

# Objetivo

**Capacitar os participantes a utilizar a ferramenta de programação A1, proporcionando conhecimentos sobre as funções e recursos do software e os conceitos padronizados pela norma, atendendo os seguintes requisitos:**

- ◆ **Desenvolver programas de acordo com o padrão IEC 61131-3;**
- ◆ **Conhecer as vantagens práticas da Norma (Programação estruturada);**
- ◆ **Capacitar o entendimento de um programa IEC;**
- ◆ **Conhecer as características das linguagens de programação padronizadas.**

# Agenda

◆ **Norma IEC 61131-3**

◆ **Software A1**

◆ **Exercícios**

---

## **Programação IEC 61131-3 :**

**Mudando os conceitos de programação da  
Automação Industrial**

**-**

**o estado atual, as ferramentas para  
estruturação, as atividades e bibliotecas**

---

## Imagine

**Você está:**

- **Trabalhando com 4 marcas de controladores diferentes**
  - **Usando diferentes linguagens de programação**
  - **Se esforçando para adequar a linguagem dos engenheiros de software com os engenheiros eletricitas e de manutenção no chão de fábrica**
  - **E vendo o seu concorrente se saindo melhor**
-

---

# PROPÓSITOS PARA OS SISTEMAS ABERTOS:

- ◆ Diferentes linguagens de programação;
  - ◆ Qualidade do software;
  - ◆ Custo do software;
  - ◆ Portabilidade de aplicações;
  - ◆ Reutilização de software.
-

**A variedade atual de problemas pode ser amplamente reduzida pela padronização**

**... e qual padrão está disponível ?**

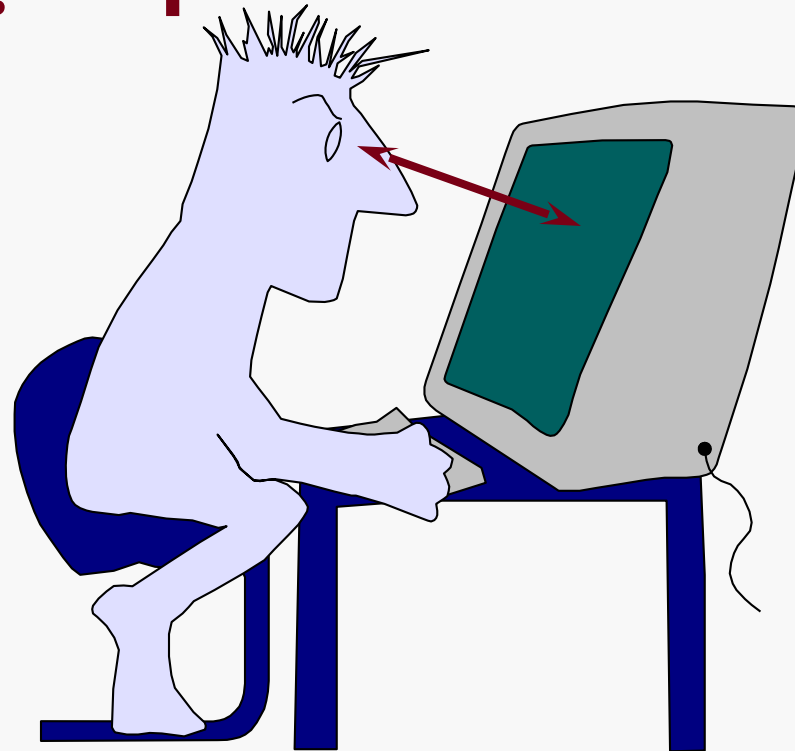
---

# Linguagens de Programação IEC 61131-3/ Programação para Controle Industrial

...com suporte para pessoas  
com diferentes formações



# Linguagens de Programação IEC 61131-3/ Programação para Controle Industrial



**A interface entre o programador e o sistema de controle**



# **NORMA IEC- 61131-3**

---

# Padronização Internacional de Linguagens

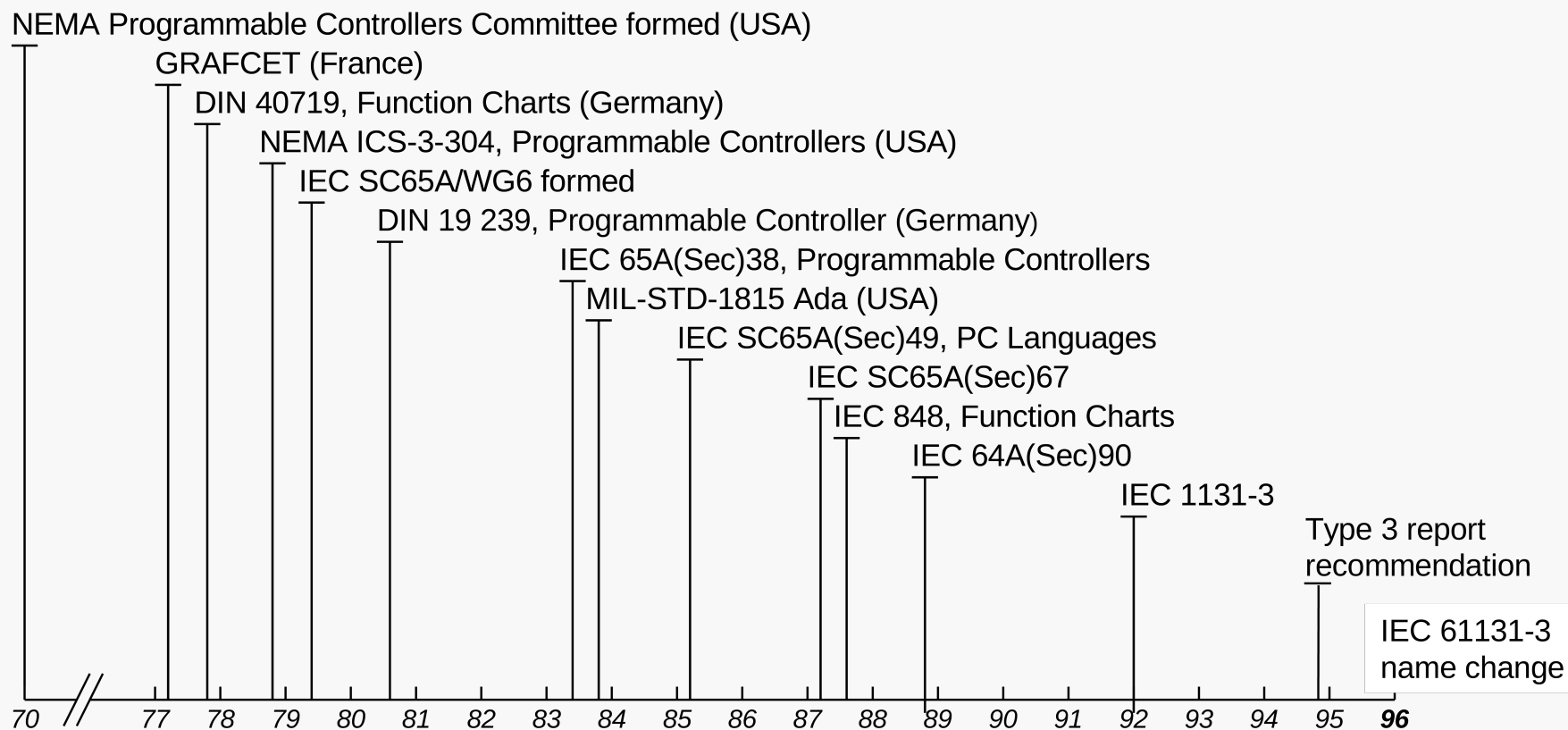


Gráfico 1 – Evolução da padronização das linguagens de programação.

# O que a norma visa padronizar

Parte	Título	Conteúdo	Publicação
Parte 1	General Information	Definição da terminologia e conceitos.	2003 (2ª Ed.)
Parte 2	Equipment requirements and tests	Teste de verificação e fabricação eletrônica e mecânica.	2003 (2ª Ed.)
Parte 3	Programmable Languages	Estrutura do software do CLP, linguagens e execução de programas.	2003 (2ª Ed.)
Parte 4	User guidelines	Orientações para seleção, instalação e manutenção de CLP's.	2004 (2ª Ed.)
Parte 5	Communications	Funcionalidades para comunicação com outros dispositivos.	2000 (1ª Ed.)
Parte 6	Reservada		
Parte 7	Fuzzy Control Programming	Funcionalidades de software, incluindo blocos funcionais padrões para tratamento de lógica nebulosa dentro de CLP's.	2000 (1ª Ed.)
Parte 8	Guidelines for the Application and Implementation of Programming Languages	Orientações para implementação das linguagens IEC 1131-3.	2003 (2ª Ed.)

# Vantagens da IEC 61131-3 para programar

## ◆ **Padrão Internacional Aceito**

- *Passo a passo de como os fornecedores deverão atendê-la;*
- *Estruturas uniformes.*

## ◆ **Gera economia do seu tempo**

- *Modelo de software e definição sólida dos tipos de dados padrões os “Data Types”;*
- *Você aprende apenas uma vez para diferentes tipos de controle;*
- *Reduz a dificuldade de entendimento e os erros;*
- *Funções e blocos de funções padrões;*
- *Orienta a reutilização do software testado.*

## ◆ **Suporte seguro e qualidade na programação**

- *Fácil e confortável estruturação;*
- *Erros na programação de tipos de dados proibidos.*

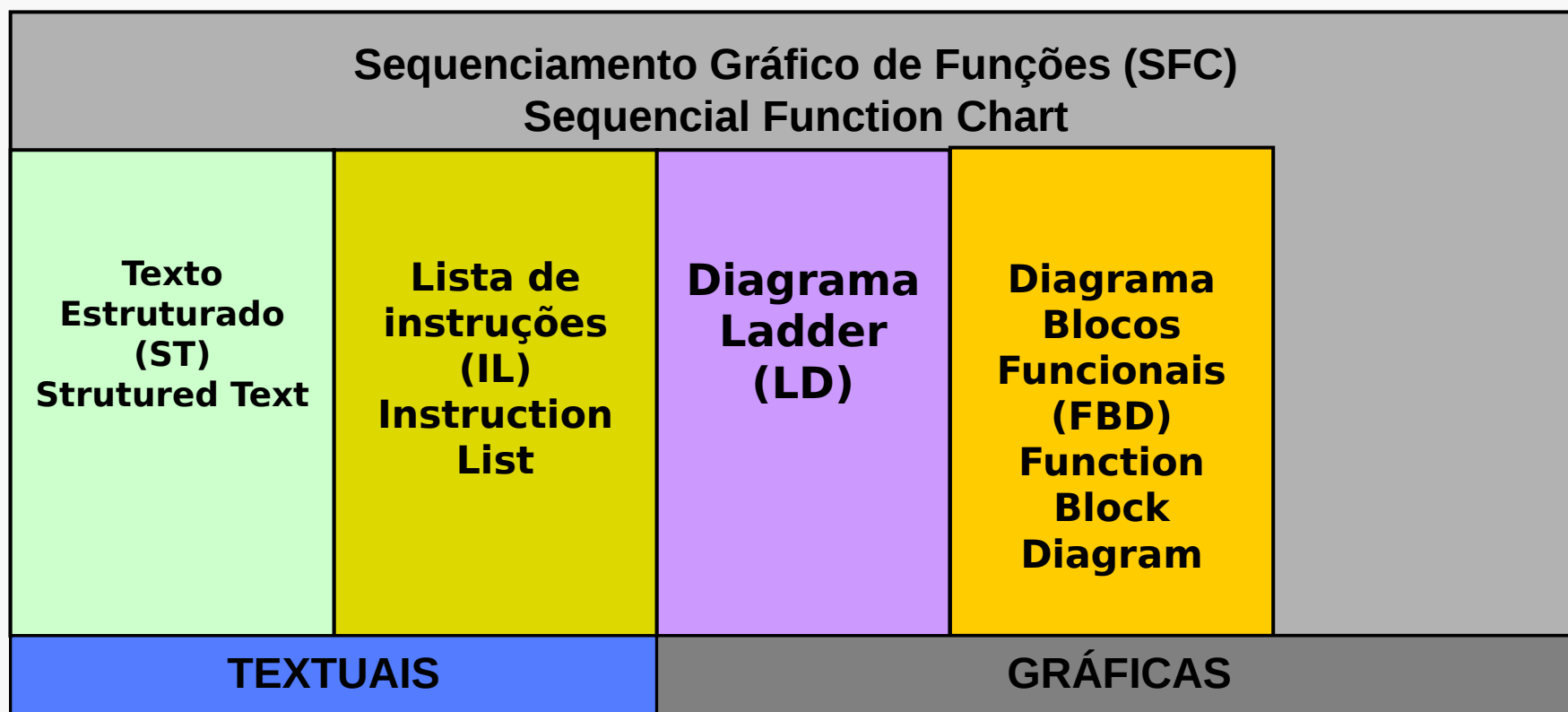
## ◆ **Oferece a melhor linguagem para cada problema**

- *Explicação consistente das cinco linguagens;*
- *Duas linguagens textuais e gráficas;*
- *Estruturação de linguagem, permitindo revisão;*
- *Disponibilidade de linguagem de alto nível;*
- *Possibilidade de “misturar” diferentes tipos de linguagens.*

# **Norma IEC 61131-3**

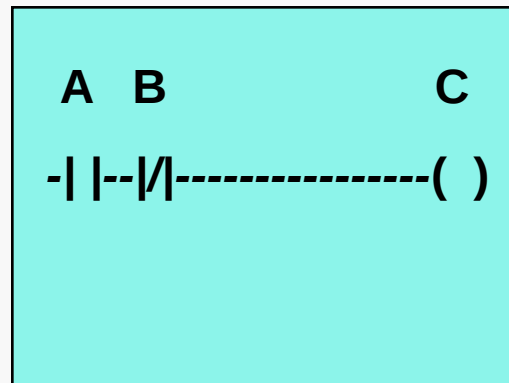
***Elementos Comuns***

***Linguagens de Programação***



# Diagrama Ladder (LD)

- ◆ Padronizada, conjunto reduzido de símbolos da programação ladder convencional



# Lista de Instruções (IL)

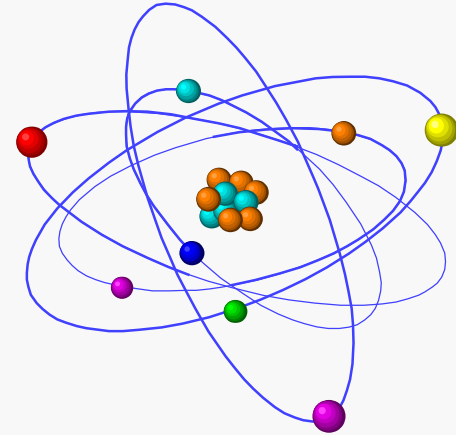
- ◆ Apenas uma operação, tal como o armazenamento de um valor, é permitido por linha de programa

LD	A
ANDN	B
ST	C



# Texto Estruturado (ST)

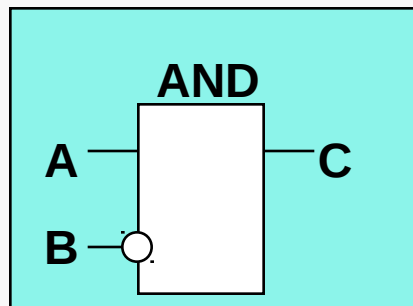
- ◆ Linguagem estruturada de alto nível
- ◆ Sintaxe semelhante ao Pascal



```
C:= A  AND  NOT B
```

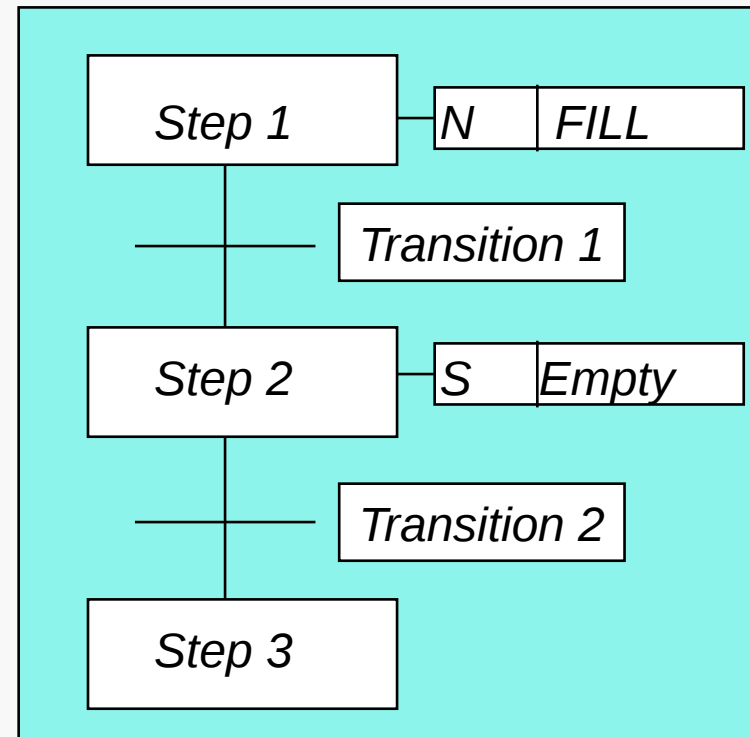
# Diagramas de Blocos de Função (FBD)

- ◆ Linguagem gráfica, amplamente usada na Europa
- ◆ Permite que os elementos de programa sejam representados como blocos para serem interligados de forma análoga ao diagrama de circuitos
- ◆ Usada em muitas aplicações que envolvem o fluxo de informação ou dados entre os componentes de controle



# Sequenciamento gráfico de funções SFC

- Baseado no Grafcet e Redes de Petri
- Padrão para programação de processos Batch
- Adequada para Estruturação de programas
- Controle de estados - máquina de estados
- Tomada de decisão – árvore de decisões



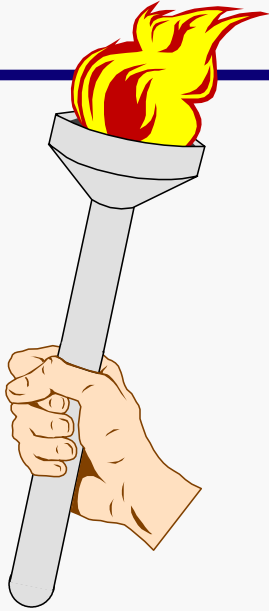
## Elementos de programação em **SFC**

**Passo** : estado do programa onde as ações são executadas

**Transição** : condição pela qual o programa muda de estado, passando de um ou mais passos antecessores para um ou mais passos sucessores

**Ação** : atividade de controle executada em um determinado passo

**Ramificação** : permite gerar divergência e convergência na sequência do programa



## **Missão da PLCopen**

**Nós queremos ser a associação líder na  
definição dos assuntos relacionados  
à programação de controle  
para suportar o uso  
de normas internacionais nesta área.**

**A PLCOpen foi fundada em 1992, a qual é uma associação independente destinada a promover e suportar o uso da norma IEC 61131-3.**

**Site: <http://www.plcopen.org>**

**Site: <http://www.iec61131.com.br>**

---

# PARTE COMUM

---

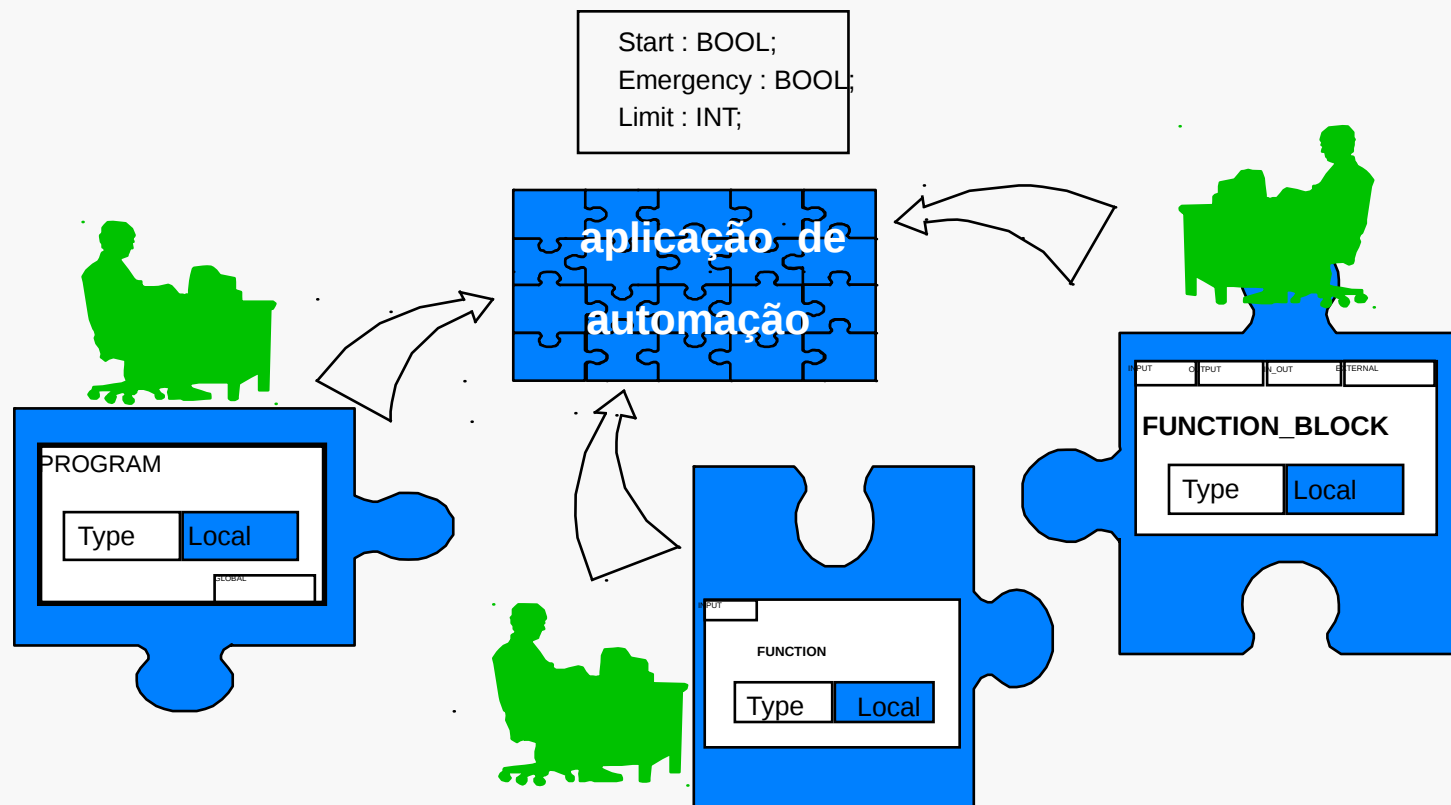
# **A norma IEC 61131-3**

*Elementos Comuns*

*Linguagens de Programação*



# Programas : projeto hierarquizado

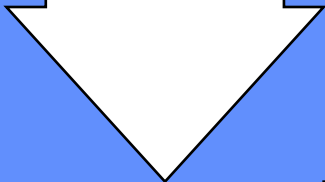


# Norma IEC 61131-3


***Elementos Comuns***

***Linguagens de Programação***

*De cima  
para baixo*



*De baixo  
para cima*



## ELEMENTOS COMUNS

Tipos de Dados e Variáveis  
Configuração, Recursos, Tarefas

**POUs - Unidades de Organização de  
Programa**

- \* Funções
- \* Blocos de Função (FB's)
- \* Programas

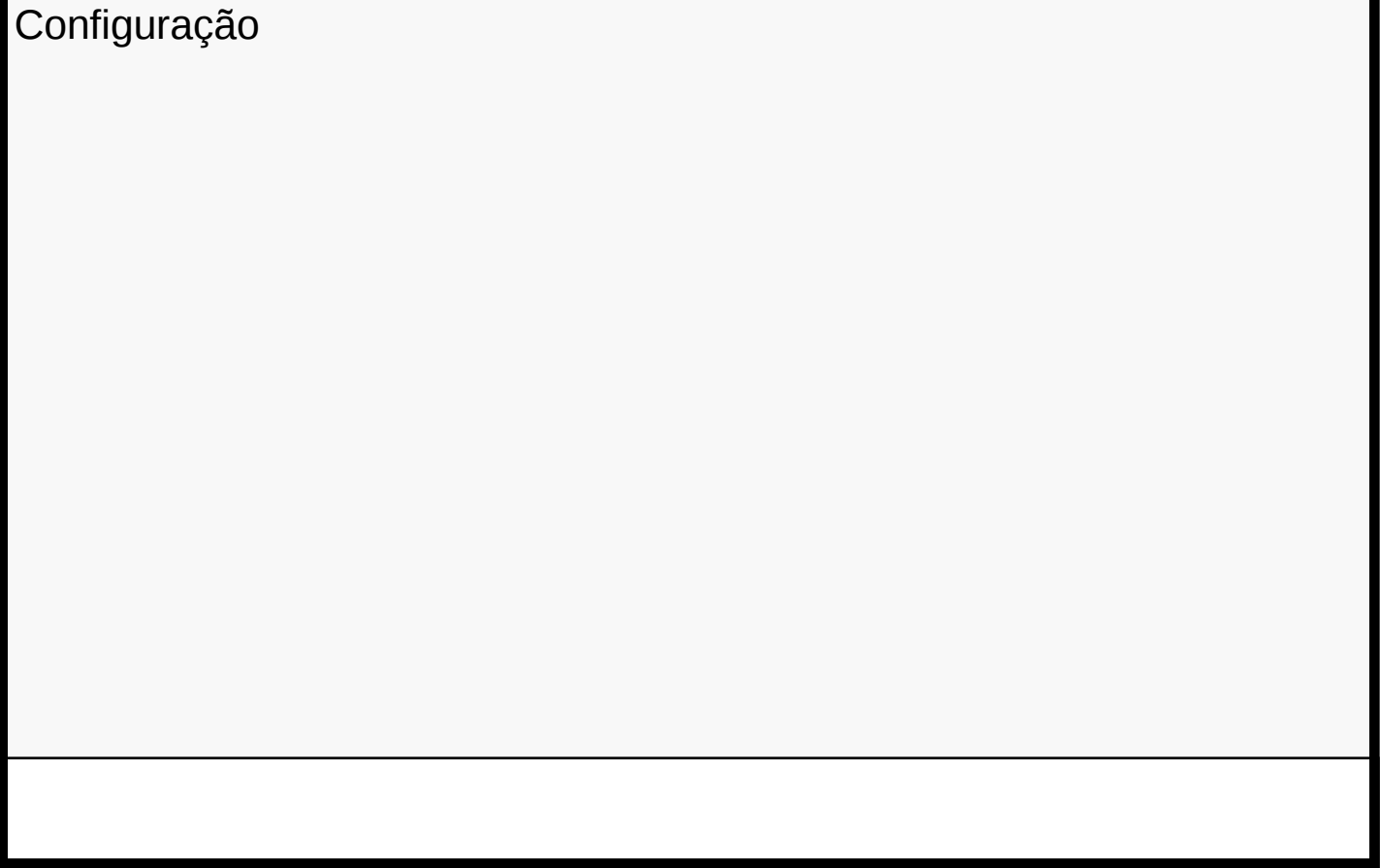
---

# MODELO DE SOFTWARE

- **Configuração**
  - **Recursos**
  - **Tarefas**
  - **Blocos Funcionais**
  - **Funções**
  - **Access Path**
  - **Fluxo de Controle**
-

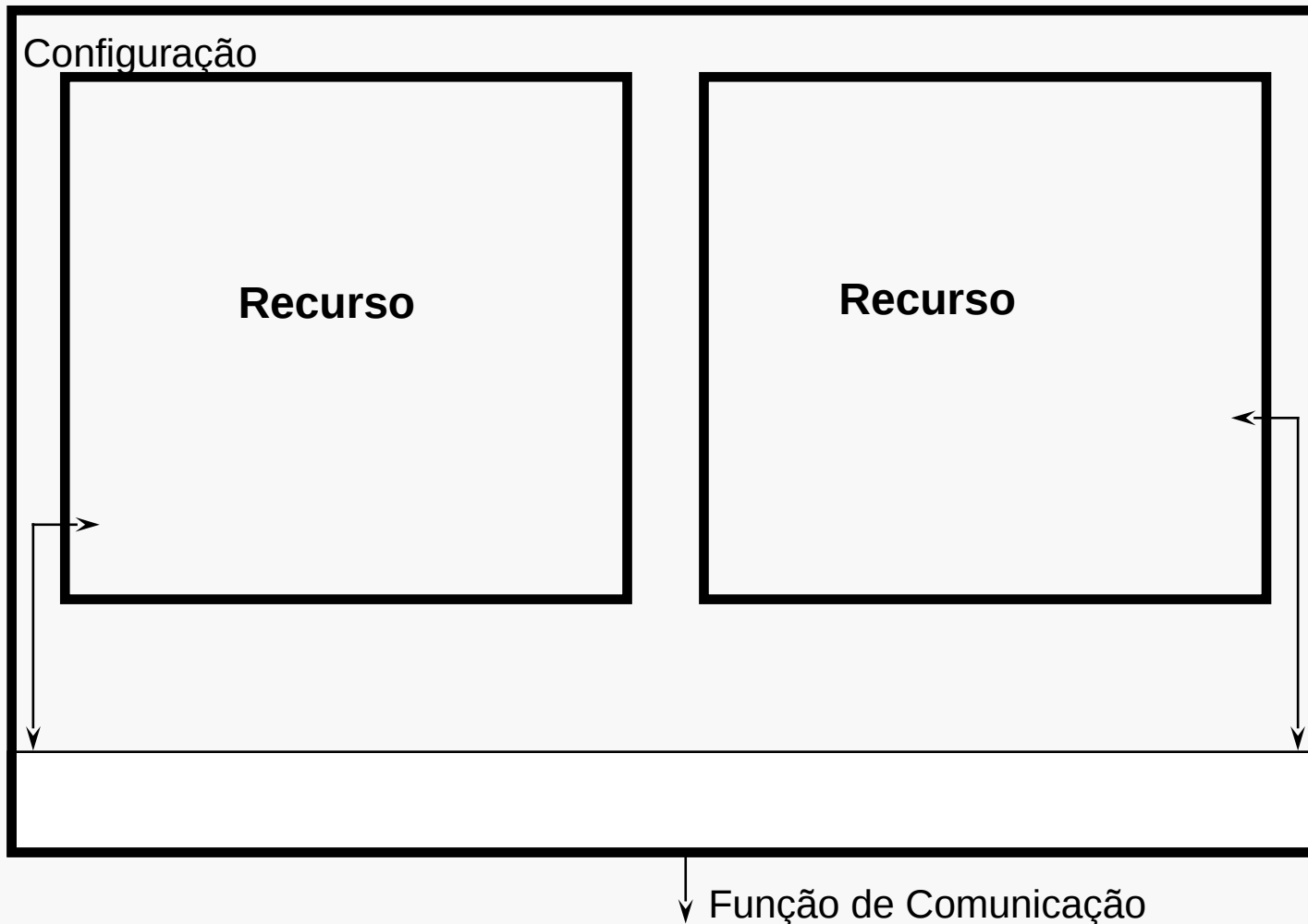
# IEC 61131-3 Modelo de Software

Configuração

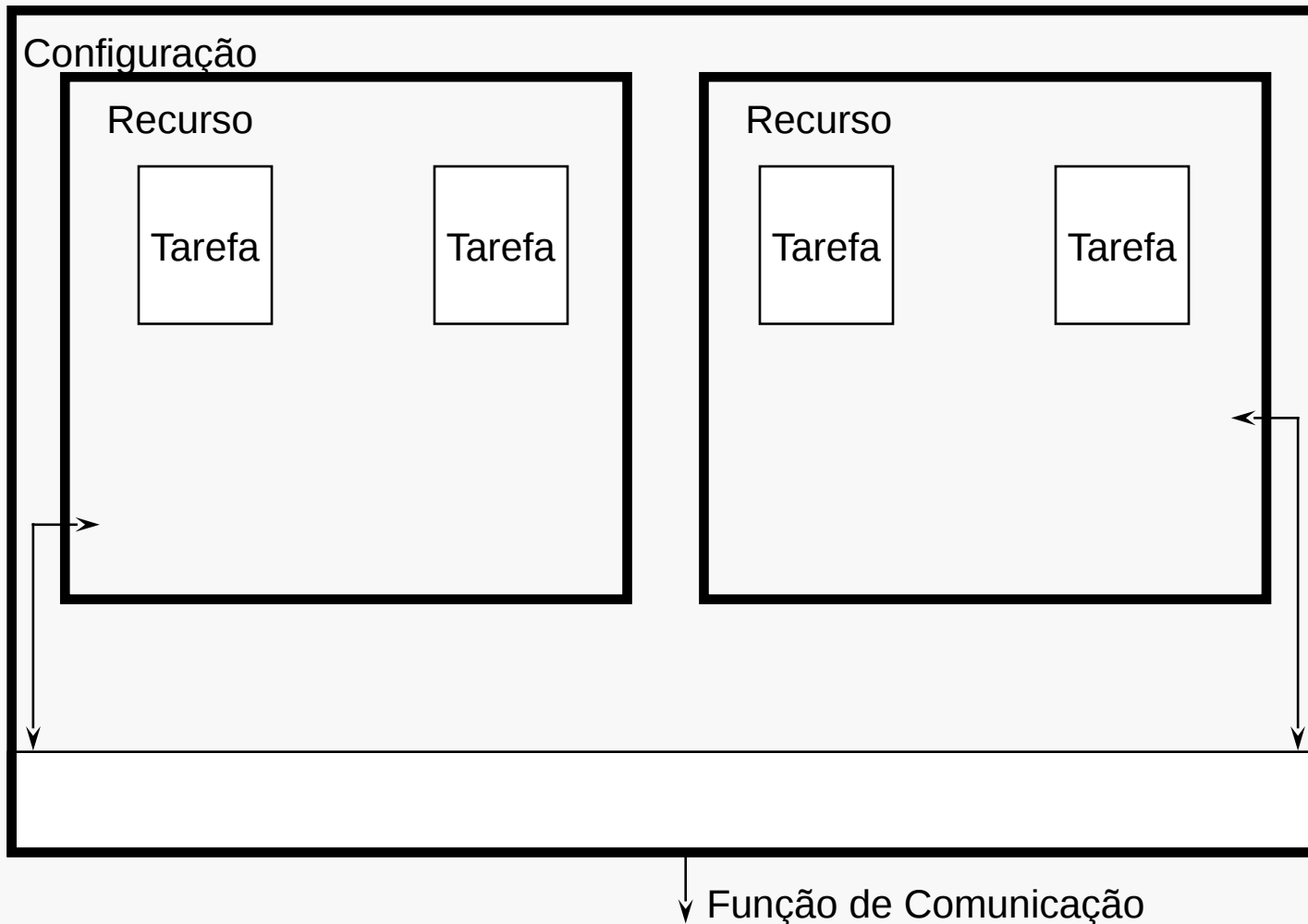


↓ Função de Comunicação

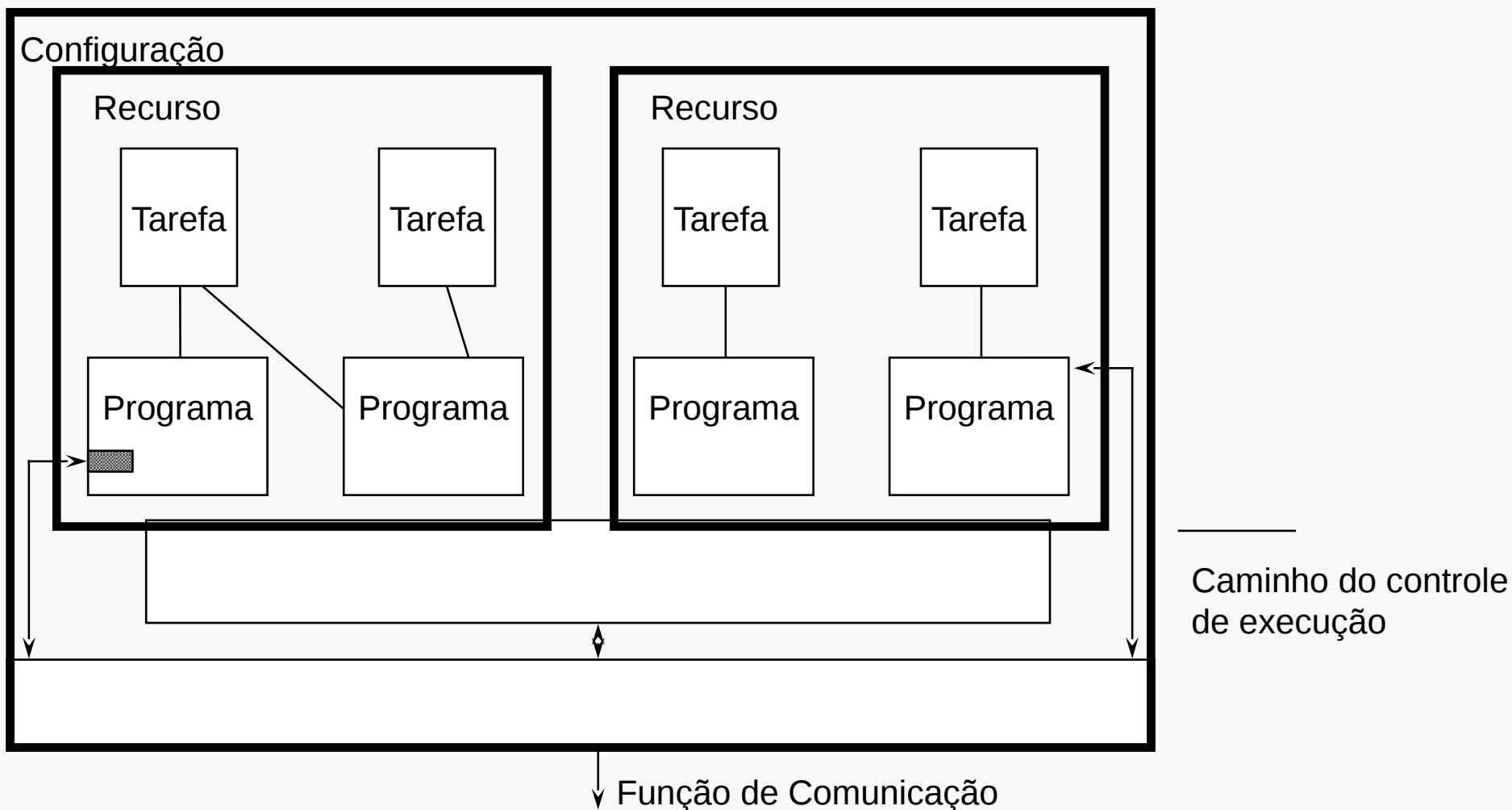
# IEC 61131-3 Modelo de Software



# IEC 61131-3 Modelo de Software

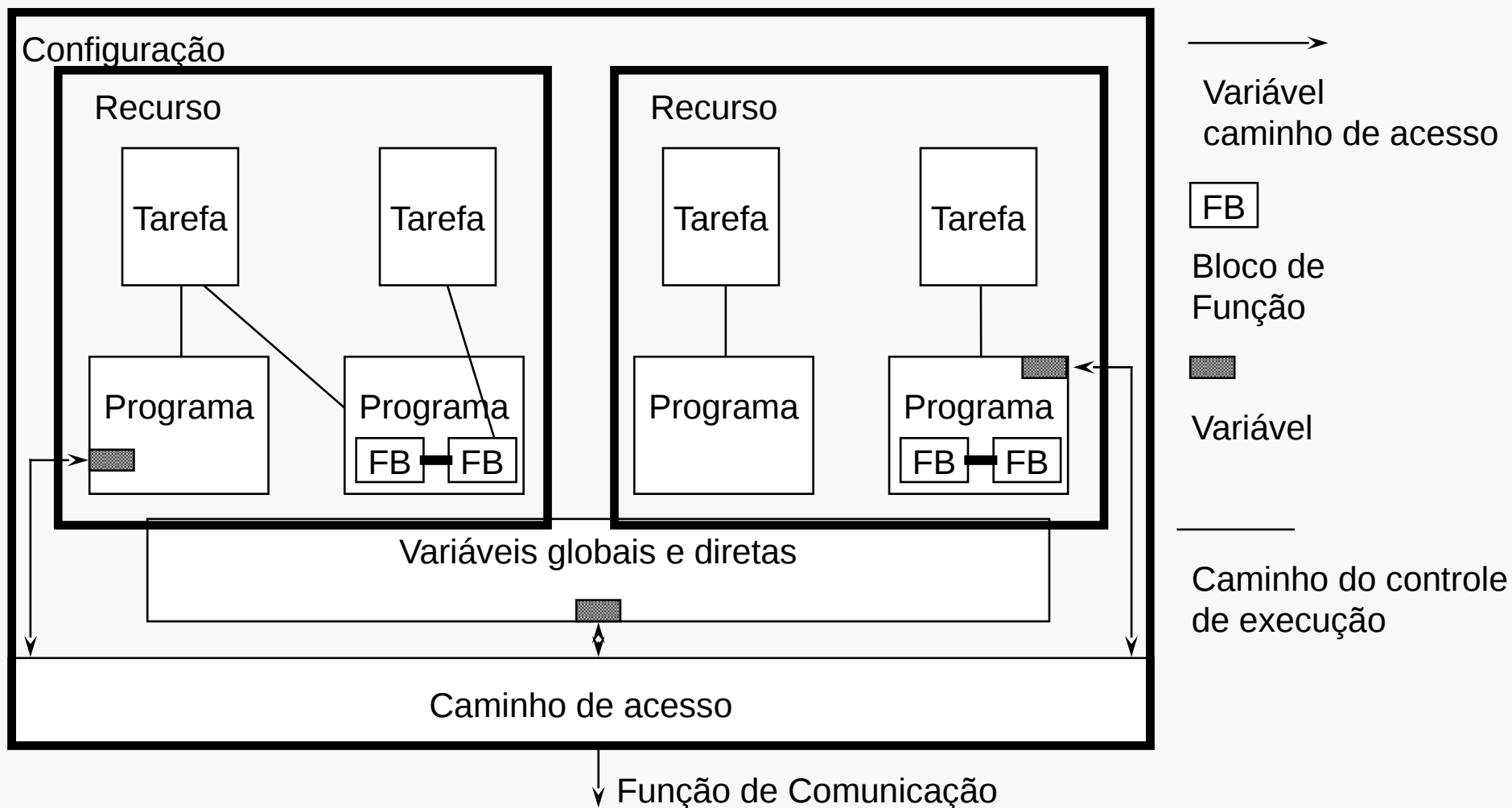


# IEC 61131-3 Modelo de Software

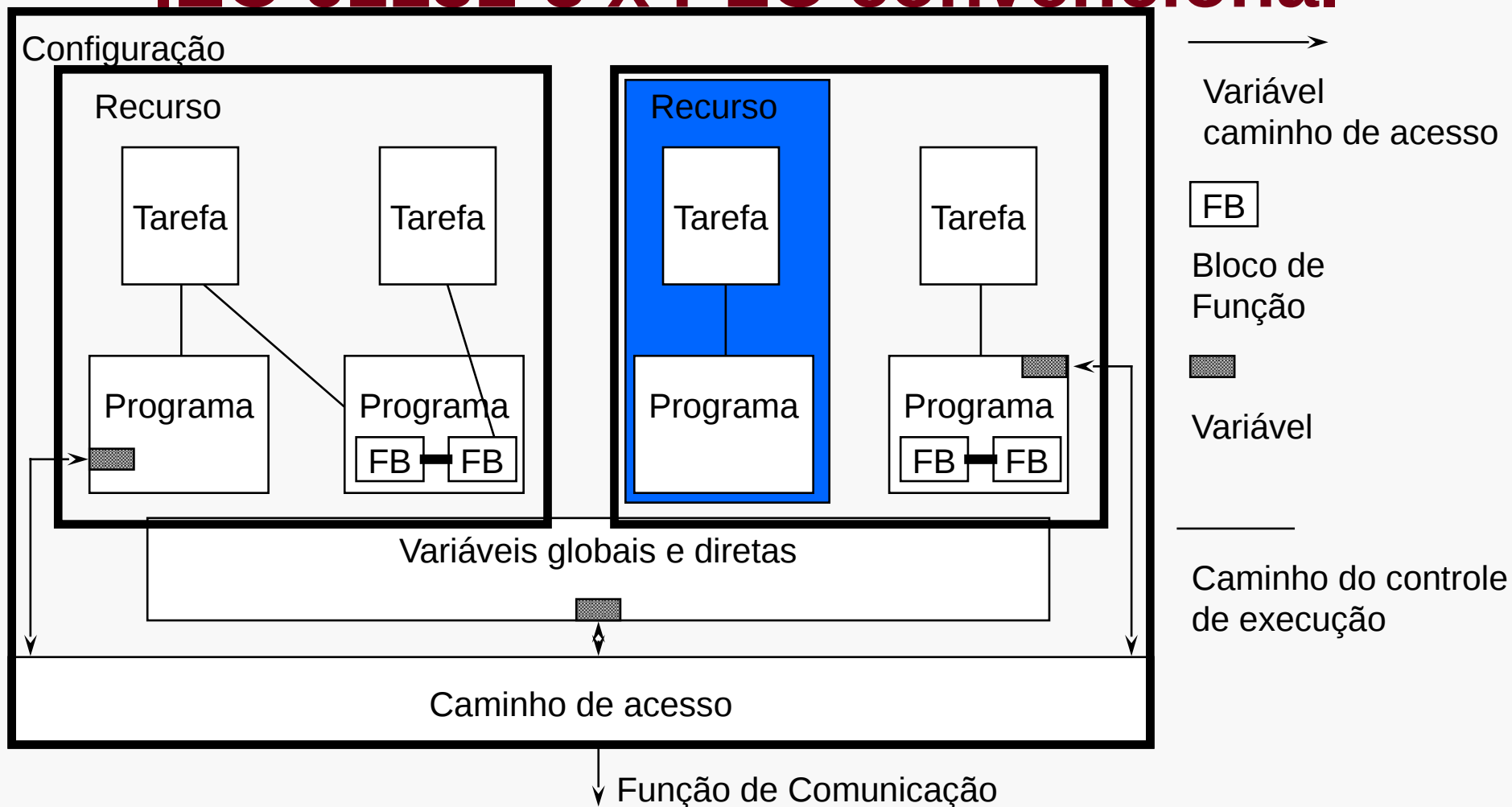




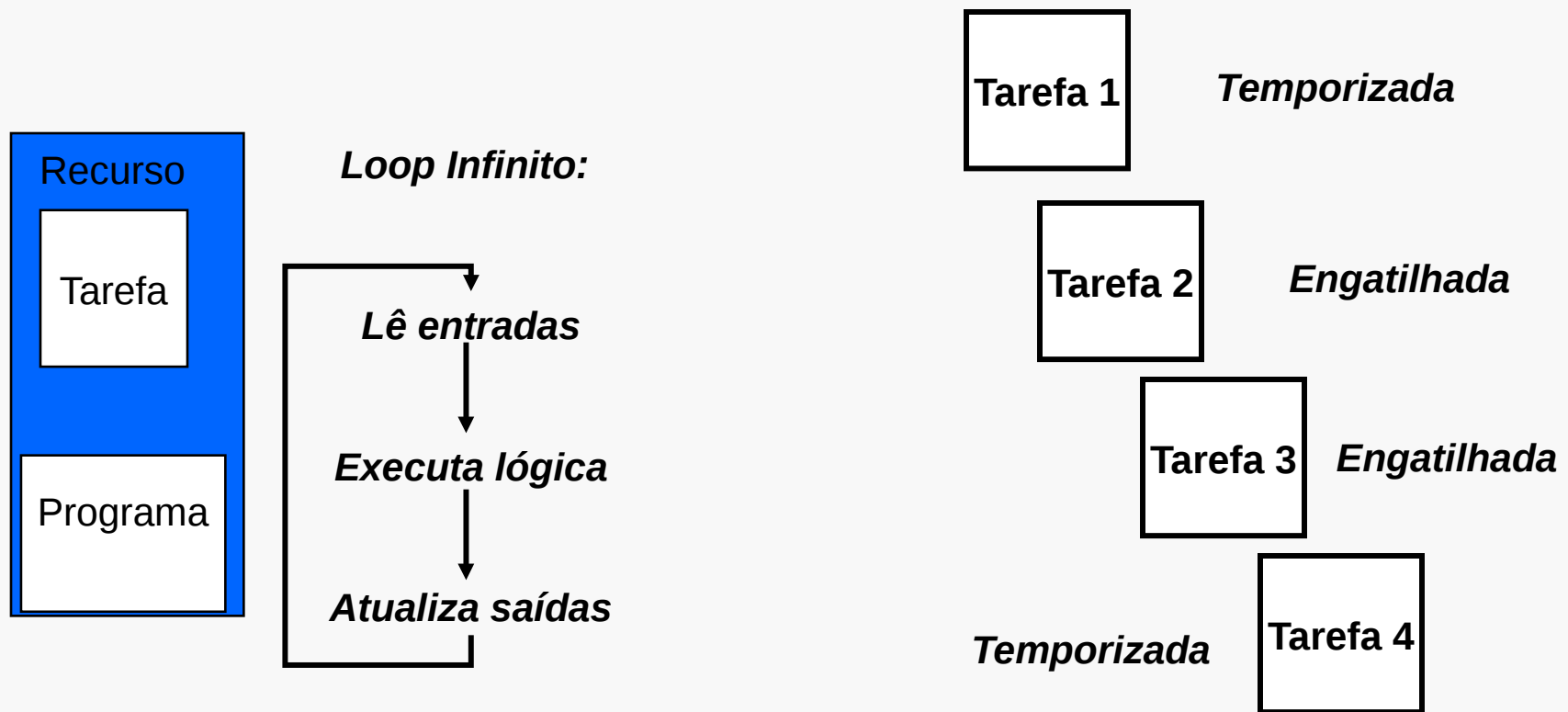
# IEC 61131-3 Modelo de Software



# IEC 61131-3 x PLC convencional



# PLC convencional x IEC 61131-3



---

# **Blocos funcionais (Functions Blocks)**

**O conceito de Blocos funcionais é um dos mais importantes da norma IEC61131-3, para permitir o projeto de software de forma hierárquica e estruturada.**

**Blocos funcionais podem ser utilizados para a criação de elementos de software totalmente reutilizáveis, desde a criação de outros Blocos funcionais mais simples, até Programas mais complexos.**

---

---

# Funções (Functions)

**Funções são elementos de software que não aparecem no modelo de software.**

**Funções não possuem persistência, existindo apenas em tempo de execução, assim como sub-rotinas.**

**Funções podem ter apenas uma saída, sem considerar a saída ENO para controle de execução, ao contrário dos Blocos de Funções que podem ter várias.**

**O resultado pode ser um tipo de dado simples de múltiplos elementos (vetores e estruturas). As funções trigonométricas são os tipos mais comuns de funções.**

---

# Tarefas (TASKS)

---

## Tipos de Tasks :

**Controlado por uma base de tempo:** executado a cada período definido pelo usuário

**Controlado por evento:** executado por um determinado evento.

**Prioridade:** define a ordem em que os *programs* serão executados, começando o de maior prioridade com o menor índice.

---

---

# O que é uma Task?

**Existem 2 tipos diferentes de *Task*:**

- Preemptivas (ou periódicas)
- Não-Preemptivas (não-periódicas)

**Sendo estas divididas em 3 subcategorias: Cíclica, Tempo e Evento.**

---

---

# PREEMPTIVAS

É recomendado para sistemas que devam apresentar comportamento determinístico no tempo. Neste sistema quando o intervalo de uma *Task* de maior prioridade vence, a *Task* em execução sofre preempção (é suspensa) e a nova *Task* de prioridade maior passa a executar imediatamente.

Quando a *Task* de maior prioridade termina, a *Task* suspensa anteriormente volta a executar do ponto onde parou.

---



---

# NÃO-PREEMPTIVAS

Neste tipo de escalonamento uma *Task* sempre completa seu processamento, uma vez iniciado. Após a sua execução, uma *Task* só será escalonada, quando o seu intervalo de execução se esgotar.

Uma *Task* não-preemptiva pode ser interrompida por uma preemptiva. O intervalo entre a execução de *Tasks* pode variar muito neste tipo de escalonamento.

---

---

## **Caminho de acesso (Access Path)**

Os caminhos de acesso permitem a transferência de dados entre diferentes configurações.

Cada configuração pode definir um número de variáveis para acesso por configurações remotas. Estas variáveis podem ser de leitura, escrita ou ambos.

A norma assume que estarão disponíveis mecanismos de comunicação para troca de informações, não abordando a forma à ser adotada.

---

---

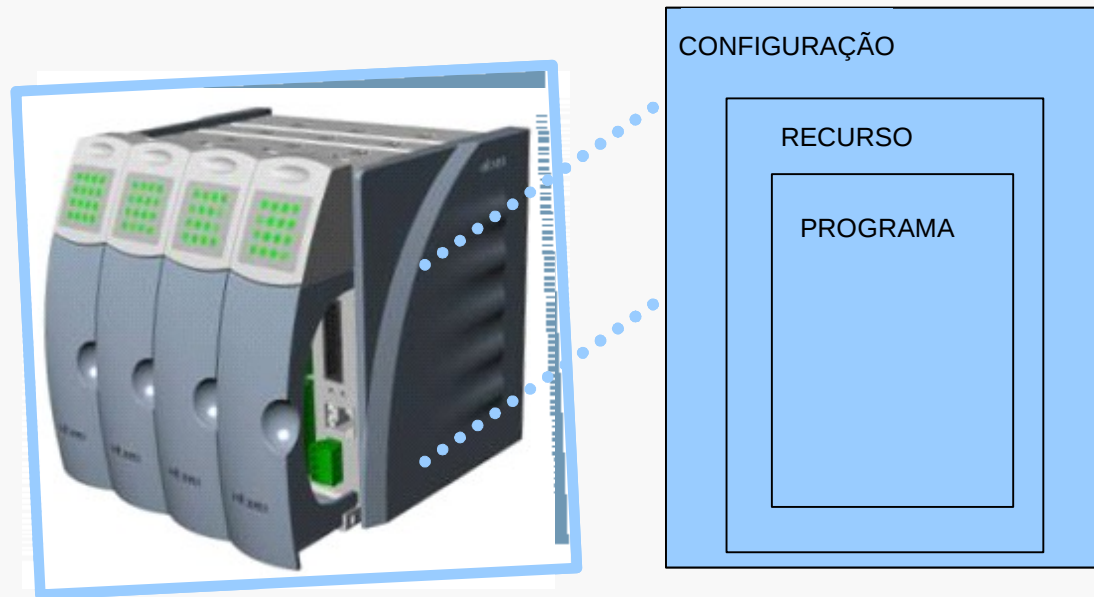
# Fluxo de Controle

**A norma IEC não define os mecanismos para controle de execução dos elementos de software, os quais são dependentes da implementação.**

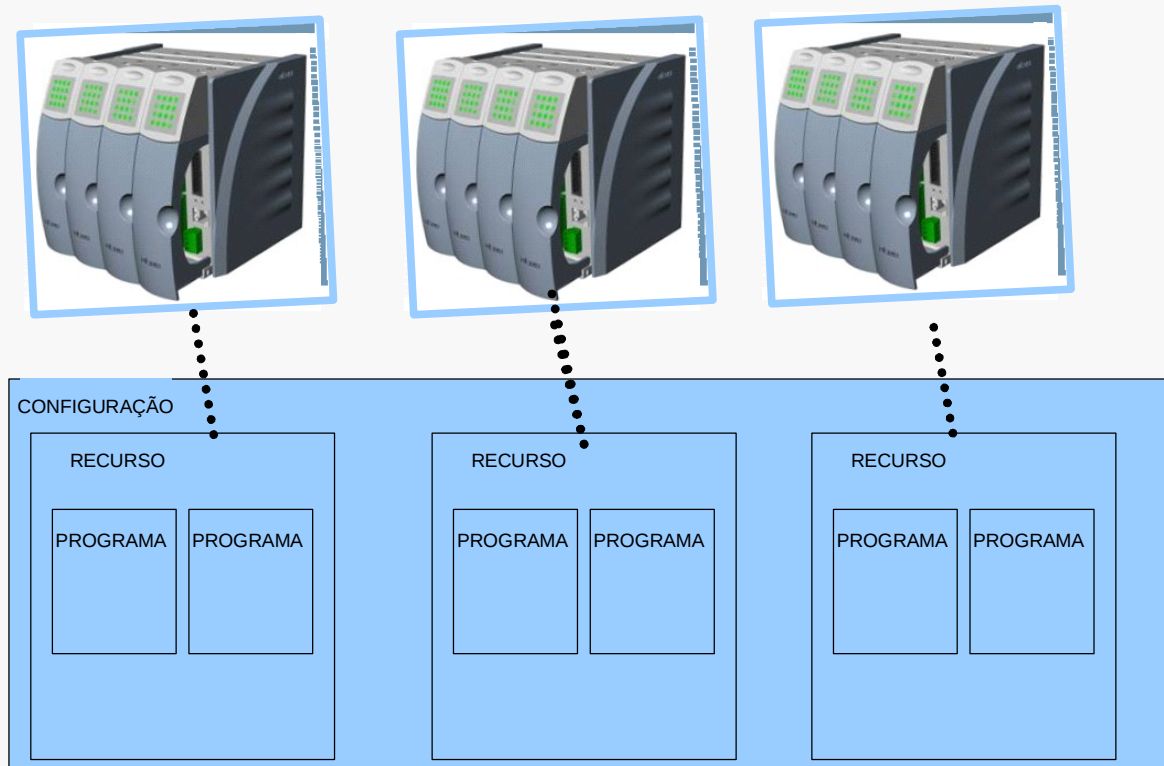
**Entretanto, são definidos os comportamentos na partida e parada do sistema.**

---

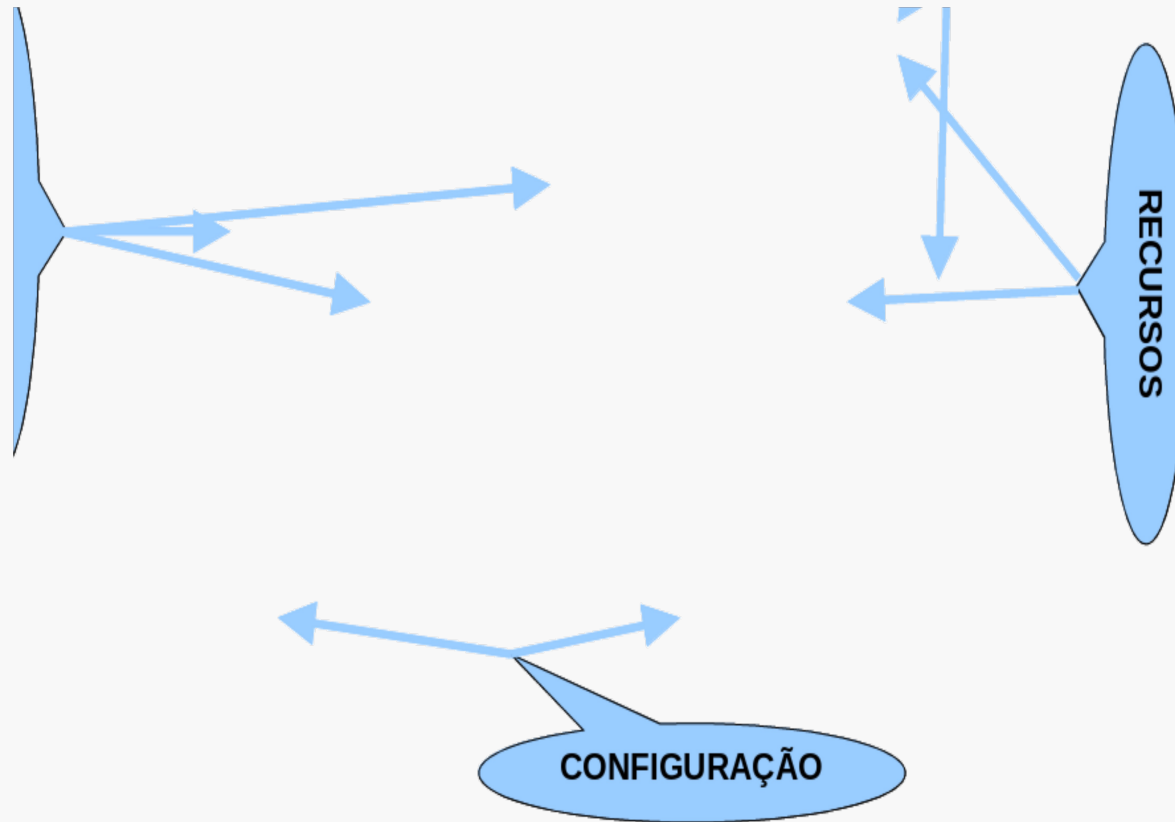
# EXEMPLOS DE SISTEMAS REAIS



# EXEMPLOS DE SISTEMAS REAIS



# EXEMPLOS DE SISTEMAS REAIS



# IEC 61131-3 : Elementos Comuns

## Variáveis e Tipos de Dados

O que é isto?

**01010101 10101010**

**Historicamente**

- **Referência a uma posição física de memória**
- **Referência a uma entrada física**

---

# **IEC 61131-3 : Elementos Comuns Variáveis e Tipos de Dados**

## **Sensor\_Temperatura\_1 : Integer**

- **Representação simbólica**
  - **Área própria para mapeamento de I/O**
  - **Código independente do hardware**
  - **Altamente transparente e compreensível**
  - **Menos erros**
-



# Tipos de Dados (Date types)

TIPO DE DADO	TAMANHO (em memória)	INTERVALO
<b>BOOL</b>	1 bit	TRUE e FALSE
<b>BYTE</b>	8	INFORMAÇÃO BINÁRIA
<b>SINT</b>	8 bits	-127 a +127
<b>INT</b>	16 bits	-32768 a +32767
<b>DINT</b>	32 bits	-2147483648 a +2147483647
<b>LINT</b>	64 bits	$2e+63$ a $(2e+63)-1$
<b>USINT</b>	8 bits	0 a 256
<b>UINT</b>	16 bits	0 a 65535
<b>UDINT</b>	32 bits	0 a 4294967295
<b>ULINT</b>	64 bits	0 a $((2e+64)-1)$
<b>WORD</b>	16 bits	0 a FFFF
<b>DWORD</b>	32 bits	0 a FFFFFFFF
<b>LWORD</b>	64 bits	0 a FFFFFFFFFF

# Tipos de Dados (Date types)

TIPO DE DADO	TAMANHO (em memória)	INTERVALO
<b>REAL</b>	32 bits	-3.40282346638528860e+38 a 3.40282346638528860e+38 Underflow: 1.1754943508222875e-38
<b>LREAL</b>	64 bits	
<b>DATE</b>	32 bits	01/01/2000 a 31/12/2080
<b>TIME</b>	32 bits	0 a 49 <b>d</b> 17 <b>h</b> 2 <b>m</b> 47 <b>s</b> 290 <b>ms</b>
<b>TIME_OF_DAY</b>	32 bits	00:00:00 a 23:59:59
<b>DATE_AND_TIME</b>	32 bits	
<b>STRING</b>	-----	Caracteres ASCII
<b>ARRAY</b>	-----	-----

# Tipos de Dados (Data types)

## Uso de literais para os tipos de dados:

### ◆ Inteiro:

- Decimais: -123            0            +463            23\_123
- Binários: 2#1111\_1111(255 decimal)
- Octal: 8#377 (255 decimal)
- Hexadecimal: 16#FF (255 decimal)

### ◆ Ponto flutuante:

- 10.123            +12\_123.21            -1.65E-10


### ◆ Tempo:

- Declarações: d=dias, h=horas, m=minutos, s=segundos, ms=milisegundos.
  - Forma curta: T#12d3h3s            T#12d3.5h
  - Forma longa: TIME#6d\_5h\_3m\_4s



# VALOR INICIAL DOS DADOS

Endereço	Nome	Tipo de dado	Descrição	Valor Inicial
%MT1		TIME		
%MT2		TIME		
%MT3		TIME		
%MT4		TIME		
%MT5		TIME		



---

# **IEC 61131-3 : Elementos Comuns Variáveis:**

- **Variáveis de escopo global e local**
- **Variáveis de representação direta**

# Variáveis de Representação Direta

Sinal inicial (IEC std)	Identificação de memória	Tamanho do dado		DESCRIÇÃO
<b>%</b>	<b>M</b> (Acesso à memória) <b>I</b> (Entrada física do CLP) <b>Q</b> (Saída física do CLP)	<b>X</b>	(1 bit)	Acesso à variáveis booleanas.
		<b>W</b>	(16 bits)	Acesso à variáveis com 16 bits de tamanho: INT, UINT e WORD.
		<b>D</b>	(32 bits)	Acesso à variáveis com 32 bits de tamanho: DINT, UDINT, DWORD, TIME, DATE e TOD.
		<b>B</b>	(8 bits)	Acesso à variáveis com 8 bits
		<b>L</b>	(64 bits)	Acesso à variáveis com 64 bits

---

## **Vantagens das POU's**

- ◆ **Crie suas próprias bibliotecas de FBs (por tipo de aplicação)**
  - ◆ **FBs são testados e documentados**
  - ◆ **Faça bibliotecas acessíveis em todo o mundo**
  - ◆ **Reutilize o máximo possível**
  - ◆ **Mude da programação para a criação de redes de FBs**
  - ◆ **Economize 40% no próximo projeto**
-



# POU (Program Organization Unit)

Tipo de POU	Aplicado como	Comentário
<b>Program</b>	<b>Instância de um Program</b>	<b>Permite reutilização no nível macro, como programas para reatores, transportadores, caldeiras, etc.</b>
<b>Function Block</b>	<b>Instância de um Function Block</b>	<b>Possibilita a reutilização desde simples a complexas estratégias de controle e algoritmos, como controle PID, filtros, motores, etc.</b>
<b>Function</b>	<b>Função</b>	<b>Usada para tratamento comum de dados, como lógica E, OU, seno, cosseno, soma e etc.</b>

**Como utilizar a IEC 61131-3**

**Estruturação do Desenvolvimento de  
Software com a IEC 61131-3**

---

# Ciclo de Desenvolvimento de Software

**Projeto**

/

**Desenvolvimento**

/

**Instalação**

/

**Manutenção..**

**fases**

---

# **... ciclo de desenvolvimento de software...**

**aperfeiçoamentos.....**

**..... novos requisitos ...**

**.... novas funcionalidades ....**

**.... novas necessidades ...**

**“... a história sem fim do software ”**

---

## Por que Estruturar ?

- ◆ O crescente papel do software no sistema de qualidade: erros custam dinheiro;
- ◆ Os requisitos aumentaram dramaticamente: 100 linhas de código no passado e agora 10.000 linhas;
- ◆ Desenvolvimento de software: não mais o trabalho de uma única pessoa, mas de um time com diferentes conhecimentos e formações;
- ◆ Comissionamento, Instalação, Manutenção e Aperfeiçoamentos são partes essenciais.

---

# Vantagens da Estruturação

- ◆ Melhor visualização
  - ◆ Melhor base para comunicação (interna)
  - ◆ Orientado para a solução de problemas
  - ◆ Base para a reutilização do software
  - ◆ “Auto-documentação”
-

**Como criar um programa de controle  
para isto  
de forma estruturada?**

---

**Como fazer isso com a IEC 61131-3 ?**

**7 Passos para o Sucesso**

---



- 
- **Identificação das interfaces externas do sistema de controle;**
  - **Definição dos principais sinais trocados entre o sistema de controle e o resto do processo/sistemas;**
  - **Definição de todas as interações com o operador , ações de controle e dados de supervisão;**
  - **Análise do problema de controle de cima para baixo quebrando o mesmo em partes lógicas;**
  - **Definição dos blocos funcionais necessários;**
  - **Definição dos tempos de ciclo exigidos pelas diferentes partes da aplicação;**
  - **Configuração do sistema através da definição dos recursos, conexão dos programas e blocos funcionais com as tarefas.**
-

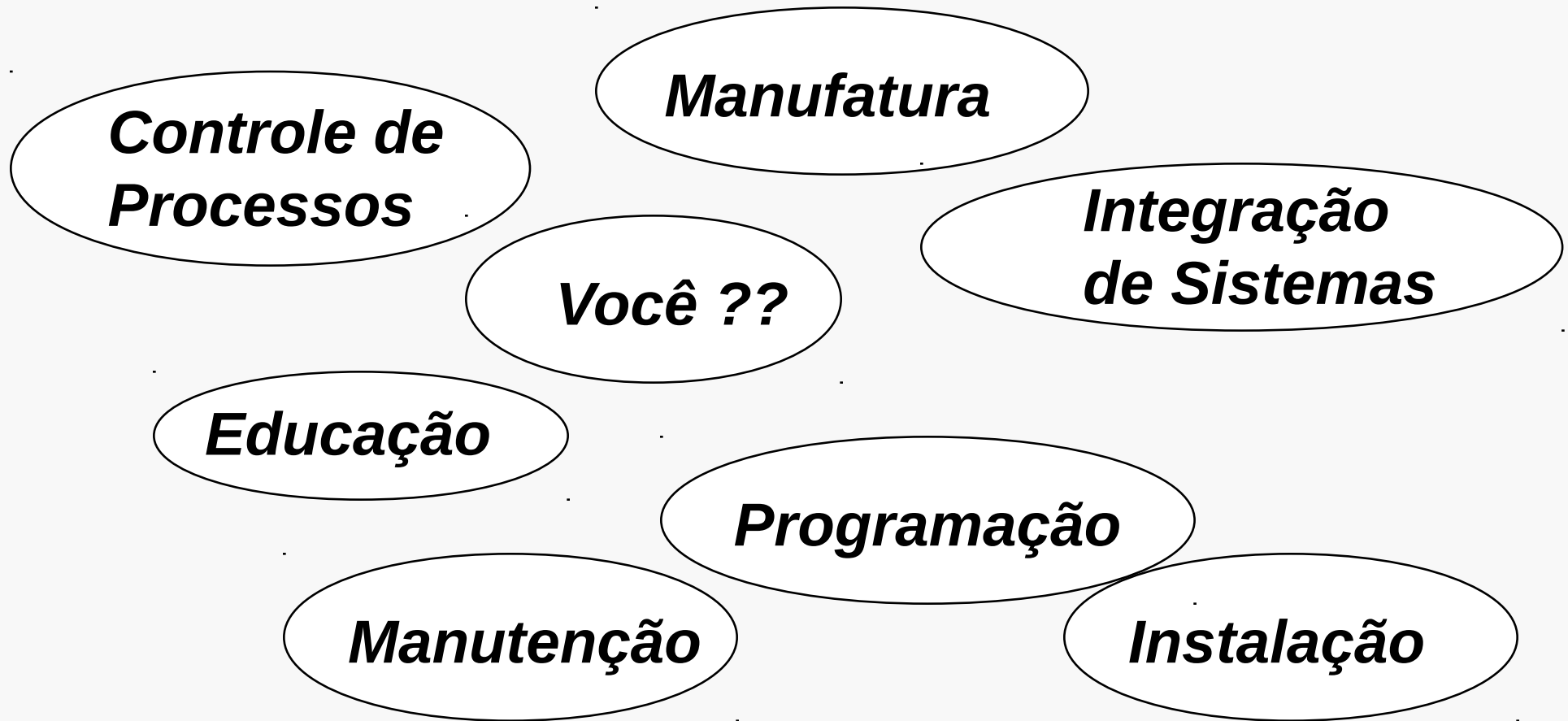
# Conclusão

- ◆ **O processo de desenvolvimento de software mudou:**
  - Mais requisitos..
  - Mais funcionalidades..
  - Mais código..
  - Mais pessoas envolvidas..
  - ... Mais necessidades / expectativas
- ◆ **Estruturação e Decomposição são partes importantes do desenvolvimento de software moderno**
- ◆ **A IEC 61131-3 tem as bases para satisfazer suas necessidades**

**Qual é o Benefício de tal Norma ?**

---

# Usuários? Quais Usuários?



# Usuários? Quais Usuários?

- Linhas de montagem de automóveis
- Plantas de tratamento de água
- Máquinas para processamento e embalagem de alimentos
- Fabricação de cabos
- Automação de salas para produção de semicondutores
- Montanhas russas
- Plantas nucleares

**Esta ampla gama engloba diferentes tipos de conhecimento**

---

# Qual é o benefício de tal Norma ?

- ◆ Reduz o desperdício dos recursos humanos ( em treinamento, depuração, manutenção e consultoria)

# Qual é o benefício de tal Norma ?

- ◆ Reduz o desperdício de recursos humanos ( em treinamento, depuração, manutenção e consultoria)

Orienta a solução de problemas para a reutilização de software (reduz o investimento na aplicação e a dependência dos fornecedores)

# Qual é o benefício de tal Norma ?

- ◆ Reduz o desperdício de recursos humanos ( em treinamento, depuração, manutenção e consultoria)
- ◆ Orienta a solução de problemas para a reutilização de software (reduz o investimento na aplicação e a dependência dos fornecedores)

**Reduz a dificuldade de entendimento e os erros**



# Qual é o benefício de tal Norma ?

- ◆ Reduz o desperdício de recursos humanos ( em treinamento, depuração, manutenção e consultoria)
- ◆ Orienta a solução de problemas para a reutilização de software (reduz o investimento na aplicação e a dependência dos fornecedores)
- ◆ Reduz a dificuldade de entendimentos e os erros

**Técnicas de programação usadas em outros ambientes  
(controle industrial em geral)**

# Qual é o benefício de tal Norma ?

- ◆ Reduz o desperdício de recursos humanos ( em treinamento, depuração, manutenção e consultoria)
- ◆ Orienta a solução de problemas para a reutilização de software (reduz o investimento na aplicação e a dependência dos fornecedores)
- ◆ Reduz a dificuldade de entendimentos e os erros
- ◆ Técnicas de programação usadas em outros ambientes (controle industrial em geral)

**Combina de forma consistente diferentes componentes de diferentes lugares, empresas, países ou projetos**

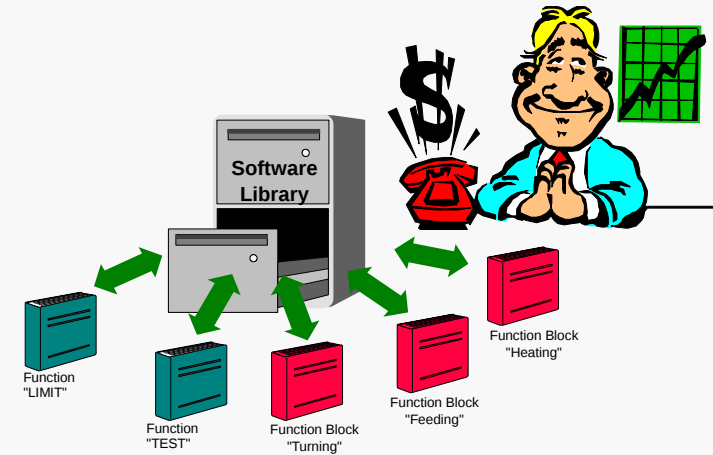
# Qual é o benefício de tal Norma ?

- ◆ Reduz o desperdício de recursos humanos ( em treinamento, depuração, manutenção e consultoria)
- ◆ Orienta a solução de problemas para a reutilização de software (reduz o investimento na aplicação e a dependência dos fornecedores)
- ◆ Reduz a dificuldade de entendimentos e os erros
- ◆ Técnicas de programação usadas em outros ambientes (controle industrial em geral)
- ◆ Combina com harmonia diferentes componentes de diferentes lugares, empresas, países ou projetos

**Amplia a conectividade (proteção do investimento)**

# Qual é o benefício de tal Norma ?

- ◆ Reduz o desperdício de recursos humanos ( em treinamento, depuração, manutenção e consultoria)
- ◆ Orienta a solução de problemas para a reutilização de software (reduz o investimento na aplicação e a dependência dos fornecedores)
- ◆ Reduz a dificuldade de entendimentos e os erros
- ◆ Técnicas de programação usadas em outros ambientes (controle industrial em geral)
- ◆ Combina com harmonia diferentes componentes de diferentes lugares, empresas, países ou projetos



---

## Benefícios trazidos com a adoção da norma IEC61131-3 :

1. Há uma definição sólida dos tipos de dados padrão os “Data types” , que especifica exatamente como o conteúdo de uma variável tem que ser interpretado.

**Vantagem:** Desse modo, para cada tipo de dados, apenas as operações pertinentes são liberadas pelo aplicativo, evitando que o usuário faça operações não permitidas. Por exemplo operações matemáticas para dados tipos numéricos e não strings.

Isto aumenta a segurança com que se programa e também reduz o tempo de programação.

---

---

## Benefícios trazidos com a adoção da norma IEC61131-3 :

3. O programa pode ser subdividido em elementos menores denominados de POU (Program Organization Unit) :  
Programs , Functions e Functions Block

**Vantagem:** O programa ao ser estruturado em sub-tarefas deixando-o mais claro e fácil de entender , evitando os programas tipo macarrão

---

---

## Benefícios trazidos com a adoção da norma IEC61131-3 :

4. Todas os POU's podem conter variáveis locais, isto é os dados, que são reconhecidos somente dentro deste POU. Este princípio de encapsulamento de dados também esta presente nas modernas linguagens de programação.

**Vantagem:** Uma real diminuição do trabalho de programação é possível, porque os programadores não necessitam controlar a posição das variáveis evitando sobreposição das mesmas e tornando o código reaproveitável

---

---

## Benefícios trazidos com a adoção da norma IEC61131-3 :

### 5. Criação de function e functions Blocks de usuário

**Vantagem:** criação de biblioteca de funções de usuário torna-se-a um hábito saudável , sendo utilizadas como “caixas pretas”, importando apenas definir suas variáveis de entrada e saída sem a necessidade de conhecer o funcionamento interno.

---



---

## Benefícios trazidos com a adoção da norma IEC61131-3 :

6. As funções e os blocos da função transformam-se em simbólicos puro, isto é ficam independentes dos endereços.

**Vantagem:.** Functions e Functions Block ficam independentes do hardware utilizado e em muitos casos independentes do fornecedor do hardware sendo portanto 100% reusáveis.

---

---

## **CPU com inovação tecnológica: 400405BF**

**Processador : BF532 da Analog Device**

**Clock interno de 400Mhz e externo de 133Mhz**

Obs: Clock do XA é de 22Mhz

**Performance: 10 vezes mais rápido que o processador XA**

**Primeira CPU Atos com circuito impresso de 6 layers**

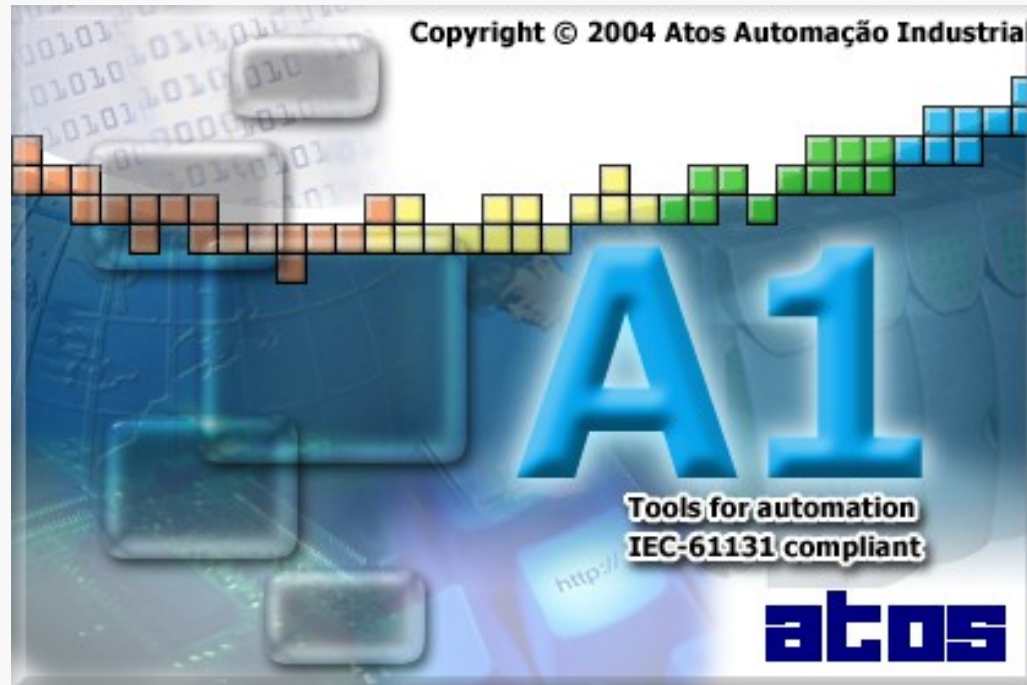
**Sua programação será feita no padrão IEC61131**

---

## Comparação do BF com as CPU's atuais :

Produto	Memória Flash	Memória Firmware	Memória RAM
4004.01	32K	64K	32K
4004.05B	32K	64K/(128K)	64K (128K)
400405R	2x 64K (128K)	64K / (128K)	2 X 64K (128K)
400405T	2x 64K (128K)	64K / (128K)	5 x 64K (512K)
400405BF	2M	16M (SDRAM)	256K

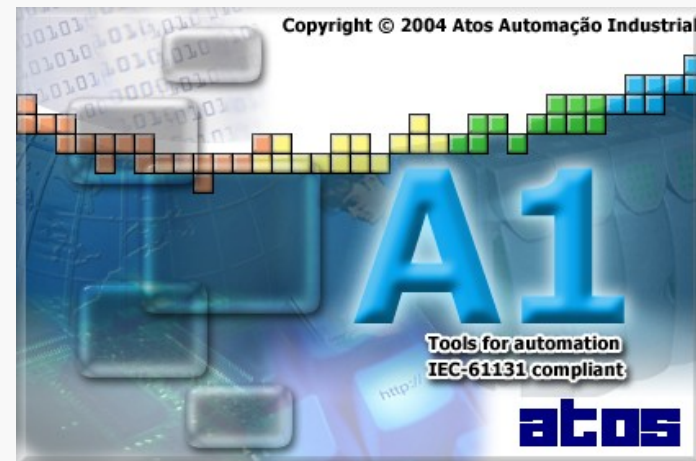
# Software A1



# Informações Gerais

## Configurações mínimas para rodar o A1:

- Processador: Pentium III 500MHz com 256Mb de RAM.
- Vídeo: 800x600 pixels (fontes pequenas)
- Espaço disponível no HD: 60 Mb
- Sistema Operacional: 2000 ou XP



# Como utilizar os módulos na programação?

Os módulos definidos na configuração de hardware são acessados através do mapeamento de memória global de I/O, disponível na janela de variáveis globais do sistema.

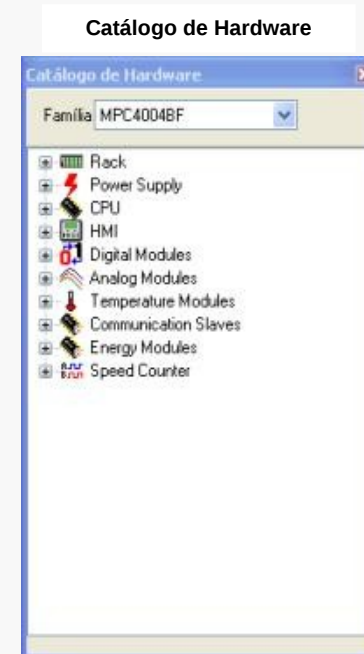
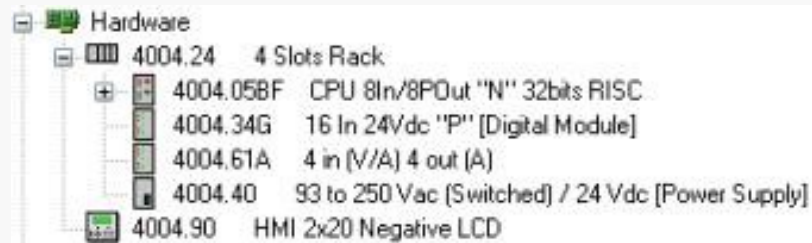


Welcome Page Variáveis Globais		
Grupo	Expansion Board	
I/O	Slot 0 / 4004.058F / CPU 8In/8POut "N" 32bits RISC	
	Slot 8 / 4004.61 / 4 in (V/A) 4 out (V - 0 to +10Vdc)	
	Slot 1 / 4004.54G / 16 In / 16 Out 24Vdc "P" [Digital Module]	
	Slot 0 / 4004.058F / CPU 8In/8POut "N" 32bits RISC	
Endereço		
%I0.0	BOOL	DIGITAL INPUT 0
%I0.1	BOOL	DIGITAL INPUT 1
%I0.2	BOOL	DIGITAL INPUT 2
%I0.3	BOOL	DIGITAL INPUT 3
%I0.4	BOOL	DIGITAL INPUT 4
%I0.5	BOOL	DIGITAL INPUT 5
%I0.6	BOOL	DIGITAL INPUT 6
%I0.7	BOOL	DIGITAL INPUT 7
%Q0.0	BOOL	DIGITAL OUTPUT 0
%Q0.1	BOOL	DIGITAL OUTPUT 1
%Q0.2	BOOL	DIGITAL OUTPUT 2
%Q0.3	BOOL	DIGITAL OUTPUT 3
%Q0.4	BOOL	DIGITAL OUTPUT 4
%Q0.5	BOOL	DIGITAL OUTPUT 5
%Q0.6	BOOL	DIGITAL OUTPUT 6
%Q0.7	BOOL	DIGITAL OUTPUT 7

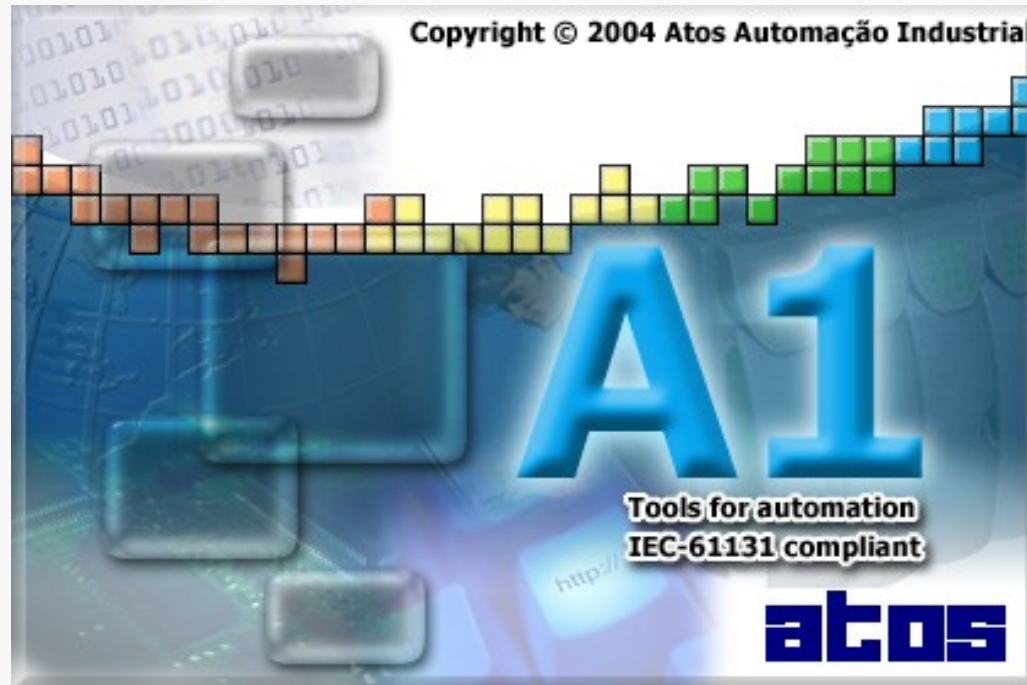
Assim sendo, você apenas escolhe o módulo e qual será o seu mapeamento de memória.

# Configurando o Hardware

O hardware pode ser configurado de forma rápida e prática.



# Software A1





---

# Agenda Software A1

- ◆ **Informações Gerais A1**

- ◆ **IHM**

- ◆ **Prog Editor**

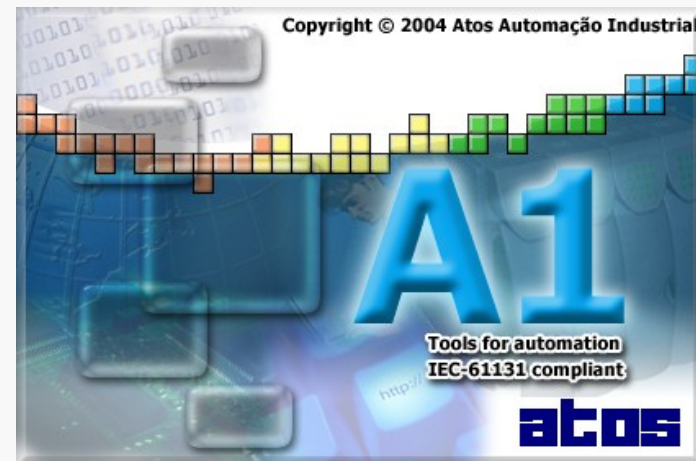
- ◆ **Instruções**

---

# Informações Gerais

## Configurações mínimas para rodar o A1:

- Processador: Pentium III 500MHz com 256Mb de RAM.
- Vídeo: 800x600 pixels (fontes pequenas)
- Espaço disponível no HD: 60 Mb
- Sistema Operacional: 2000 ou XP



# Como utilizar os módulos na programação?

Os módulos definidos na configuração de hardware são acessados através do mapeamento de memória global de I/O, disponível na janela de variáveis globais do sistema.

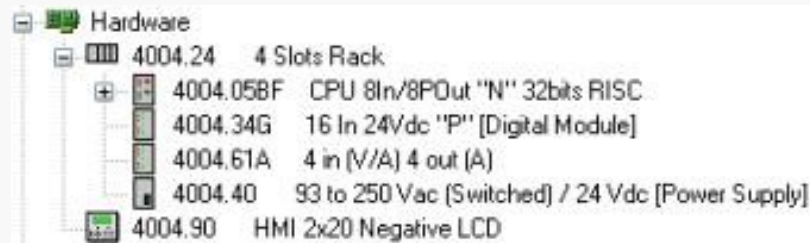


Welcome Page		Variáveis Globais	
Grupo		Expansion Board	
I/O		Slot 0 / 4004.058F / CPU 8In/8POut "N" 32bits RISC	
		Slot 8 / 4004.61 / 4 in (V/A) 4 out (V - 0 to +10Vdc)	
		Slot 1 / 4004.54G / 16 In / 16 Out 24Vdc "P" [Digital Module]	
		Slot 0 / 4004.058F / CPU 8In/8POut "N" 32bits RISC	
Endereço			
%I0.0		BOOL	DIGITAL INPUT 0
%I0.1		BOOL	DIGITAL INPUT 1
%I0.2		BOOL	DIGITAL INPUT 2
%I0.3		BOOL	DIGITAL INPUT 3
%I0.4		BOOL	DIGITAL INPUT 4
%I0.5		BOOL	DIGITAL INPUT 5
%I0.6		BOOL	DIGITAL INPUT 6
%I0.7		BOOL	DIGITAL INPUT 7
%Q0.0		BOOL	DIGITAL OUTPUT 0
%Q0.1		BOOL	DIGITAL OUTPUT 1
%Q0.2		BOOL	DIGITAL OUTPUT 2
%Q0.3		BOOL	DIGITAL OUTPUT 3
%Q0.4		BOOL	DIGITAL OUTPUT 4
%Q0.5		BOOL	DIGITAL OUTPUT 5
%Q0.6		BOOL	DIGITAL OUTPUT 6
%Q0.7		BOOL	DIGITAL OUTPUT 7

Assim sendo, você apenas escolhe o módulo e qual será o seu mapeamento de memória.

# Configurando o Hardware

O hardware pode ser configurado de forma rápida e prática.



Catálogo de Hardware



---

# Configurações do Hardware

## ◆ Jumpers de endereçamento

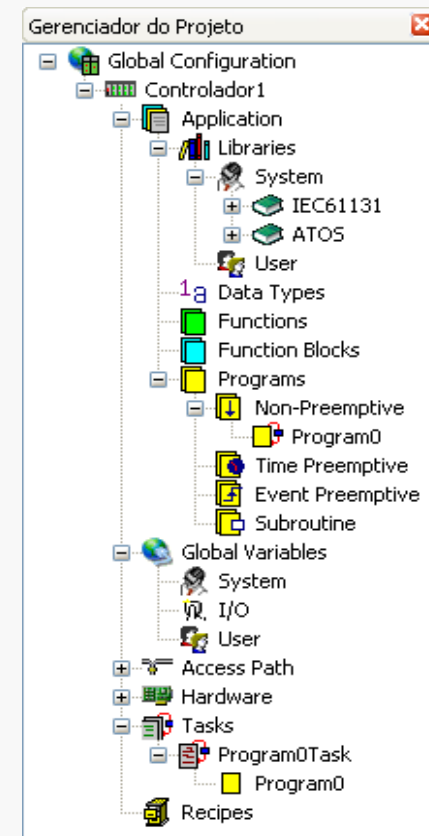
- Placas com jumpers de Grupo podem ser usadas afim de obter um número máximo de pontos em um mesmo CLP.

## ◆ Regras para inclusão de unidades no(s) bastidor(es)

- Os módulos 4004.60 e 4004.61 só podem ser colocadas num total de 08;
  - Os módulos digitais com 8 pontos, como por exemplo 4004.37, 4004.51, 4004.39, etc., só podem ser inseridos no total de 07 módulos;
  - O Contador Rápido e o Multiplex, só podem ser alocados uma única vez;
-

# Gerenciador de Projetos


O gerenciador de projeto mostra através de uma árvore hierárquica de opções, todas as informações relativas ao projeto.



# IHM

## Propriedades da IHM

As propriedades da IHM se apresentam de forma simples e amigável, assim possibilitando uma fácil configuração.

Geral	
Contrast	100 
Linguagem	Portugues
Enable Recipes	false
Code	4004P92
Descrição	- HMI 4x20 LCD

*Propriedades gerais da IHM*

Alarmes	
Primeira variável	
Primeira tela	
Quantidade	0
Tempo ON (ms)	0
Tempo OFF (ms)	0
Timeout	0

*Propriedades de Alarmes*

# Tipos de IHM

## O A1 programa duas séries distintas de IHM:

- LCD 2x20 (IHMs de display LCD e Negative LCD, com 2 linhas de 20 caracteres);
- LCD 4x20 (IHMs de display LCD e LCD Big Digits, com 4 linhas de 20 caracteres).
- O número de telas disponíveis para programação depende apenas do espaço disponível na memória do CLP



Atos Automacao  
Industrial Ltda.

*Display LCD IHM 2x20*



Atos Automacao  
Industrial Ltda.

*Display LCD IHM 4x20*



# Contraste da tela

O software A1 possibilita alterar o contraste da tela numa escala de 0 (menor contraste) a 100 (maior contraste).



Atos Automacao  
Industrial Ltda.



Atos Automacao  
Industrial Ltda.

# Alarmes

A função Alarmes na guia *Propriedades da IHM* permite ao programador alertar, através de mensagens na tela da IHM, a ocorrência de alarmes/alertas ocorridos na máquina ou processo.

Alarmes	
Primeira variável	
Primeira tela	
Quantidade	0
Tempo ON (ms)	0
Tempo OFF (ms)	0
Timeout	0

---

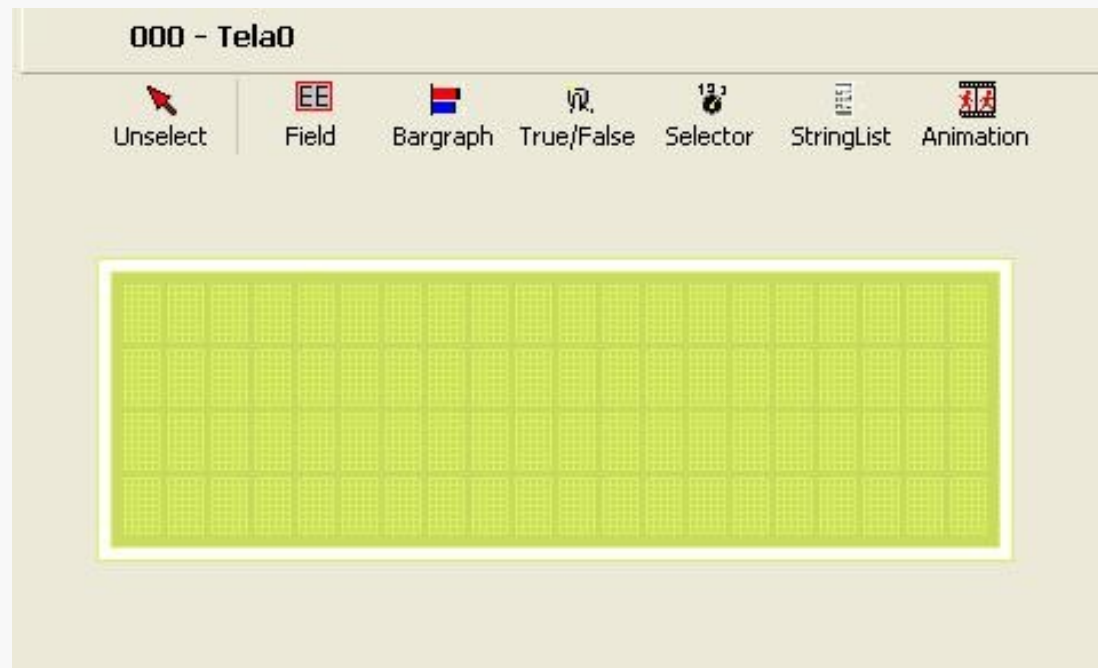
# Senhas

Este item permite a configuração de até oito níveis de senha para edição dos campos nas telas. As senhas podem ser alfa-numéricas, com limite de oito caracteres e letras (maiúsculas) disponíveis no teclado da IHM (A, B, C, D, E e F).

- Segurança Principal: Inserida no campo *Password 1*, dá acesso a todas as telas para edição;
  - Senhas intermediárias: Inseridas nos campos *Password 2* a *7*, são níveis de prioridades;
  - Segurança Menor: Inserida no campo *Password 8*, é o nível mais baixo de prioridade para edição das telas.
-

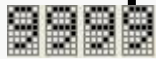

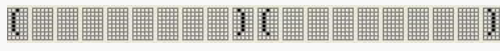




# Inserir Tela

Cada tela possui um campo de propriedades para configuração de características (Propriedades da Tela). Na parte superior da tela (Área de Objetos) são exibidos os objetos (Field, Bargraph, True/False, Selector, StringList e Animation) para serem inseridos.



# Programação das telas

## ◆ Descrição dos Campos

- Field: 
- Bargraph: 
- True/False: 
- Selector: 
- Stringlist: 
- Animation: 
- Texto: 

# Propriedades da Tela

Geral	
Id	0
Nome	Tela0
Security Level	Acesso livre
Tela-alvo S1	
Tela-alvo S2	
Descrição	
Navegação	Início/Fim

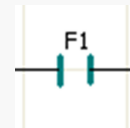
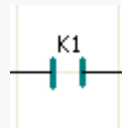
*Menu de propriedades da Tela*

# Teclas K e Teclas F

As Teclas K e as Teclas F podem ser usadas de duas maneiras diferentes:

1. Na lógica do programa *Ladder*;
2. Como chamada de tela.

Na lógica do programa ladder utilizando durante a programação



Como chamada de tela, a guia *K* (Keys) ou a guia *F* (Keys) possibilitam ao programador sua utilização como chamada de tela.

# Auxílio Manutenção

Para ter acesso à tela de auxílio à manutenção é necessário ter uma IHM programada na configuração de hardware.

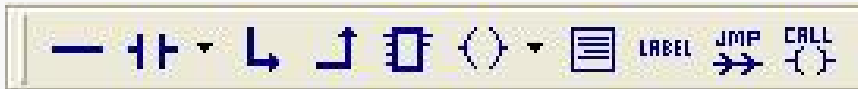
A tela de auxílio à manutenção é acessada pela tecla  da IHM. Para sair, pressione-a novamente

```
[1      ] MANUTENCAO  
[00.00] TRUE
```













# PROG EDITOR

## Barra de ferramentas Ladder

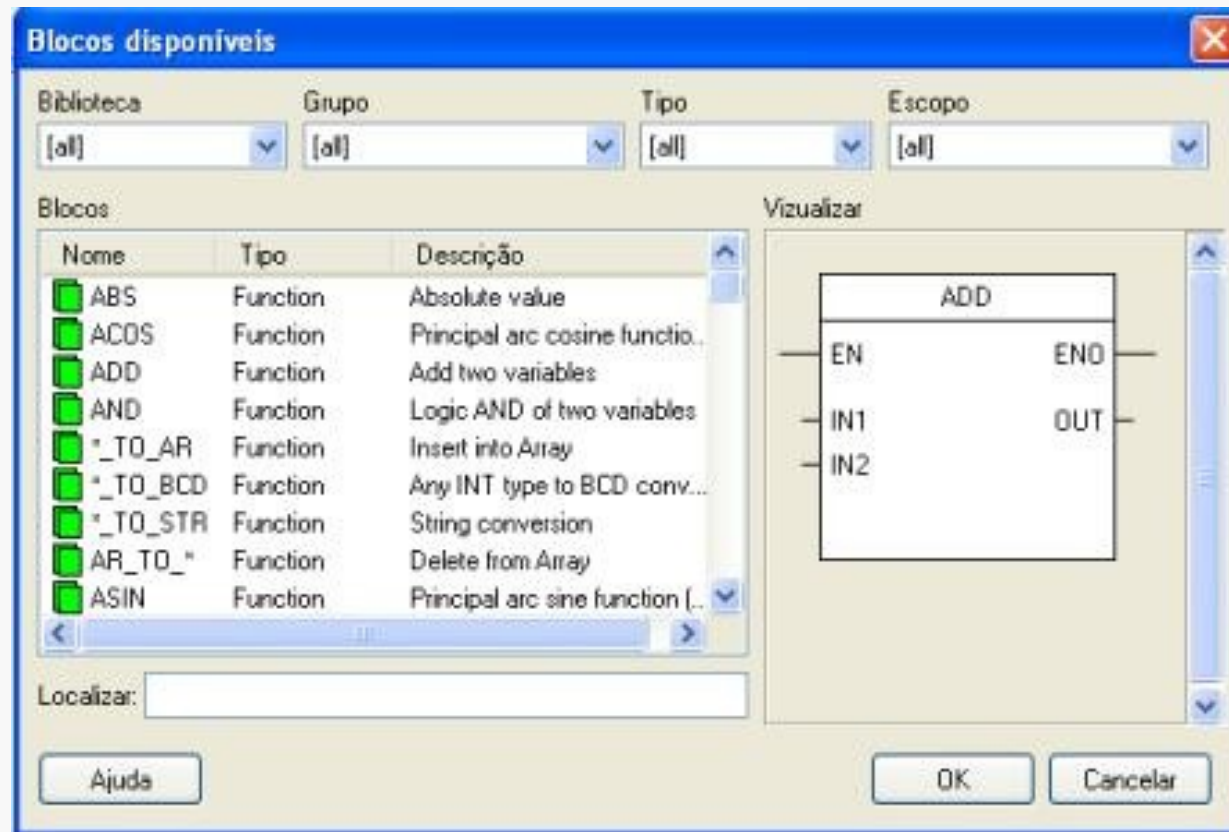


**A barra de ferramentas *ladder* possui todos os componentes para a edição de seu programa.**

# Ferramentas

Botão	Descrição
	Inserir nova linha (padrão: abaixo da linha corrente).
	Inserir contato (padrão: contato aberto).
	Abrir braço paralelo.
	Fechar braço paralelo.
	Inserir bloco de função ( <i>Function</i> ou <i>Function Block</i> ).
	Inserir saída (padrão: saída normal).
	Inserir comentário na linha corrente.
	Inserir <b>LABEL</b> em uma linha.
	Inserir <b>JUMP</b> para um Label.
	Inserir chamada ( <b>CALL</b> ) para uma sub-rotina.

# Inserindo Blocos e Funções



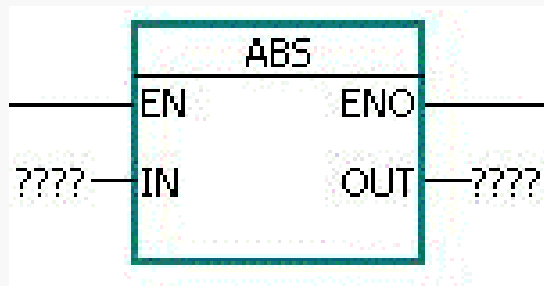
# Instruções

**As instruções são divididas em 15 grupos:**

- Aritméticas
- Array
- Bi-estáveis
- Comparação
- Conversão
- Data/hora
- Detecção de borda
- Logarítmicas
- Lógica entre registros
- Movimentação de dados
- Manipulação de strings
- Seleção
- Temporização e contagem
- Trigonométricas
- Especiais

# Aritméticas

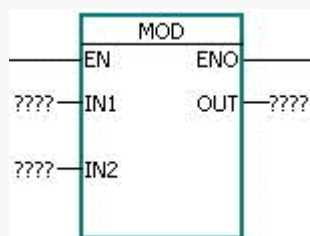
## ABS - Módulo



	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	IN	INT , DINT e REAL	Variável de entrada
	ENO	BOOL	Cópia do valor booleano de EN
	OUT	INT , DINT e REAL	Resultado (mesmo tipo de dado de IN)
SAÍDA			
FLAG	NOME	DESCRIÇÃO	
OV	Overflow	Será ligado se houver estouro da variável de saída ou se a variável de entrada não for válida (NAN por exemplo), caso contrário permanecerá sempre desligado.	
Z	Zero	Será ligado se o resultado for ZERO.	

# Aritméticas

## MOD – Módulo de uma divisão

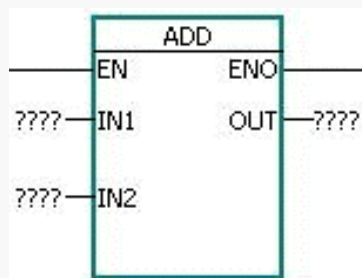


$$OUT = MOD \left( \frac{IN1}{IN2} \right)$$

	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	IN1, IN2	INT , DINT , UINT ,UDINT , WORD , DWORD , e CONSTANTE	Variáveis de entrada (ambas as entradas devem ser do mesmo tipo de dado).
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN
	OUT	INT , DINT , UINT ,UDINT , WORD e DWORD	Resultado (mesmo tipo de dado das entradas).
	FLAG	NOME	DESCRIÇÃO
	OV	Overflow	Será ligado se houver estouro da variável de saída ou divisão por ZERO.
	Z	Zero	Será ligado se o resultado for ZERO.
	N	Sinal	Será ligado se o resultado for NEGATIVO.

# Aritméticas

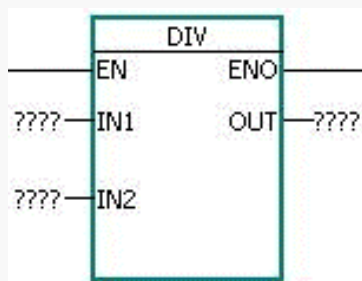
## ADD - Adição



	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	IN1, IN2	INT , DINT , UINT ,UDINT , WORD , DWORD , REAL , TIME , DATE , TOD e CONSTANTE	Variáveis de entrada (ambas as entradas devem ser do mesmo tipo de dado).
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN
	OUT	INT , DINT , UINT ,UDINT , WORD , DWORD , REAL , TIME , DATE , TOD e CONSTANTE	Resultado (mesmo tipo de dado de IN1 e IN2)
	FLAG	NOME	DESCRIÇÃO
	OV	Overflow	Será ligado se houver estouro de variável.
	Z	Zero	Será ligado se o resultado for ZERO.
	N	Sinal	Será ligado se o resultado for NEGATIVO.

# Aritméticas

## DIV - Divisão



NOME	TIPO DE DADO	DESCRIÇÃO
------	--------------	-----------

### ENTRADA

EN

BOOL

Habilita execução da instrução

IN1, IN2

INT , DINT , UINT , UDINT,  
WORD , DWORD ,  
REAL e CONSTANTE

Variáveis de entrada (ambas as entradas devem ser do mesmo tipo de dado).

ENO

BOOL

Cópia do valor booleano de EN

### SAÍDA

OUT

INT , DINT , UINT , UDINT,  
WORD , DWORD ,  
REAL e CONSTANTE

Resultado (mesmo tipo de dado das entradas).

FLAG	NOME	DESCRIÇÃO
------	------	-----------

OV

**Overflow**

Será ligado se houver divisão por ZERO.

N

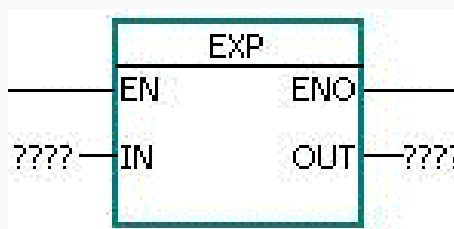
**Sinal**

Será ligado se o resultado for NEGATIVO.



# Aritméticas

## EXP - Exponencial



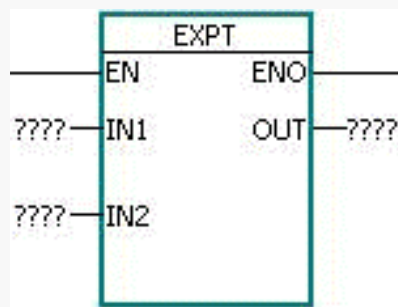
$$OUT = e^{IN}$$

	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	IN	REAL	Variável de entrada
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN
	OUT	REAL	Resultado

FLAG	NOME	DESCRIÇÃO
OV	Overflow	Será ligado se houver estouro da variável de saída ou se a variável de entrada não for válida (NAN por exemplo), caso contrário permanecerá sempre desligado.
Z	Zero	Será ligado se o resultado for ZERO.
N	Sinal	Será ligado se o resultado for NEGATIVO.

# Aritméticas

## EXPT - Potência ( $X^Y$ )



$$OUT = (IN1)^{IN2}$$

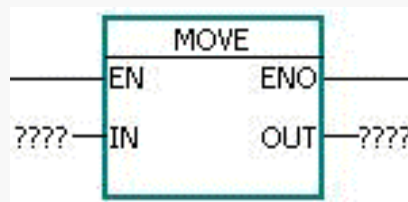
	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	IN1, IN2	REAL	Variáveis de entrada
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN
	OUT	REAL	Resultado

FLAG	NOME	DESCRIÇÃO
OV	Overflow	Será ligado se houver estouro da variável de saída ou se a variável de entrada não for válida (NAN por exemplo), caso contrário permanecerá sempre desligado.
Z	Zero	Será ligado se o resultado for ZERO.
N	Sinal	Será ligado se o resultado for NEGATIVO.

# Aritméticas

## MOVE - Movimentação de dados

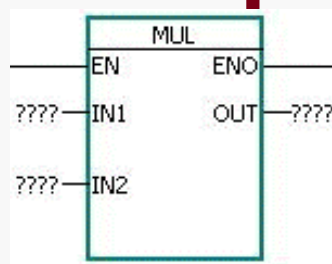


	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	IN	INT, DINT, UINT, UDINT, WORD, DWORD, REAL, TIME, DATE, TOD e CONSTANTE	Variável de origem do dado.
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN
	OUT	INT, DINT, UINT, UDINT, WORD, DWORD, REAL, TIME, DATE e TOD	Variável de destino do dado (mesmo tipo de dado da entrada).

FLAG	NOME	DESCRIÇÃO
-----	-----	Nenhum <i>flag</i> é afetado

# Aritméticas

## MUL - Multiplicação

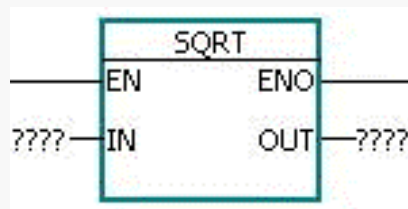


	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	IN1, IN2	INT , DINT , UINT , UDINT , WORD , DWORD , REAL e CONSTANTE	Variáveis de entrada (ambas as entradas devem ser do mesmo tipo de dado).
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN
	OUT	INT , DINT , UINT , UDINT , WORD , DWORD e REAL .	Resultado (mesmo tipo de dado das entradas).

FLAG	NOME	DESCRIÇÃO
OV	Overflow	Será ligado se houver estouro de variável.
Z	Zero	Será ligado se o resultado for ZERO.
N	Sinal	Será ligado se o resultado for NEGATIVO.

# Aritméticas

## SQRT - Raiz Quadrada

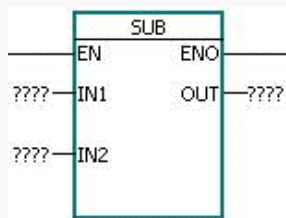


$$OUT = \sqrt{IN}$$

NOME		TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	IN1, IN2	REAL e CONSTANTE.	Variáveis de entrada.
	ENO	BOOL	Cópia do valor booleano de EN
	OUT	REAL	Resultado
FLAG		NOME	DESCRIÇÃO
SAÍDA			
	OV	Overflow	Será ligado se houver estouro da variável de saída ou se a variável de entrada não for válida (NAN por exemplo), caso contrário permanecerá sempre desligado.
	Z	Zero	Será ligado se o resultado for ZERO.

# Aritméticas

## SUB - Subtração

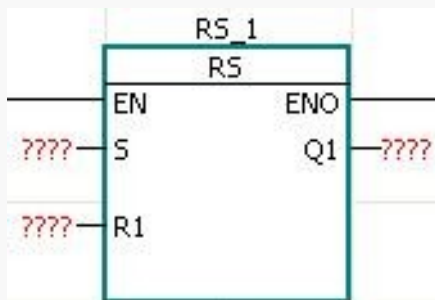


	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	IN1, IN2	INT , DINT, UINT ,UDINT , WORD , DWORD, REAL , TIME , DATE , TOD e CONSTANTE	Variáveis de entrada (ambas as entradas devem ser do mesmo tipo de dado).
	ENO	BOOL	Cópia do valor booleano de EN
SAÍDA	OUT	INT , DINT, UINT ,UDINT , WORD , DWORD, REAL , TIME , DATE e TOD	Resultado (mesmo tipo de dado das entradas).

FLAG	NOME	DESCRIÇÃO
OV	Overflow	Será ligado se houver estouro de variável.
Z	Zero	Será ligado se o resultado for ZERO.
N	Sinal	Será ligado se o resultado for NEGATIVO.

# Bi-estáveis

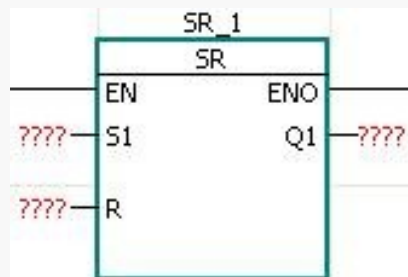
## RS



NOME		TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	S1	BOOL	Entrada Set
	R	BOOL	Reset dominante
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN
	Q1	BOOL	Saída
FLAG	NOME	DESCRIÇÃO	
-----	-----	Nenhum <i>flag</i> é afetado	

# Bi-estáveis

## SR

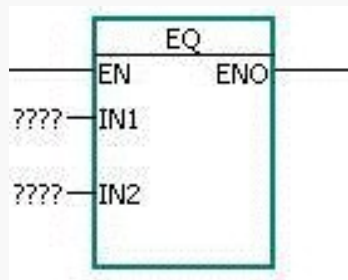


	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	S1	BOOL	Entrada Set dominante
	R	BOOL	Reset
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN
	Q1	BOOL	Saída
FLAG	NOME	DESCRIÇÃO	
----	----	Nenhum <i>flag</i> é afetado	



# Comparação

## EQ - Igual



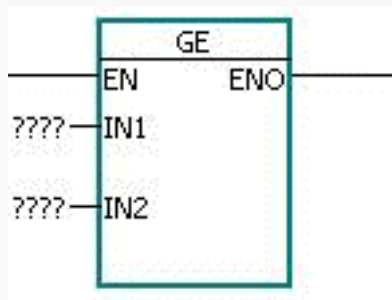
$$ENO = (IN1 = IN2)$$

	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	IN1, IN2	INT , UINT , WORD , DINT , UDINT , DWORD , REAL , STRING , TIME , DATE , TOD e CONSTANTE.	Variáveis de comparação (ambas as entradas devem ser do mesmo tipo de dado).
SAÍDA	ENO	BOOL	Resultado da comparação

FLAG	NOME	DESCRIÇÃO
-----	-----	Nenhum <i>flag</i> é afetado

# Comparação

## GE - Maior ou igual que

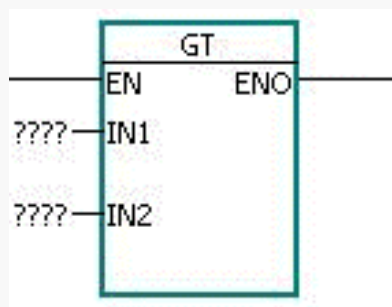


$$ENO = (IN1 \geq IN2)$$

	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	IN1, IN2	INT , UINT , WORD , DINT , UDINT , DWORD , REAL , STRING , TIME , DATE , TOD e CONSTANTE.	Variáveis de comparação (ambas as entradas devem ser do mesmo tipo de dado).
SAÍDA	ENO	BOOL	Resultado da comparação
	FLAG	NOME	DESCRIÇÃO
	-----	-----	Nenhum <i>flag</i> é afetado

# Comparação

## GT - Maior que

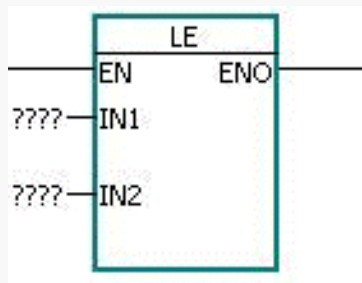


$$ENO = (IN1 > IN2)$$

	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	IN1, IN2	INT , UINT , WORD , DINT , UDINT , DWORD , REAL , STRING , TIME , DATE , TOD e CONSTANTE.	Variáveis de comparação (ambas as entradas devem ser do mesmo tipo de dado).
	ENO	BOOL	Resultado da comparação
SAÍDA			
FLAG	NOME	DESCRIÇÃO	
-----	-----	Nenhum <i>flag</i> é afetado	

# Comparação

## LE - Menor ou igual que



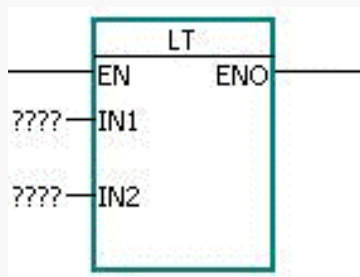
$$ENO = (IN1 \leq IN2)$$

	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	IN1, IN2	INT, UINT, WORD, DINT, UDINT, DWORD, REAL, STRING, TIME, DATE, TOD e CONSTANTE.	Variáveis de comparação (ambas as entradas devem ser do mesmo tipo de dado).
	ENO	BOOL	Resultado da comparação

FLAG	NOME	DESCRIÇÃO
-----	-----	Nenhum flag é afetado

# Comparação

## LT - Menor que

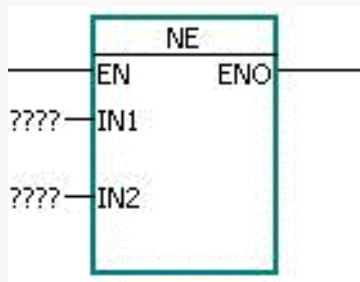


$$ENO = (IN1 < IN2)$$

	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	IN1, IN2	INT, UINT, WORD, DINT, UDINT, DWORD, REAL, STRING, TIME, DATE, TOD e CONSTANTE.	Variáveis de comparação (ambas as entradas devem ser do mesmo tipo de dado).
	ENO	BOOL	Resultado da comparação
SAÍDA			
FLAG	NOME	DESCRIÇÃO	
-----	-----	Nenhum flag é afetado	

# Comparação

## NE - Diferente

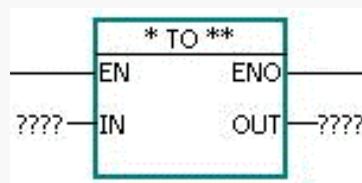


$$ENO = (IN1 \neq IN2)$$

	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	IN1, IN2	INT, UINT, WORD, DINT, UDINT, DWORD, REAL, STRING, TIME, DATE, TOD e CONSTANTE.	Variáveis de comparação (ambas as entradas devem ser do mesmo tipo de dado).
	ENO	BOOL	Resultado da comparação
SAÍDA			
	FLAG	NOME	DESCRIÇÃO
	-----	-----	Nenhum flag é afetado

# Conversão

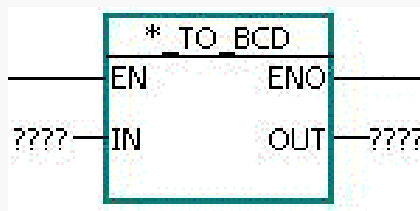
**\* \_TO\_ \*\***



	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	IN	INT , DINT, UINT , UDINT, WORD e DWORD , REAL , TIME, DATE e TOD .	Valor a ser convertido
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN
	OUT	INT, DINT, UINT, UDINT, WORD, DWORD, REAL, TIME, DATE e TOD .	Valor convertido
	FLAG	NOME	DESCRIÇÃO
	OV	Overflow	Será ligado se houver estouro de variáve e conversões para tipos de dados sem sinal ligam este flag se o valor de <b>IN</b> for negativo. nesta situação, a saída <b>OUT</b> não terá seu conteúdo modificado mantendo o último valor existente.

# Conversão

## \*\_TO\_BCD - Converte para BCD



	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	IN	UINT, UDINT, INT e DINT	Valor a ser convertido p/ BCD
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN
	OUT	BCD (WORD ou DWORD )	Valor convertido em BCD (deve ser armazenado em uma variável do tipo de dado WORD ou DWORD )

FLAG	NOME	DESCRIÇÃO
OV	Overflow	Será ligado se o valor de IN for maior que 9999 (WORD ) ou 99999999 (DWORD ).
N	Sinal	Se o valor de IN for negativo, será colocado em OUT o módulo do valor convertido e este flag será ligado.



# Conversão

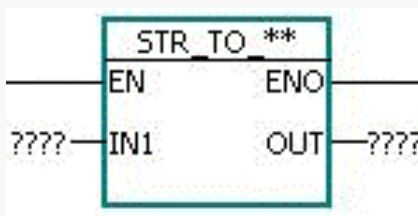
## BCD\_TO\_\*\* - Converte valor BCD para tipo UINT/UDINT



	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	IN	BCD (WORD ou DWORD )	Valor em BCD a ser convertido (deve estar armazenado em uma variável de tipo de dado WORD ou DWORD )
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN
	OUT	UINT , UDINT , INT e DINT	Valor convertido para o tipo de dado escolhido (UINT ou UDINT ).
FLAG	NOME	DESCRIÇÃO	
-----	-----	Nenhum <i>flag</i> é afetado.	

# Conversão

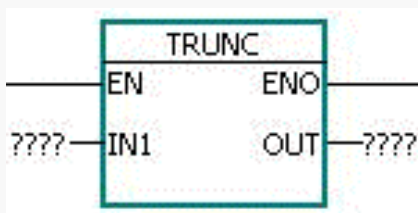
## STR\_TO\_\*\* - Converte STRING para UINT



	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	IN	STRING	String a ser convertida em um valor numérico
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN
	OUT	INT e DINT	String convertida para o tipo de dado escolhido (INT ou DINT ).
FLAG	NOME	DESCRIÇÃO	
-----	-----	Nenhum <i>flag</i> é afetado	

# Conversão

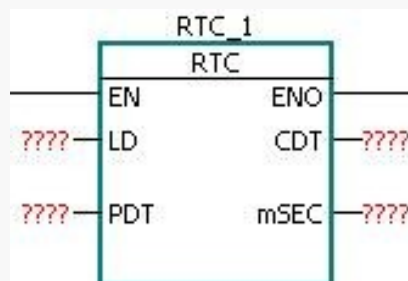
## TRUNC - Truncamento



	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução.
	IN	REAL	Valor a ser arredondado.
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN.
	OUT	INT , DINT , UINT e UDINT	Valor arredondado.
	FLAG	NOME	DESCRIÇÃO
	OV	Overflow	Será ligado se houver estouro de variável.
	N	Sinal	Se o valor de IN for negativo, será colocado em OUT o módulo do valor convertido para os tipos sem sinal e este flag será ligado.

# Data/Hora

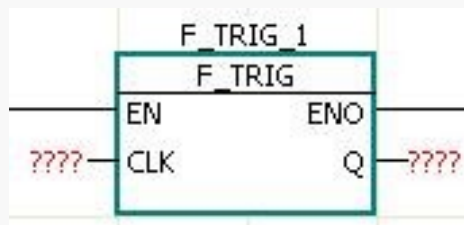
## RTC\_ATOS - Real Time Clock



	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	LD	BOOL	Quando habilitado, salva IN no RTC. Se desabilitado, mostra data e hora do sistema em CDT e mSEC.
	PDT	TOD e DATE	Valor de entrada para ajuste do RTC
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN
	CDT	TOD e DATE	Carrega data/hora atual do sistema
	mSEC	UINT	Carrega milissegundos do sistema
FLAG	NOME	DESCRIÇÃO	
-----	-----	Nenhum flag é afetado	

# Detecção de Borda

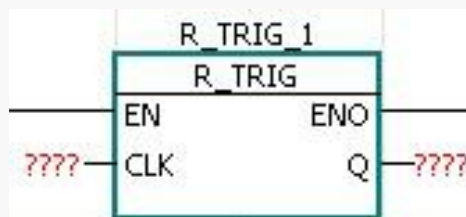
## F\_TRIG



	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	CLK	BOOL	Variável de entrada
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN
	Q	BOOL	Saída
	FLAG	NOME	DESCRIÇÃO
	----	----	Nenhum <i>flag</i> é afetado.

# Detecção de Borda

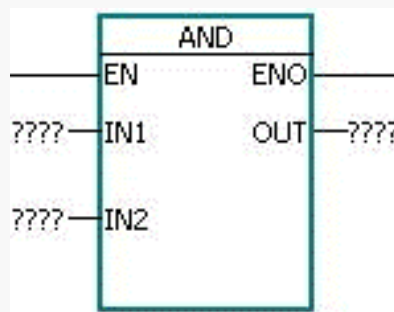
## R\_TRIG



	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	CLK	BOOL	Variável de entrada
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN
	Q	BOOL	Saída
FLAG	NOME	DESCRIÇÃO	
-----	-----	Nenhum <i>flag</i> é afetado	

# Lógica entre Registros

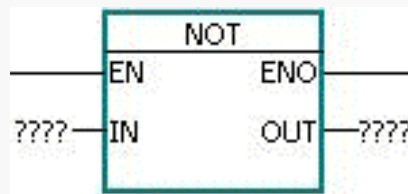
## AND



	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	IN1, IN2	WORD , DWORD e CONSTANTE	Variáveis de entrada (ambas as entradas devem ser do mesmo tipo de dado).
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN
	OUT	WORD e DWORD	Resultado (mesmo tipo de dado das entradas).
FLAG	NOME	DESCRIÇÃO	
----	----	Nenhum <i>flag</i> é afetado	

# Lógica entre Registros

## NOT

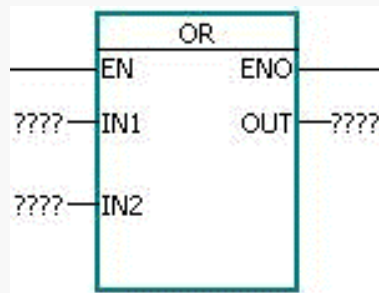


	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	IN	WORD , DWORD e CONSTANTE	Variáveis de entrada (ambas as entradas devem ser do mesmo tipo de dado).
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN
	OUT	WORD e DWORD	Resultado (mesmo tipo de dado das entradas).
FLAG	NOME	DESCRIÇÃO	
----	----	Nenhum <i>flag</i> é afetado	



# Lógica entre Registros

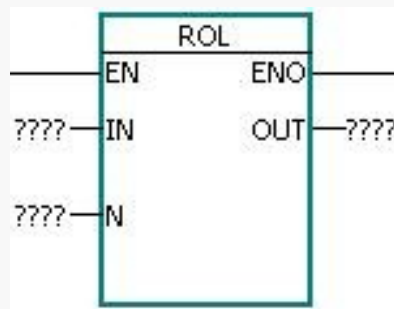
## OR



	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	IN1, IN2	WORD , DWORD e CONSTANTE	Variáveis de entrada (ambas as entradas devem ser do mesmo tipo de dado).
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN
	OUT	WORD e DWORD	Resultado (mesmo tipo de dado das entradas).
FLAG	NOME	DESCRIÇÃO	
-----	-----	Nenhum <i>flag</i> é afetado	

# Movimentação de Dados

## ROL - Rotação à esquerda



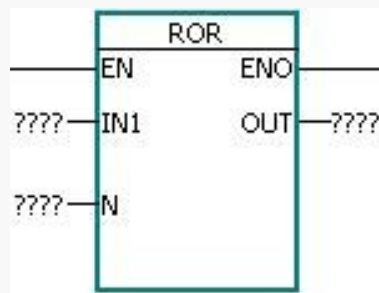
NOME	TIPO DE DADO	DESCRIÇÃO
------	--------------	-----------

ENTRADA	EN	BOOL	Habilita execução da instrução
	IN	WORD , DWORD e CONSTANTE	Variável com conteúdo a ser rotacionado.
	N	UINT e CONSTANTE	Número de rotações à esquerda dos bits da variável definida em IN.
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN
	OUT	WORD e DWORD	Valor rotacionado de N vezes (mesmo tipo de dado da entrada IN).

FLAG	NOME	DESCRIÇÃO
----	----	Nenhum flag é afetado

# Movimentação de Dados

## ROR - Rotação à direita

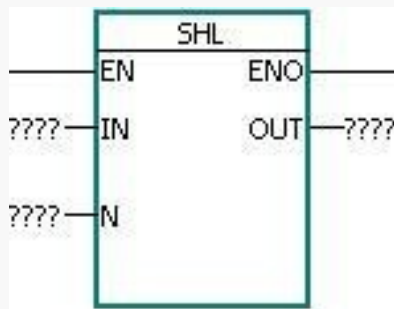


	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	IN	WORD , DWORD e CONSTANTE	Variável com conteúdo a ser rotacionado.
	N	UINT e CONSTANTE	Número de rotações à direita dos bits da variável definida em IN.
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN
	OUT	WORD e DWORD	Valor rotacionado de N vezes (mesmo tipo de dado da entrada IN).

FLAG	NOME	DESCRIÇÃO
-----	-----	Nenhum <i>flag</i> é afetado

# Movimentação de Dados

## SHL - Deslocamento à esquerda

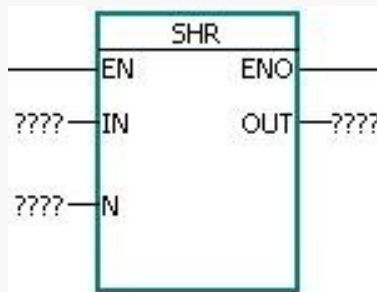


	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	IN	WORD , DWORD e CONSTANTE	Variável com conteúdo a ser deslocado.
	N	UINT e CONSTANTE	Número de deslocamentos à esquerda dos bits da variável definida em IN.
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN
	OUT	WORD e DWORD	Valor deslocado de N vezes (mesmo tipo de dado da entrada IN).

FLAG	NOME	DESCRIÇÃO
-----	-----	Nenhum <i>flag</i> é afetado

# Movimentação de Dados

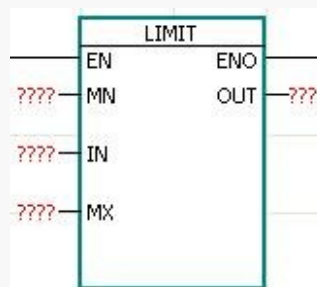
## SHR - Deslocamento à direita



	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	IN	WORD , DWORD e CONSTANTE	Variável com conteúdo a ser deslocado.
	N	UINT e CONSTANTE	Número de deslocamentos à direita dos bits da variável definida em IN.
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN
	OUT	WORD e DWORD	Valor deslocado de N vezes (mesmo tipo de dado da entrada IN).
	FLAG	NOME	DESCRIÇÃO
	-----	-----	Nenhum <i>flag</i> é afetado

# Seleção

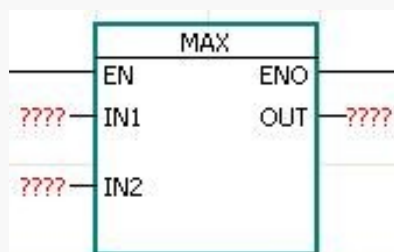
## LIMIT



NOME		TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução.
	MN	INT , UINT , DINT , UDINT , WORD , DWORD , REAL , TIME , DATE , TOD e DATE_AND_TIME	Valor mínimo.
	In		Entrada.
	MX		Valor máximo.
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN.
	OUT	INT , UINT , DINT , UDINT , WORD , DWORD , REAL , TIME , DATE , TOD e DAT	Saída.
FLAG		NOME	DESCRIÇÃO
-----		-----	Nenhum <i>flag</i> é afetado

# Seleção

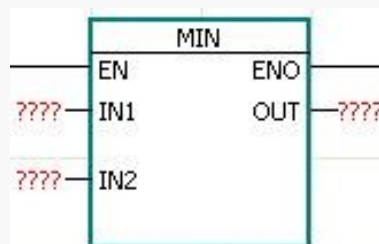
## MAX



	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	In1	INT , UINT , DINT , UDINT , WORD , DWORD , REAL , TIME , DATE , TOD e DATE_AND_TIME	Entrada 1
	In2		Entrada 2
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN
	OUT	INT , UINT , DINT , UDINT , WORD , DWORD , REAL , TIME , DATE , TOD e DATE_AND_TIME	Saída copiada depois da comparação entre o maior valor de In1 e In2.
	FLAG	NOME	DESCRIÇÃO
	-----	-----	Nenhum <i>flag</i> é afetado

# Seleção

## MIN

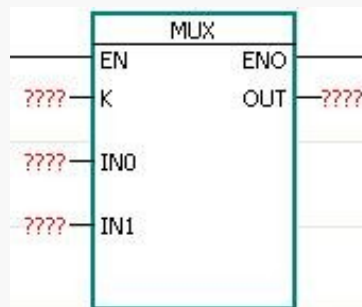


	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	In1	INT , UINT , DINT , UDINT , WORD , DWORD , REAL , TIME , DATE , TOD e DAT	Entrada 1
	In2		Entrada 2
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN
	OUT	INT , UINT , DINT , UDINT , WORD , DWORD , REAL , TIME , DATE , TOD e DAT E_AND_TIME	Saída copiada depois da comparação entre o menor valor de In1 e In2.
	FLAG	NOME	DESCRIÇÃO
	-----	-----	Nenhum flag é afetado



# Seleção

## MUX

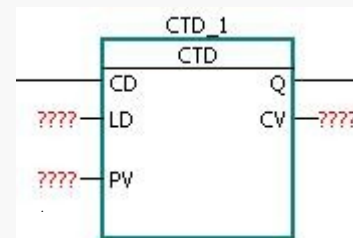
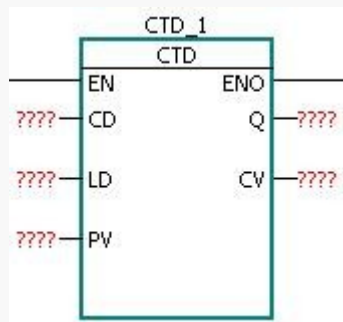


	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução.
	K	UINT	Seleciona entrada a ser copiada.
	In1	INT , UINT , DINT , UDINT , WORD , DWORD , REAL , TIME , DATE , TOD e DAT	Entrada 1.
	In2		Entrada 2.
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN.
	OUT	INT , UINT , DINT , UDINT , WORD , DWORD , REAL , TIME , DATE , TOD e DATE_AND_TIME	Saída copiada depois de setado número da entrada em K.
FLAG	NOME	DESCRIÇÃO	
-----	-----	Nenhum <i>flag</i> é afetado	



# Temporização e Contagem

## CTD - Contador Decrescente

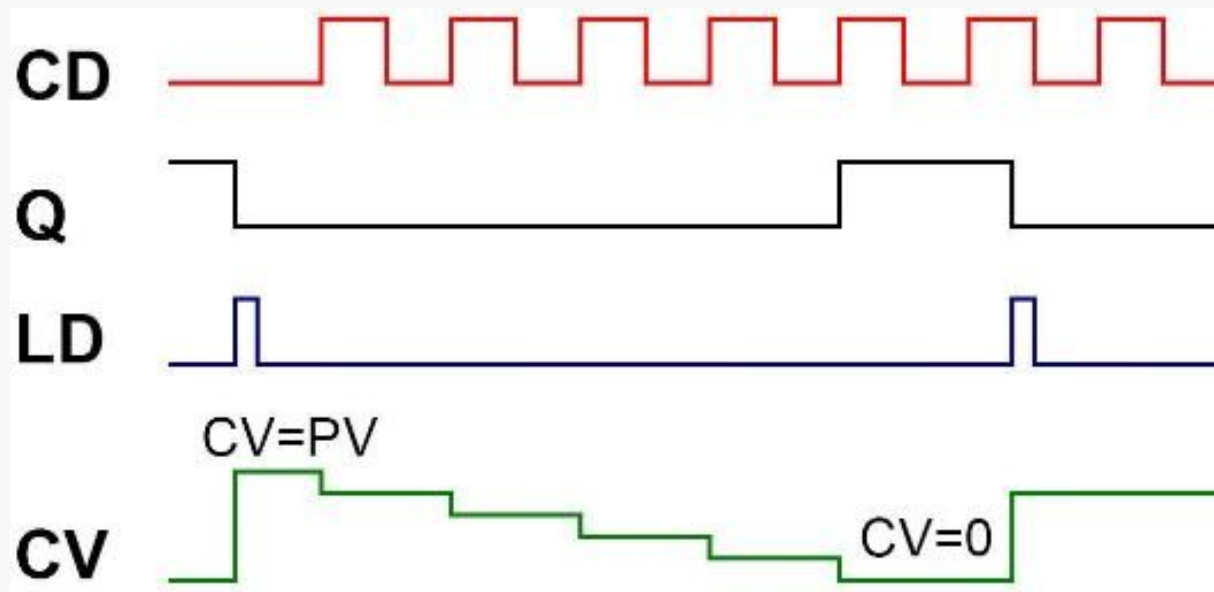


	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução. Entrada <b>opcional</b> . Existente somente no modo <u>com</u> EN/ENO.
	CD	BOOL	Sinal de contagem (pulso).
	LD	BOOL	Carrega Preset (PV) em CV (efetivo).
	PV	INT , UINT , DINT e UDINT	Preset do contador.
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN. Saída <b>opcional</b> . Existente somente no modo <u>com</u> EN/ENO.
	Q	BOOL	Saída do contador.
	CV	INT , UINT , DINT e UDINT	Efetivo do contador.
	FLAG	NOME	DESCRIÇÃO
	----	----	Nenhum <i>flag</i> é afetado

# Temporização e Contagem

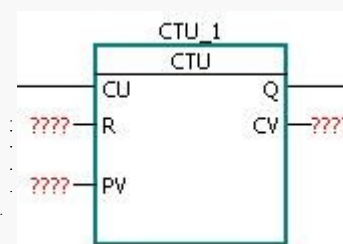
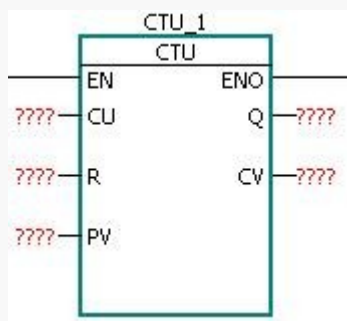
## CTD - Contador Decrescente

*Exemplo gráfico de funcionamento:*



# Temporização e Contagem

## CTU - Contador Crescente

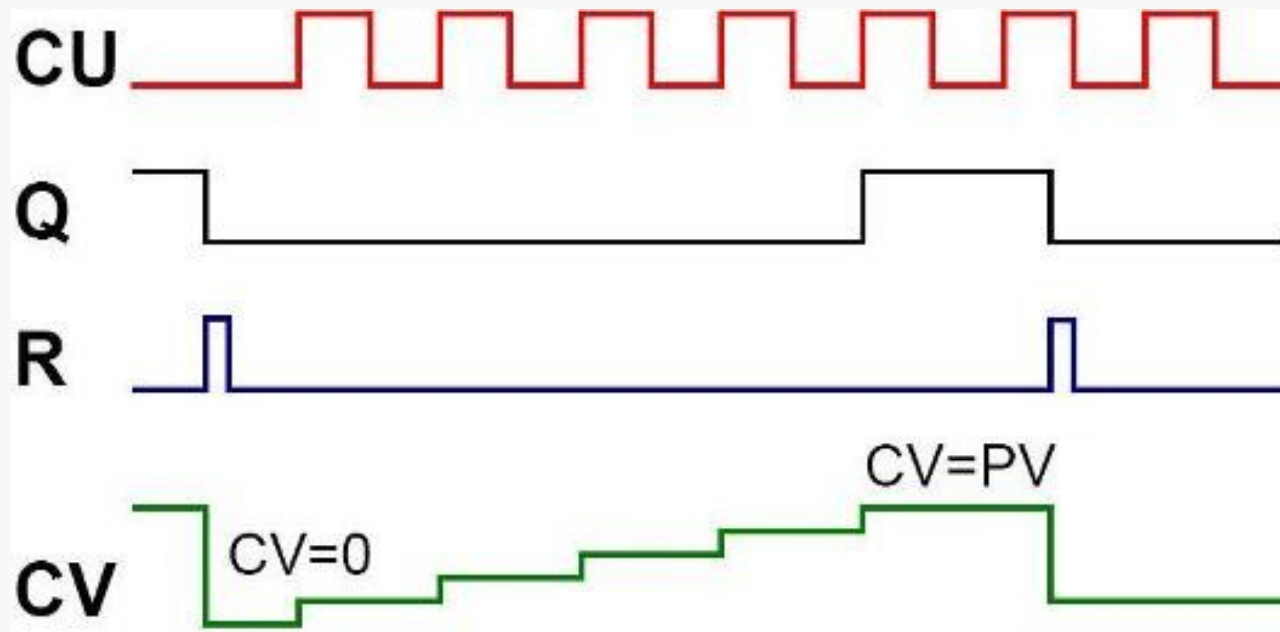


	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução. Entrada <b>opcional</b> . Existente somente no modo <u>com</u> EN/ENO.
	CU	BOOL	Sinal de contagem (pulso).
	R	BOOL	Reset de contagem (CV = 0).
	PV	INT , UINT , DINT e UDINT	Preset de contagem.
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN. Saída <b>opcional</b> . Existente somente no modo <u>com</u> EN/ENO.
	Q	BOOL	Saída do contador.
	CV	INT , UINT , DINT e UDINT	Efetivo de contagem.
	FLAG	NOME	DESCRIÇÃO
	----	----	Nenhum <i>flag</i> é afetado

# Temporização e Contagem

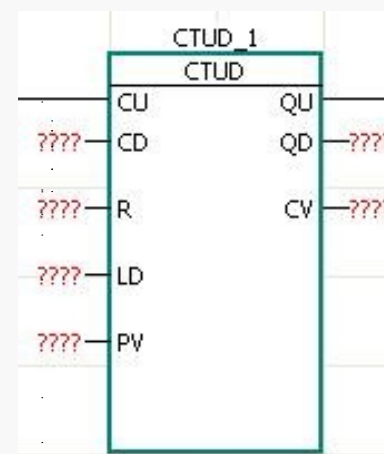
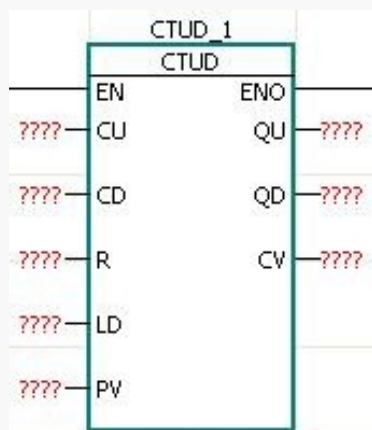
## CTU - Contador Crescente

*Exemplo gráfico de funcionamento:*



# Temporização e Contagem

## CTUD - Contador Crescente/Decrescente



	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução. Entrada <b>opcional</b> . Existente somente no modo <u>com</u> EN/ENO.
	CU	BOOL	Sinal de contagem (pulso).
	CD		Reset de contagem (CV = 0).
	R		Carrega Preset (PV) em CV (efetivo).
	LD		Preset do contador.
	PV	INT , UINT , DINT e UDINT	

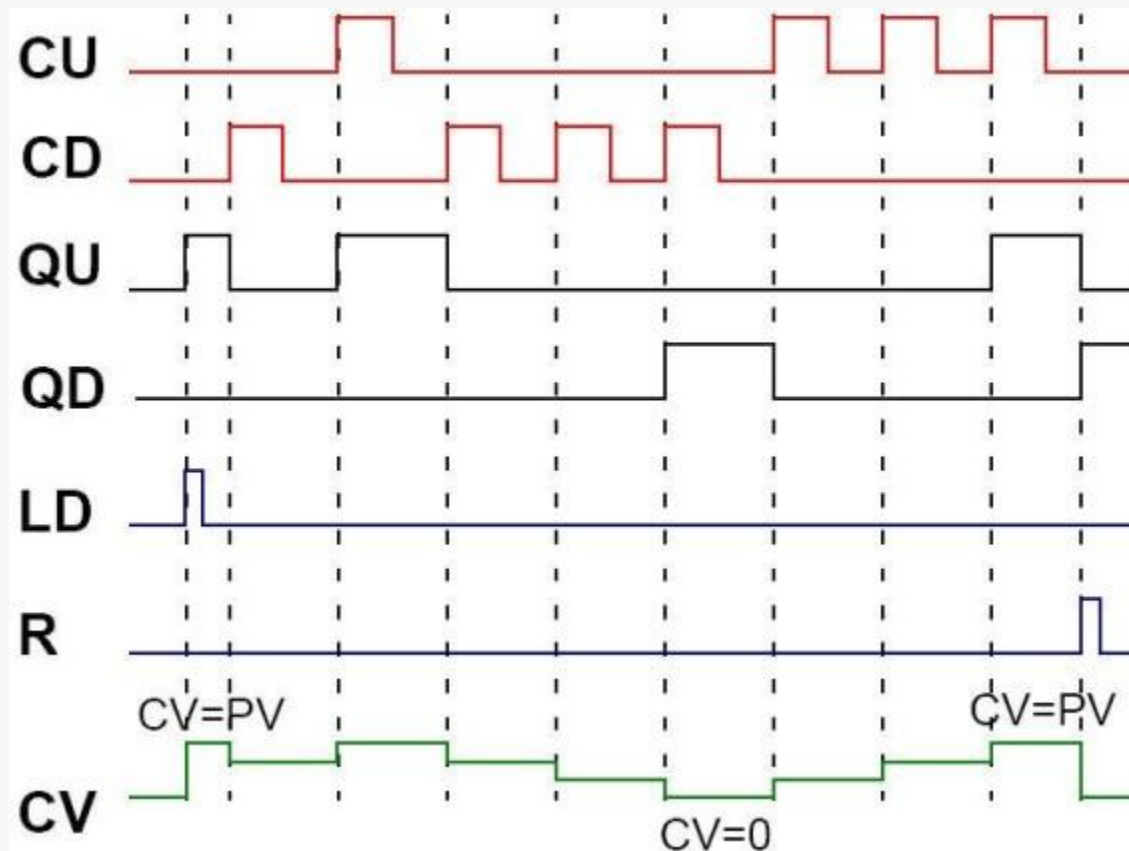
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN. Saída <b>opcional</b> . Existente somente no modo <u>com</u> EN/ENO.
	QU	BOOL	Saída do contador crescente.
	QD		Saída do contador decrescente.
	CV	INT , UINT , DINT e UDINT	Efetivo do contador.
	FLAG	NOME	DESCRIÇÃO
	-----	-----	Nenhum <i>flag</i> é afetado



# Temporização e Contagem

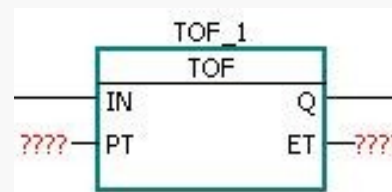
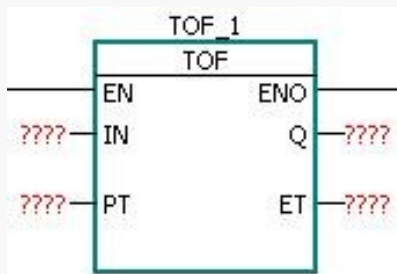
## CTUD - Contador Crescente/Decrescente

*Exemplo gráfico de funcionamento:*



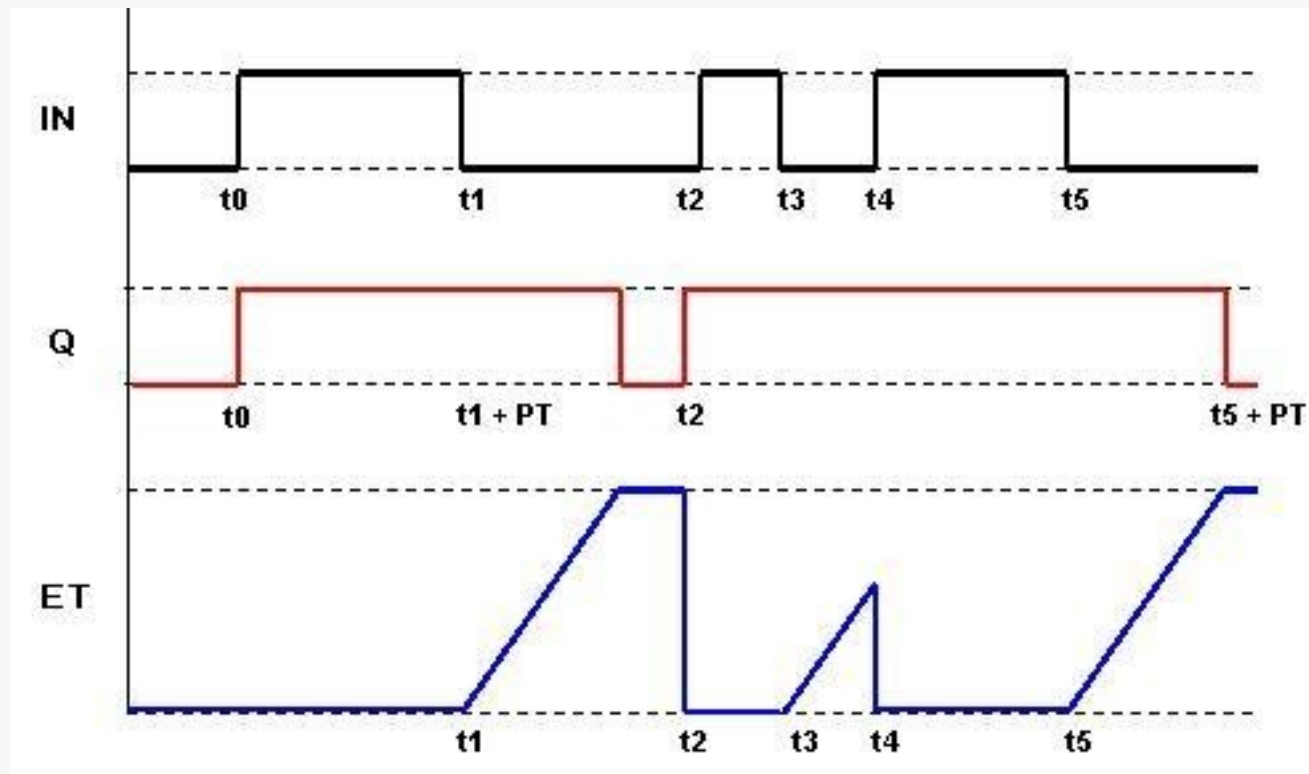
# Temporização e Contagem

## TOF - Temporizador: OFF Delay



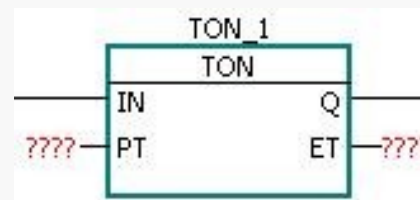
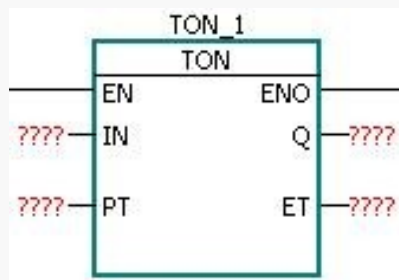
	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução. Pausa temporização ao desabilitar antes do fim da contagem de tempo. Entrada <b>opcional</b> . Existente somente no modo <u>com</u> EN/ENO.
	IN	BOOL	Iniciar temporização
	PT	TIME e CONSTANTE	Preset do temporizador
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN. Saída <b>opcional</b> . Existente somente no modo <u>com</u> EN/ENO.
	Q	BOOL	Desabilitado no fim da temporização
	ET	TIME	Efetivo do temporizador
	FLAG	NOME	DESCRIÇÃO
	-----	-----	Nenhum <i>flag</i> é afetado

*Exemplo gráfico de funcionamento:*



# Temporização e Contagem

## TON - Temporizador: ON Delay

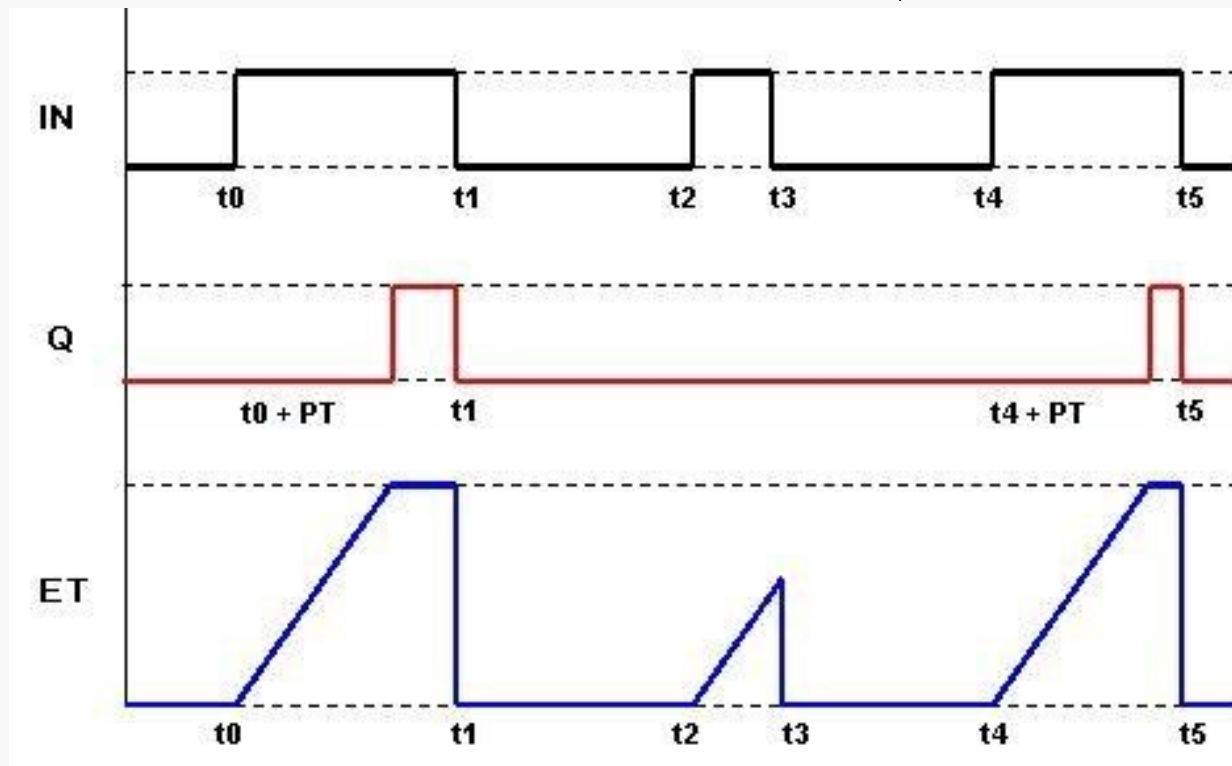


		NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN		BOOL	Habilita execução da instrução. Pausa temporização ao desabilitar antes do fim da contagem de tempo. Entrada <b>opcional</b> . Existente somente no modo <u>com</u> EN/ENO.
	IN		BOOL	Iniciar temporização
	PT		TIME e CONSTANTE	Preset do temporizador
	ENO		BOOL	Cópia do valor booleano de EN. Saída <b>opcional</b> . Existente somente no modo <u>com</u> EN/ENO.
SAÍDA	Q		BOOL	Habilitado no fim da temporização
	ET		TIME	Efetivo do temporizador
		FLAG	NOME	DESCRIÇÃO
		----	----	Nenhum <i>flag</i> é afetado

# Temporização e Contagem

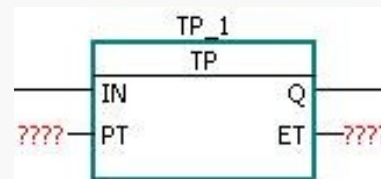
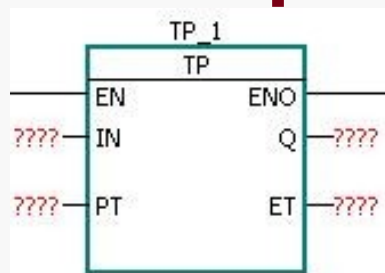
## TON - Temporizador: ON Delay

*Exemplo gráfico de funcionamento:*



# Temporização e Contagem

## TP - Temporizador: Pulse mode

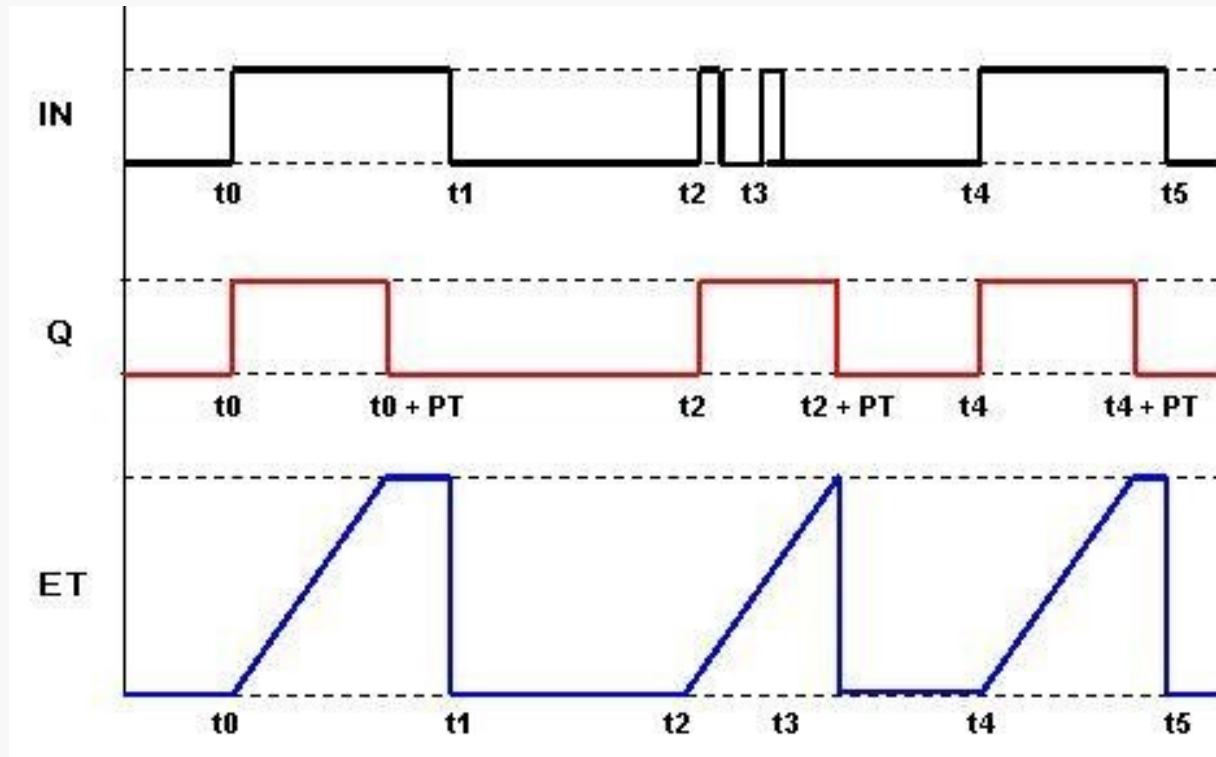


	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução. Pausa temporização ao desabilitar antes do fim da contagem de tempo. Entrada <b>opcional</b> . Existente somente no modo <u>com</u> EN/ENO.
	IN	BOOL	Iniciar temporização
	PT	TIME e CONSTANTE	Preset do temporizador
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN. Saída <b>opcional</b> . Existente somente no modo <u>com</u> EN/ENO.
	Q	BOOL	Habilitado no fim da temporização
	ET	TIME	Efetivo do temporizador
	FLAG	NOME	DESCRIÇÃO
	----	----	Nenhum <i>flag</i> é afetado

# Temporização e Contagem

## TP - Temporizador: Pulse mode

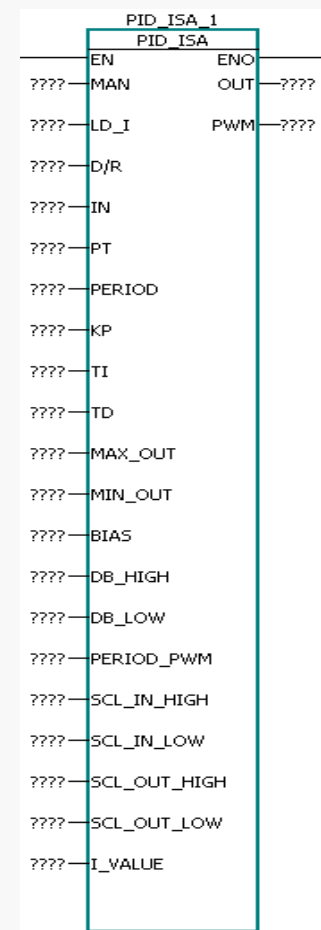
*Exemplo gráfico de funcionamento:*



# Especiais

## PID - Algoritmo PID ISA

	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	MAN	BOOL	Habilita controle PID em <a href="#">modo manual</a>
	LD_I	BOOL	Carrega valor do termo integral definido em I_VALUE
	D/R	BOOL	Define modo: <a href="#">direto/reverso</a>
	IN	INT	Variável de entrada (ex: canal de temperatura)
	PT	INT	Variável de Preset
	PERIOD	UINT	<a href="#">Período de amostragem</a>
	KP	UINT	<a href="#">Ganho proporcional</a>
	TI	UINT	<a href="#">Ganho integral</a>
	TD	UINT	<a href="#">Ganho derivativo</a>
	MAX_OUT	INT	Máximo valor da saída
	MIN_OUT	INT	Mínimo valor da saída
	BIAS	INT	Offset de saída
	DB_HIGH	INT	<a href="#">Banda morta alta</a>
	DB_LOW	INT	<a href="#">Banda morta baixa</a>
	PERIOD_PWM	UINT	Tempo do PWM da saída OUT (período)





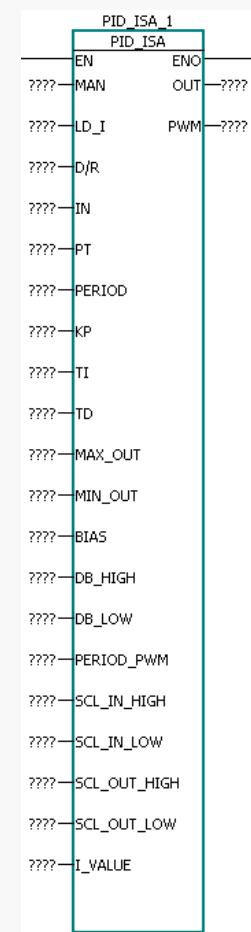
# Especiais

## PID - Algoritmo PID ISA

	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	SCL_IN_HIGH	INT	Máximo valor de escala para entrada
	SCL_IN_LOW	INT	Mínimo valor de escala para entrada
	SCL_OUT_HIGH	INT	Máximo valor de escala para saída
	SCL_OUT_LOW	INT	Mínimo valor de escala para saída
	I_VALUE	INT	Valor de carga do termo integral
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN
	OUT	INT	Variável de saída
	PWM	BOOL	Variável de saída PWM (booleana)
	FLAG	NOME	DESCRIÇÃO
	-----	-----	Nenhum flag é afetado

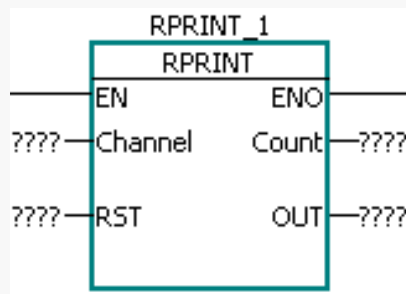
Equação PID Padrão ISA:

$$S = K \cdot \left( e(t) + K_i \int e dt + T_d \cdot de/dt \right) + \text{BIAS}$$



# Especiais

## RPRINT - Leitura de canal serial

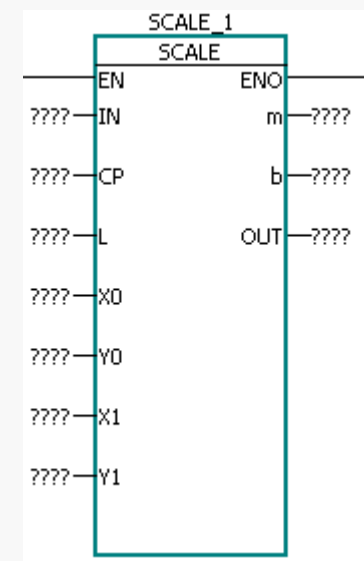


	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	Channel	BOOL	
SAÍDA	RST	BOOL	Cópia do valor booleano de EN
	ENO	BOOL	
	Count	UINT	
	OUT	STRING	
	FLAG	NOME	DESCRIÇÃO
	-----	-----	Nenhum <i>flag</i> é afetado

# Especiais

## SCALE - Ajuste de escala

	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	IN	INT , DINT , UINT , UDINT , REAL e TIME	Valor Efetivo da Entrada
	CP	BOOL	Habilita Cálculo dos Fatores “m” e “b”
	L	BOOL	Habilita Cálculo dos Limites da Saída
	X0	INT , DINT ,UINT , UDINT , REAL e TIME	Valor Inicial da Abscissa X
	Y0		Valor Inicial da Ordenada Y
	X1		Valor Final da Abscissa X
	Y1		Valor Final da Ordenada Y
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN
	m	REAL	Fator de Escalonamento
	b	REAL	Fator de Offset
	OUT	INT , DINT ,UINT , UDINT , REAL e TIME	Valor da Saída Calculada



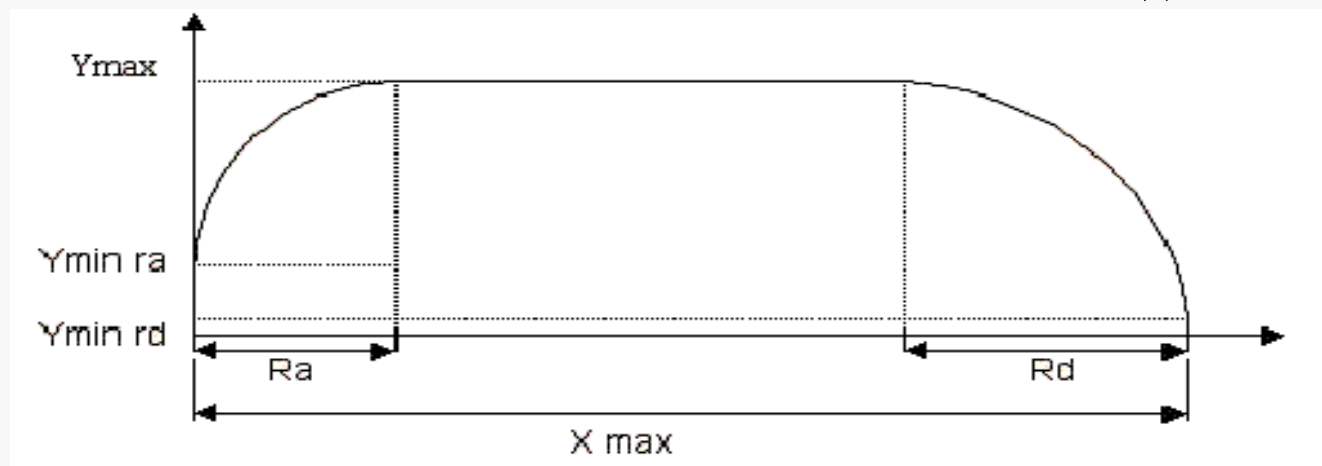
FLAG	NOME	DESCRIÇÃO
Overflow	OV	Será ligado se houver estouro de variável.
Zero	Z	Será ligado se o resultado de <b>b</b> ou da saída <b>OUT</b> for zero.
Sinal	S	Será ligado se valor de <b>OUT</b> for negativo.

# Especiais

## SCALE2G - Escala de 2º grau

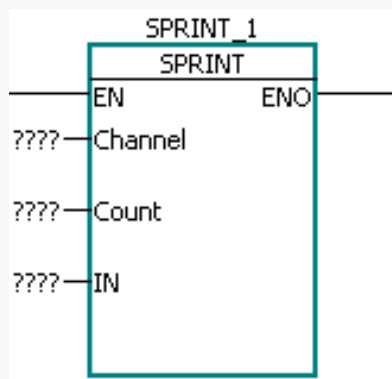
FLAG	NOME	DESCRIÇÃO	
	<b>Overflow</b>	<b>OV</b>	É ligado se houver estouro ou erro no calculo dos coeficientes.
	<b>Zero</b>	<b>Z</b>	É ligado se o valor da saída for ZERO.

SCALE_2G_1	
SCALE_2G	
EN	ENO
????-IN	OUT-????
????-K	Aa-????
????-YmRa	Ba-????
????-YmRd	Ca-????
????-Ymax	Ad-????
????-Xra	Bd-????
????-Xrd	Cd-????
????-Xmax	



# Especiais

## SPRINT - Escrita em canal serial



	NOME	TIPO DE DADO	DESCRIÇÃO
ENTRADA	EN	BOOL	Habilita execução da instrução
	Channel	BOOL	
	Count	UINT	
	IN	STRING	
SAÍDA	ENO	BOOL	Cópia do valor booleano de EN
FLAG	NOME	DESCRIÇÃO	
-----	-----	Nenhum <i>flag</i> é afetado	