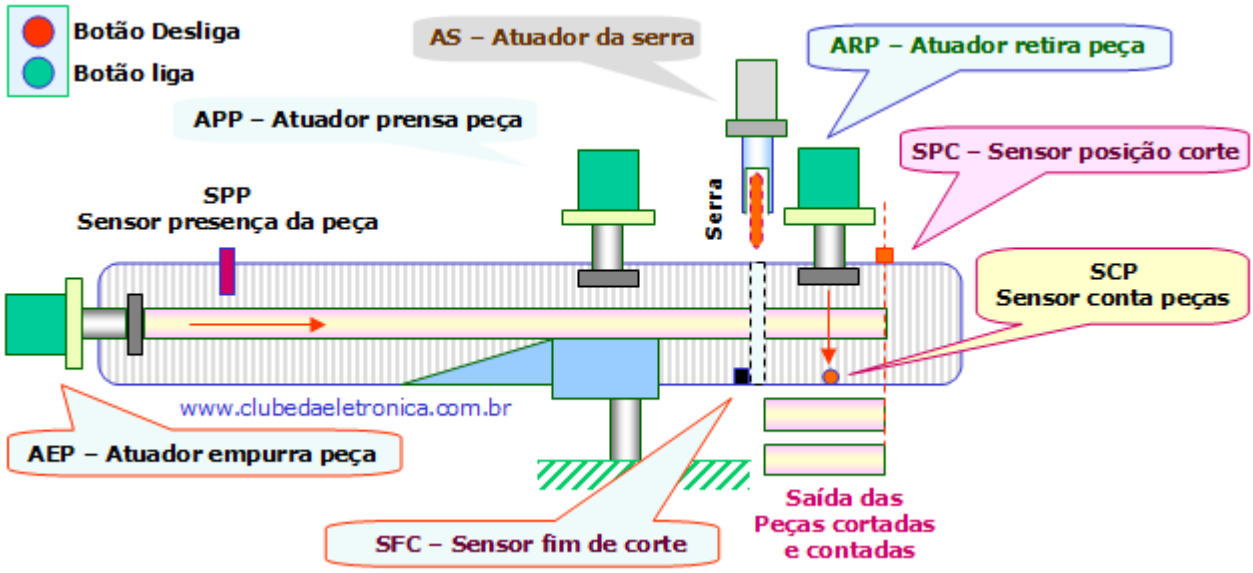
Condições de partida:

O processo só terá início se a broca estiver na posição alta, assim se estiver baixa ela deverá subir. Se estiver quebrada fica esperando a troca e o sistema não liga.

1- Defina as entradas e saídas (I/O) relacionando a sigla com o contato do CLP.
2- Fazer o diagrama de estado da solução proposta.
3- Fazer programa ladder da solução proposta (utilizar comentários)

Projeto máquina de corte automatizada

Deseja-se projetar uma serra automatizada, vejamos os critérios do cliente.



Descrição de funcionamento

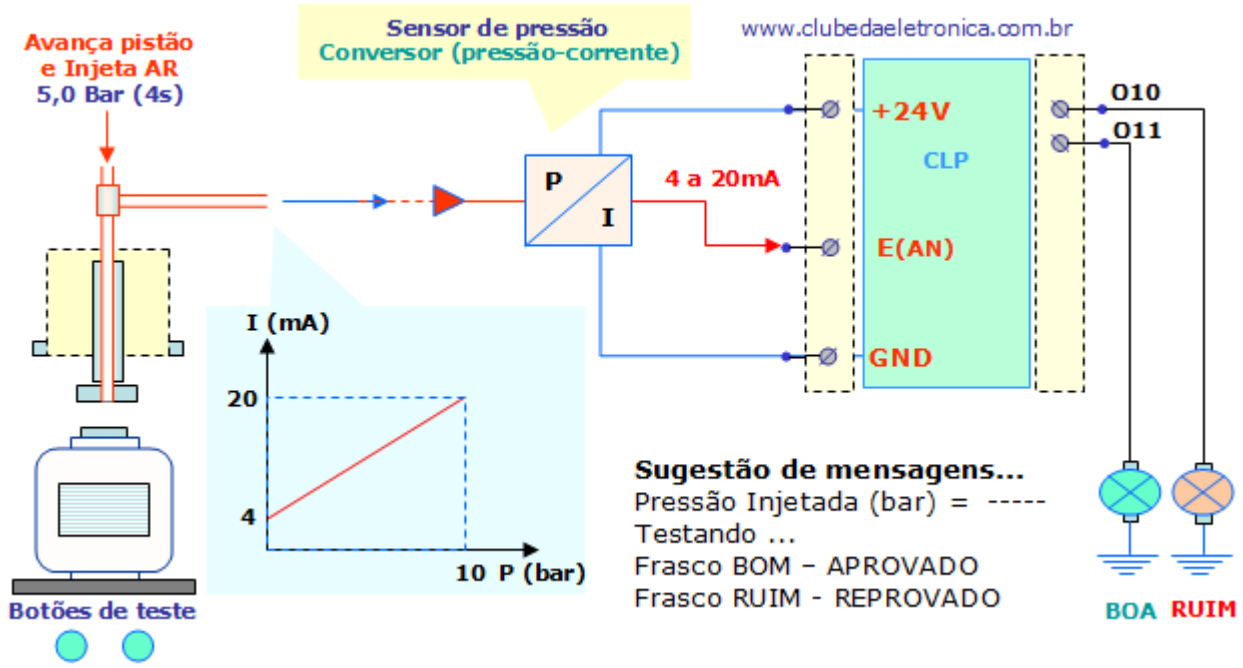
Coloca-se manualmente uma peça de 10 metros na máquina, um sensor detecta a presença da peça **(SPP)** se este for "1" e o botão liga **(BL)** for pressionado, ou seja, "1" a máquina liga, caso contrário, acenderá uma lâmpada **(L)** e na IHM deverá ser exibida a seguinte mensagem (COLOQUE A PEÇA). Uma vez **BL** e **SPP** = 1, liga o atuador de empurra peça **(AEP)** até que a peça encontre o sensor de posição de corte **(SPC)**, neste momento, desliga o **(AEP)** e liga o atuador prensa peça **(APP),** após 1 segundo liga a **SERRA** e o atuador da serra **(AS)** que ficarão acionados até que o sensor de fim de corte **(SFC)** seja alcançado e então aciona o atuador de retirada da peça **(ARP)** e o atuador que empurra a peça para corte **(AEP)**. Quando a décima peça passar pelo sensor **(SCP)** a lâmpada **(L)** acenderá e a mensagem **(COLOQUE A PEÇA)** aparecerá novamente.

Etapas para elaboração do projeto

- Definir o problema corrigindo possíveis falhas no processo.
- Identificar as variáveis e criar a lista de entradas e saídas.
- Montar o diagrama de estado da solução proposta.
- Implementar no Kit do Zap500 (não esquecer dos comentários e identificação das variáveis)

Teste de estanqueidade

Coloca-se o frasco manualmente na posição de teste, pressiona-se os botões de teste simultaneamente (push button), o cilindro avança, depois de 1 segundo injeta-se ar (5 bar) por 4s, desliga-se o ar, o sensor mede a pressão por um tempo de 5s), e se neste tempo a pressão cair o franco será rejeitado (exibir mensagem no display: **FRASCO RUIM**), se estiver não cair neste intervalo de tempo o frasco esta bom (exibir mensagem no display: **FRASCO BOM**).



"Toda idéia brilhante de hoje já foi uma idéia impraticável no passado."

Bill Gates.

www.clubedaeletronica.com.br

Referências bibliográficas:

- Circuitos digitais, Autor: Antonio Carlos de Oliveira Lourenço, Ed. Érica.
- http://www.plcopen.org/pages/tc1_standards/iec_1131_or_61131/
- http://www.cpdee.ufmg.br/~carmela/NORMA%20IEC%201131.doc
- http://www.software.rockwell.com/corporate/reference/Iec1131/
- http://www.plcopen.org/
- http://www.lme.usp.br/~fonseca/psi2562%20aula%206%20IHM.pdf
- http://www.teses.usp.br/teses/disponiveis/18/18133/tde-11072002-085859/
- http://www.redenet.edu.br/publicacoes/arquivos/20080108_144615_INDU-058.pdf
- http://www.corradi.junior.nom.br/modCLP.pdf
- http://www.cpdee.ufmg.br/~seixas/PaginaII/Download/DownloadFiles/