TEORIA DE CONTROLADOR LÓGICO PROGRAMÁVEL

Programação Instruction List (IL) – 2.a Parte

Prof. Dr. Cesar da Costa

E-mail: ccosta@ifsp.edu.br

Site: www.professorcesarcosta.com.br

INSTITUTO FEDERAL
SÃO PAULO
Campus São Paulo

❑ Temporizadores

❖ Implementação de um temporizador TON no CLP que segue a norma IEC 61131-3.

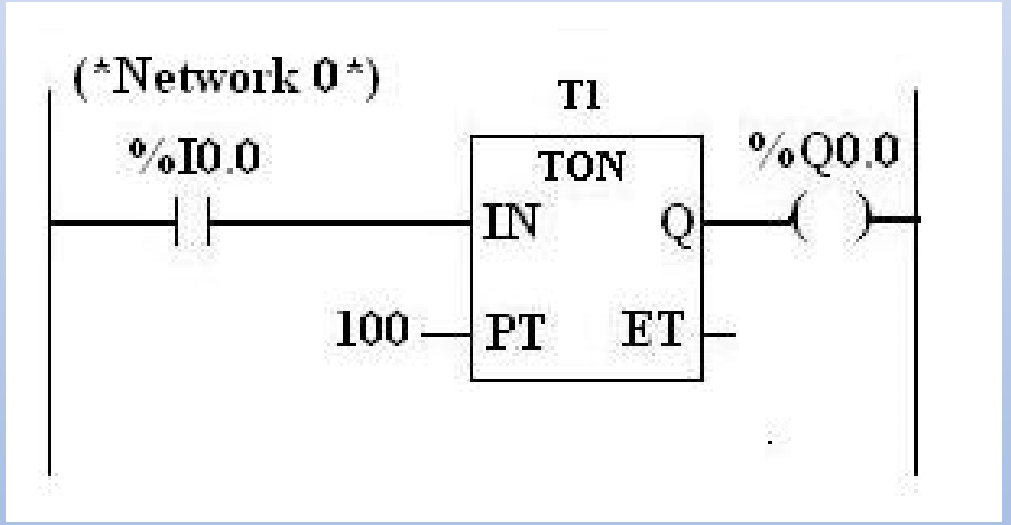
Lista de Instruções

(*Network 0*)

LD %I0.0

TON T1, 100

ST %Q0.0



❑ Contadores

- ❖ Os blocos podem ser chamados de várias maneiras e os fabricantes têm alguma liberdade de implementação.

- ❖ Pela norma IEC 61131-3 as funções podem ser chamadas diretamente, sem a necessidade de um operador que as preceda.

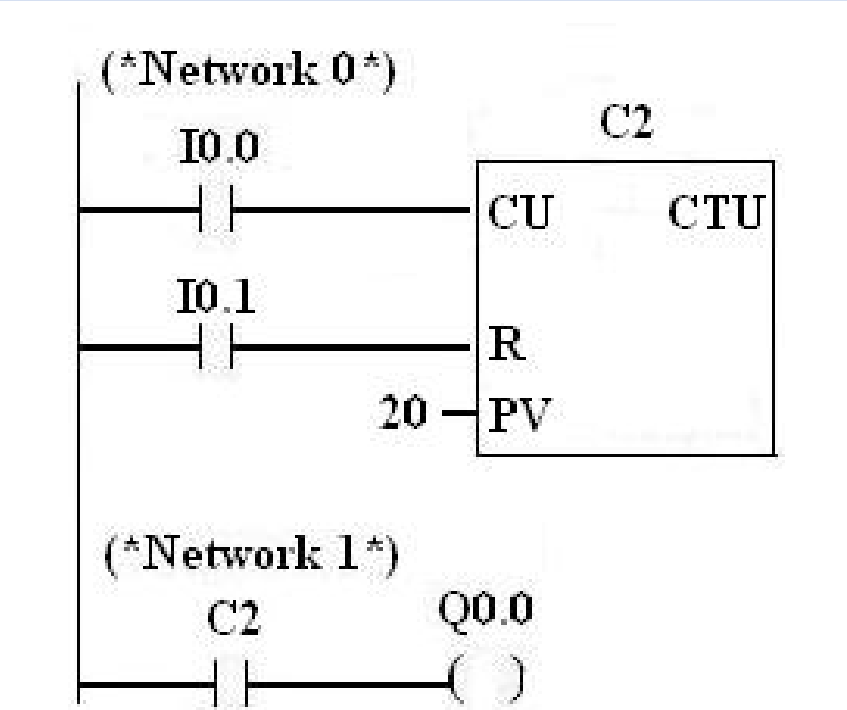
- ❖ De fato, o nome da função pode ser considerado um operador. Os parâmetros passados são: o endereço do contador, o contato que está ligado à entrada reset e o valor de PV.

9 – Contadores

- Implementação de um contador crescente em um CLP que segue a norma IEC 61131-3.

Lista de Instruções

```
(*Network 0*)

LD I0.0

LD I0.1

CTU C2, 20

(*Network 1*)

LD C2

ST % Q0.0
```

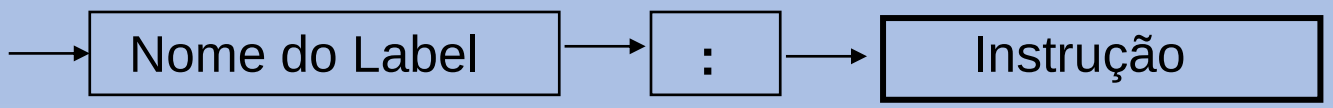


Instruções (I.L) Especiais

❖ **Label**

- Os **Labels** servem para identificar uma instrução como o **target** (alvo) de um **jump** e o uso deles é opcional. Eles são escritos no começo de uma instrução e podem estar apenas no começo de uma sequência.

- A forma de digitá-los é no começo da linha de programação, escreve-se o nome (que você vier adotar) do **label**, seguido de dois pontos..



Instruções (I.L) Especiais

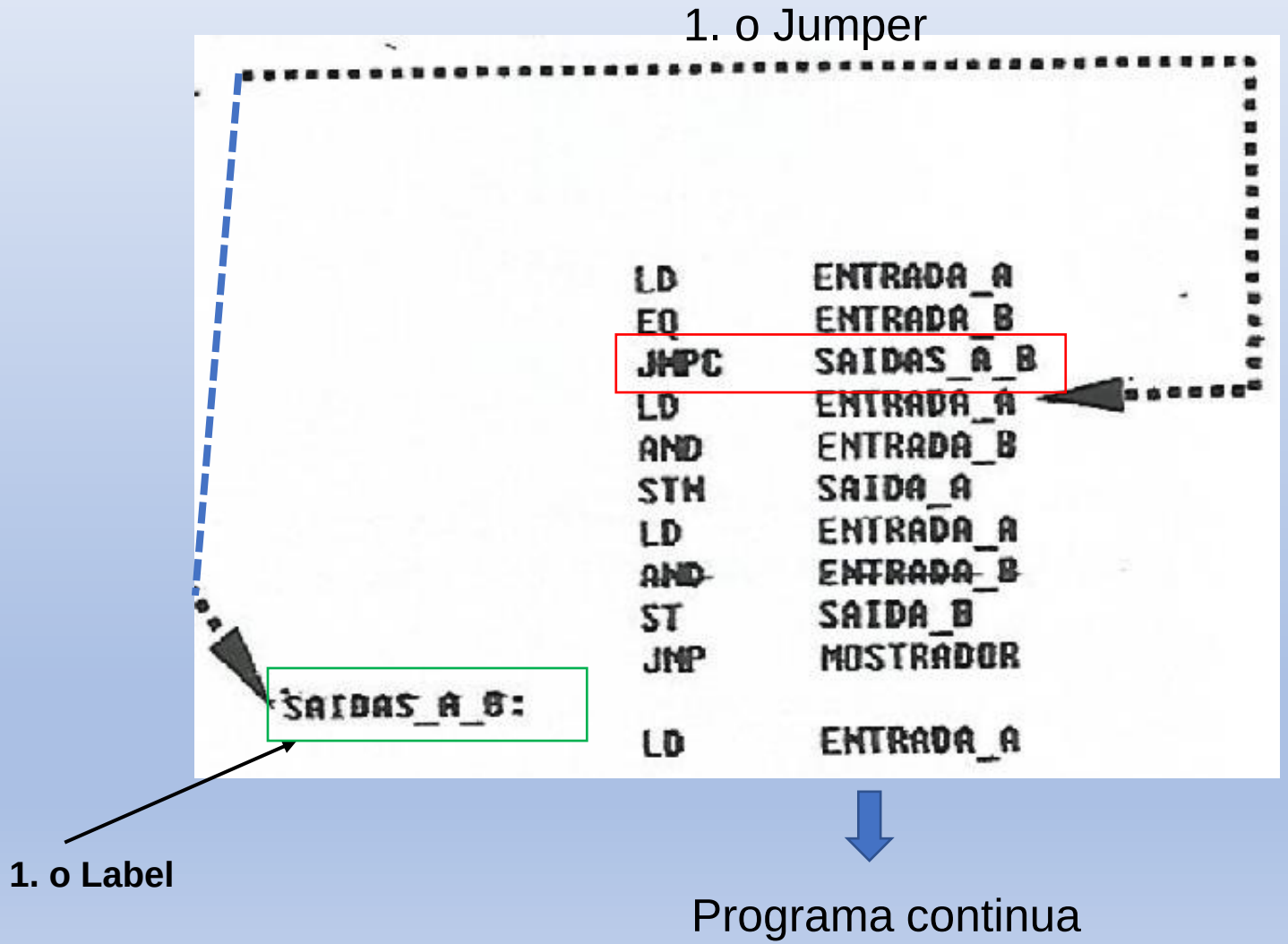
❖ **JMPC (desvio condicional)**

- O **JMPC** (jamper condicional para valor do registro igual a 1) será executado, fazendo com que o programa seja continuado (desviado) para o target **Label** . A partir do qual o programa deverá ser continuado.

- Se o registro tiver o valor igual a 0 a instrução **JMPC** não será executada e o programa será continuado para a próxima instrução.

Instruções (I.L) Especiais

- **JMPC (desvio condicional)**



Instruções (I.L) Especiais

- **JMPC (desvio condicional)**

Example:

```
LD      17
ST      lint            (* comment *)
GE      5
JMPC    next
LD      idword
EQ      istruct.sdword
STN     test
next:
```

Instruções (I.L) Especiais

- **JMPC (desvio condicional)**

Example of an IL program while using some modifiers:

```
LD      TRUE            (*load TRUE in the accumulator*)
ANDN    BOOL1           (*execute AND with the negated value of the BOOL1 variable*)
JMPC    label           (*if the result was TRUE, then jump to the label "label"*)
LDN     BOOL2           (*save the negated value of *)
ST      ERG             (*BOOL2 in ERG*)
label:

LD      BOOL2           (*save the value of *)
ST      ERG             (*BOOL2 in ERG*)
```

Instruções (I.L) Especiais

❖ **JMP (desvio incondicional)**

- Na instrução **JMP** (jamper incondicional) o desvio será executado incondicionalmente, ou seja, não importando o valor (0 ou 1) do registro de trabalho.

- O programa será continuado (desviado) para o **Label**, a partir do qual o programa deverá ser continuado.

Instruções (I.L) Especiais

Operador JMP

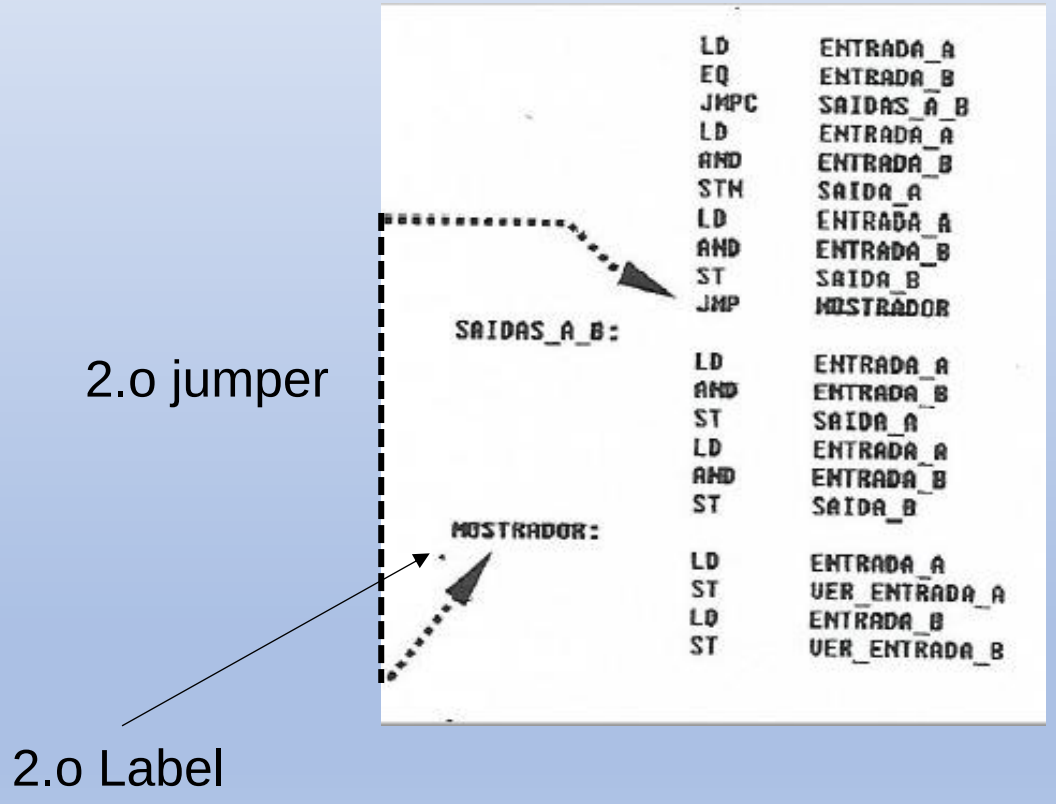
Operação desvia para uma etiqueta especificada
Modificadores permitidos C N
Operando etiqueta definida no mesmo programa IL

Exemplo:

```
(* o seguinte exemplo testa o valor de um seletor analógico (0 ou 1 ou 2)
(* para ligar uma dentre 2 saídas booleanas. A comparação "é igual a zero 0" é feita com
(* o operador JMPC                                                    *)

JMPex:    LD      selector    (* selector = 0 ou 1 ou 2 *)
          BOO                 (* conversão para booleano *)
          JMPC    test1       (* se selector = 0 então *)
          LD      true
          ST      bo0         (* bo0 := true *)
          JMP     JMPend      (* fim de programa *)
test1:    LD      selector
          SUB     1           (* decrementa selector:  agora é 0 ou 1 *)
          BOO                 (* conversão para booleano *)
```

Instruções (I.L) Especiais

- **JMP (desvio incondicional)**



Instruções (I.L) Especiais

Operador RET

Operação	termina a lista de instruções corrente. Se a IL é um subprograma, o resultado corrente é retornado ao programa que o chamou.
Modificadores permitidos	C N
Operando	(nenhum)

Exemplo:

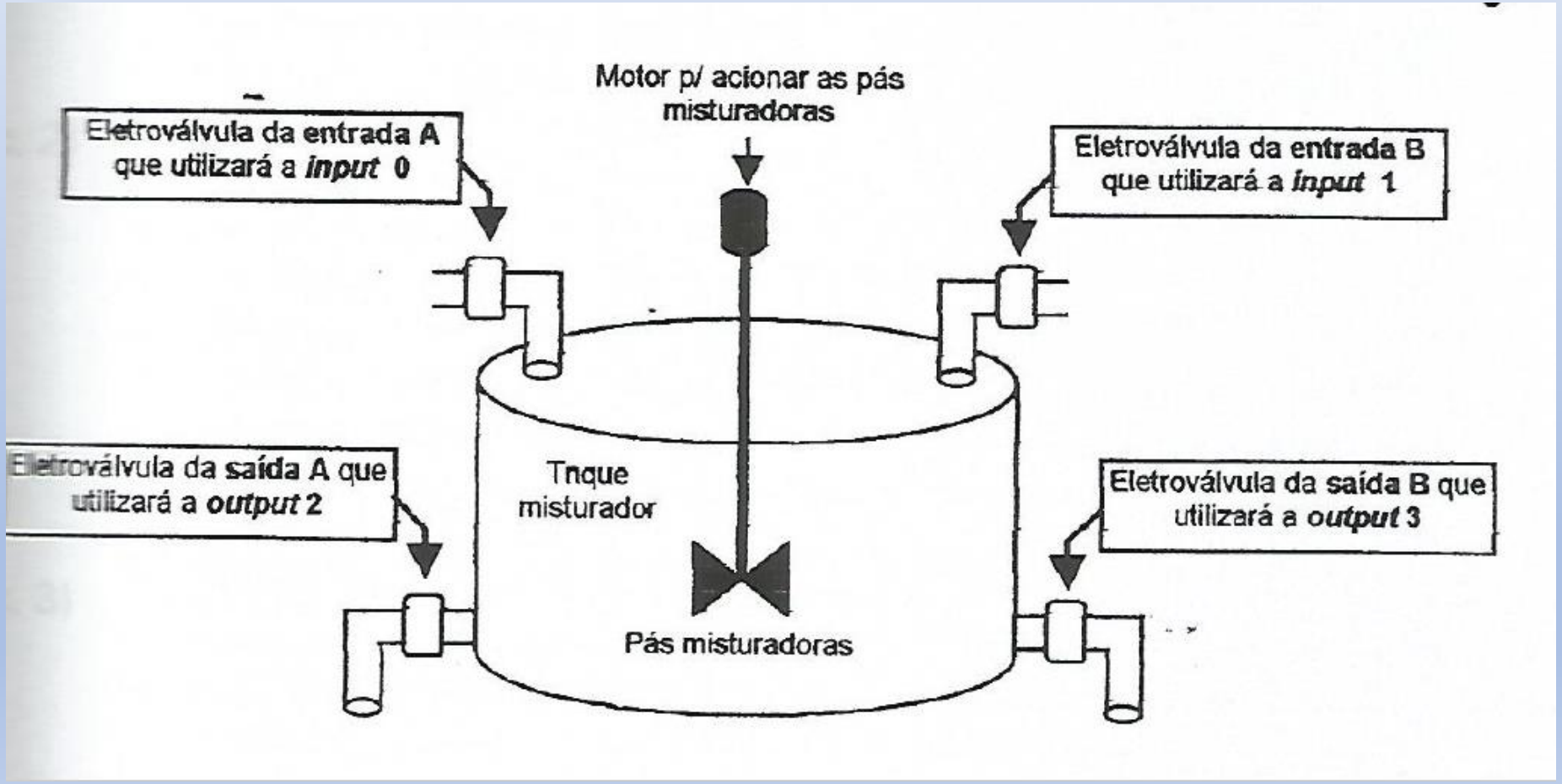
```
(* o seguinte exemplo testa o valor de um seletor analógico (0 ou 1 ou 2)
(* para ligar uma dentre 2 saídas booleanas. A comparação "é igual a zero 0" é feita com
(* o operador JMPC
```

Instruções (I.L) Especiais

```
JMPex:   LD     selector   (* selector = 0 ou 1 ou 2 *)
         BOO                (* conversão para booleano *)
         JMPC   test1      (* se selector = 0 então *)
         LD     true
         ST     bo0        (* bo0 := true *)
         RET               (* fim - retorne 0 *)
                           (* decremente seletor *)
test1:   LD     selector
         SUB    1          (* selector = 0 ou 1 *)
         BOO                (* conversão para booleano *)
         JMPC   test2      (* se selector = 0 então *)
         LD     true
         ST     bo1        (* bo1 := true *)
         LD     1          (* carrega valor real de selector *)
         RET               (* fim - retorna 1 *)
                           (* última possibilidade *)
test2:   RETNC             (* retorna se selector tem *)
                           (* um valor inválido *)
         LD     true
         ST     bo2        (* bo2 := true *)
         LD     2          (* carrega valor real de selector *)
                           (* fim - retorna 2 *)
```

Exercício de Aplicação 1: Tanque Misturador

- Em um determinado segmento de uma linha de processo, de uma indústria tem-se um tanque que recebe um determinado líquido, que deve permanecer continuamente agitado por meio de pás misturadoras, conforme mostra a Figura.



Condições de Funcionamento:

1) A vazão da **entrada A** é igual a vazão da **entrada B**, que também é igual a vazão da **saída A**, a qual é igual a vazão da **saída B**, ou seja, todas as vazões são iguais e , quando elas ocorrem, são todas constantes.

2) Tanto a **entrada A** como a **entrada B** despejam no tanque o mesmo tipo de líquido, seja por meio de só a **entrada A** aberta, seja por meio de só a **entrada B** aberta, ou ambas as entradas abertas simultaneamente.

3) O tanque deve permanecer cheio e não pode transbordar, então, se só uma das entradas abrir, a **saída A** deverá abrir; esta **saída A** deverá fechar quando a entrada que estiver aberta for fechada.

4) Porém, se as duas entradas forem abertas simultaneamente, também as duas saídas deverão ser abertas simultaneamente; obviamente as duas saídas deverão fechar quando as entradas que estiverem abertas forem fechadas..

Condições de Funcionamento:

Tabela 17 – TABELA PARA AUXILIAR A DECLARAÇÃO DAS VARIÁVEIS – DO SEGUNDO EXERCÍCIO

Variável	Campo		
	Name	***Type***	***Address***
Entrada A	ENTRADA_A	BOOL	%I0.0.0.0.0
Entrada B	ENTRADA_B	BOOL	%I0.0.0.0.1
Entrada A no painel	VER_ENTRADA_A	BOOL	%Q0.0.0.0.0
Entrada B no painel	VER_ENTRADA_B	BOOL	%Q0.0.0.0.1
Saída A	SAIDA_A	BOOL	%Q0.0.0.0.2
Saída B	SAIDA_B	BOOL	%Q0.0.0.0.3

Projeto a ser realizado:

1) Fazer o fluxograma funcional da automação do Tanque Misturador;

2) Fazer o mapa de E/S, com os endereços do CLP, e nome das variáveis do processo;

3) Fazer o programa em Instruction List;.

4) Testar o programa “off-line”;.

5) Criar uma tela de simulação;.

Possível Solução com a Linguagem I. L.:

```
1                   LD          ENTRADA_A    (* copia o valor da variável entrada A para o registro de trabalho*)
2                   EQ          ENTRADA_B    (* faz a comparação igual a da variável entrada B com o conteúdo  que
    estiver no registro de trabalho*)
3                   JMPC        SAIDAS_A_B (* se o valor da variável entrada A for igual ao valor da variável B, a instrução JMPC desvia
    para o Label SAIDAS_A_B, que esta posicionado na linha 11, a partir da qual o programa continua*)
4                   LD          ENTRADA_A
5                   AND         ENTRADA_B
6                   STN         SAIDA_A
7                   LD          ENTRADA_A
8                   AND         ENTRADA_B
9                   ST          SAIDA_B
10                  JMP         MOSTRADOR
11  SAIDAS_A_B:
12                  LD          ENTRADA_A
13                  AND         ENTRADA_B
14                  ST          SAIDA_A
15                  LD          ENTRADA_A
16                  AND         ENTRADA_B
17                  ST          SAIDA_B
18   MOSTRADOR:
19                  LD          ENTRADA_A
20                  ST          VER_ENTRADA_A
21                  LD          ENTRADA_B
22                   ST         VER_ENTRADA_B
```

Explicações da Possível Solução em I. L.:

1) Esta solução de programação não esta completa, por exemplo, falta o acionamento contínuo do motor agitador (chave LIGA e DESLIGA);

2) As linhas de programa 4 a 10 indicam o setor do programa onde as entradas são diferentes;

3) As linhas de programa 12 a 17 indicam o setor do programa onde as entradas são iguais;

4) As linhas de programa 19 a 22 indicam o setor do programa que apresenta os status das entradas;

5) As variáveis VER_ENTRADA_A e VER_ENTRADA_B são criadas para monitorar o estado das entrada A e B.

Conclusões

Referência

http://professorcesarcosta.com.br/upload/imagens_upload/Apostila%20-%20CLP%20-%20Lista%20de%20instru%C3%A7%C3%B5es.pdf

http://professorcesarcosta.com.br/upload/imagens_upload/Apostila%20Codesys%20Avancada.pdf

http://professorcesarcosta.com.br/disciplinas/n7clpteclp

http://professorcesarcosta.com.br/upload/imagens_upload/Apostila_do_Curso_Clp-1.pdf

INSTITUTO FEDERAL
SÃO PAULO
Campus São Paulo