

STANDARD IEC 61131

PRIMA PARTE – Introduzione

SECONDA PARTE – Elementi comuni

TERZA PARTE – Linguaggi di programmazione

1 - Introduzione

Un sistema di controllo di processi industriali deve avere tre caratteristiche fondamentali.

- ❑ MICROPROCESSORE (CPU con interfacce)
 - ❑ SISTEMA DI ACQUISIZIONE DATI DAL CAMPO
 - ❑ SISTEMA DI ATTUAZIONE SEGNALI DI CONTROLLO
-

1 - Introduzione

I sistemi di controllo possono essere raggruppati in due categorie.

□ **SISTEMI INTEGRATI (CUSTOM)**

vantaggi: elevate prestazioni, ottimizzazione del sistema.

svantaggi: costi progettazione, poco adattabili ad altri sistemi.

□ **SISTEMI DISTRIBUITI ("A BUS" – PLC, PC)**

vantaggi: adattabili a diversi sistemi, costi progettazione

svantaggi: non particolarmente ottimizzati.

1 - Introduzione

Struttura software per i sistemi di controllo.

- ACQUISIZIONE INGRESSI
- ELABORAZIONE PROGRAMMA
- AGGIORNAMENTO USCITE

1 - Introduzione

Sistemi di comunicazione per architetture di controllo distribuite. Si distinguono tre categorie.

- ❑ RETI INFORMATICHE (scambio dati ad alto livello dati di produzione etc. es. TCP/IP)
 - ❑ RETI PER IL CONTROLLO (svolgono funzioni di scambio dati tra i vari PLC/PC con protocolli proprietari dei costruttori)
 - ❑ RETI DI CAMPO (FIELDBUS; si interfacciano con periferiche e sensori intelligenti direttamente sugli impianti)
-

1 - Introduzione

Nei sistemi distribuiti il PLC, molto spesso, ha come “partner” di gestione del processo un PC con un sistema di supervisione ed acquisizione dati (SCADA).

... SOLO PC, SOLO PLC O ENTRAMBI ?

1 - Introduzione

SCADA

- ❑ RISORSE POTENTI DEDICATE AL MULTIMEDIA E AL MANAGEMENT DEI DATI MA POCO ADATTO ALLA LOGICA DI PROCESSO.
- ❑ BUS PCI CON CAPACITA' DI ESPANSIONE LIMITATA.

PLC

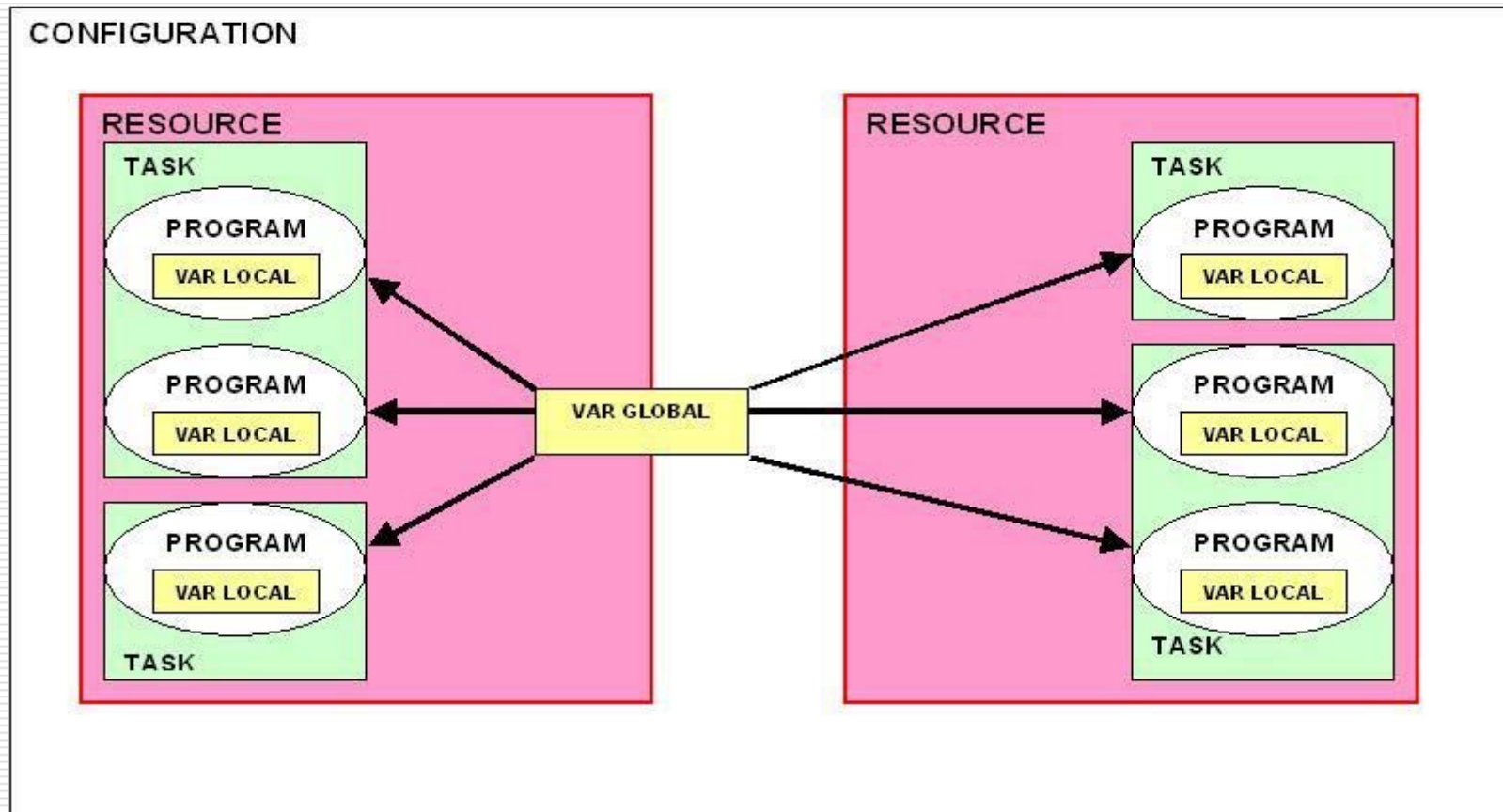
- ❑ RISORSE MENO POTENTI MA CONCENTRATE SULLA LOGICA DI PROCESSO.
- ❑ ELEVATA CAPACITA' DI ESPANSIONE ED ELEVATA VARIETA' DI SCHEDE (acquisizione ingressi, attuazione uscite, comunicazione, conteggio etc.)

2 – IEC 61131-3: elementi comuni

PERCHE' UNO STANDARD ?

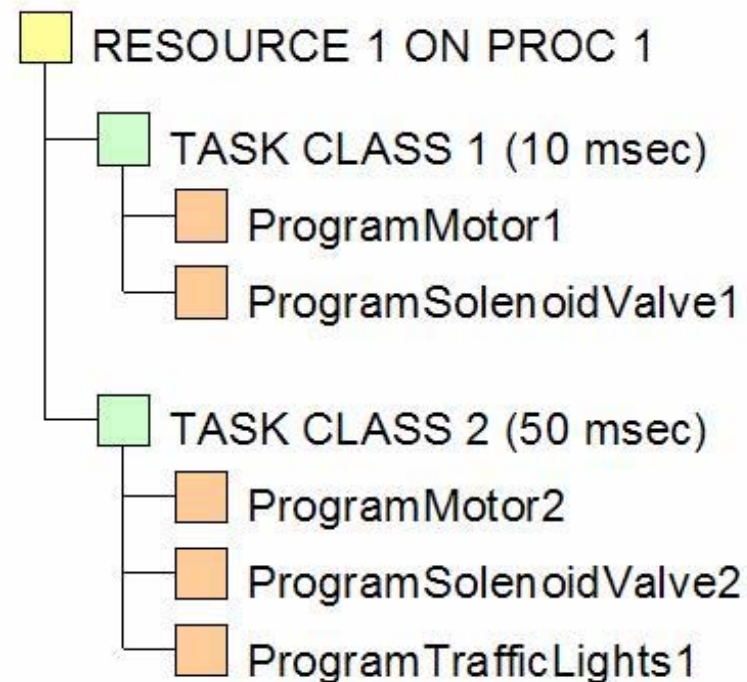
- ❑ La progettazione e lo sviluppo del software di controllo presenta alcune problematiche legate all'hardware del sistema.
 - ❑ Impossibilità di condividere codice tra sistemi differenti.
 - ❑ I committenti degli impianti impongono l'architettura del sistema costringendo il programmatore a cambiare piattaforma di sviluppo (lettura dei manuali, insicurezza sui nuovi strumenti, corsi propedeutici, ritardi etc.)
-

2 – IEC 61131-3: elementi comuni modello software



2 – IEC 61131-3: elementi comuni

Esempio architettura software di controllo



2 – IEC 61131-3: elementi comuni

- ❑ CONFIGURATION: descrizione di insieme del sistema di controllo.
 - ❑ RESOURCE: gestore di programmi e interfaccia con il campo, associata ad una CPU.
 - ❑ POU (program organization unit) : il codice ed i costrutti necessari all'elaborazione di un programma.
-

2 – IEC 61131-3: elementi comuni

POU (program organization unit)

- ❑ FB (function blocks): moduli di programma riutilizzabili con variabili statiche e temporanee, con ingressi ed una o più uscite.
 - ❑ FC (functions): moduli di programma riutilizzabili con variabili temporanee, con ingressi ed una sola uscita.
 - ❑ PROGRAM: moduli di programma ad un macro-livello.
-

2 – IEC 61131-3: elementi comuni

DICHIARAZIONE VARIABILI

DISTINZIONE PER ...

- ❑ TIPO DI UTILIZZO NELLA CONFIGURATION : **GLOBAL** e **LOCAL**.
 - ❑ TIPO DI UTILIZZO NELLA POU : **INPUT**, **OUTPUT** e **IN_OUT**.
 - ❑ TIPO DI GESTIONE DELLA MEMORIA : **A RIMANENZA**, **COSTANTI**, **ACCESSO PER INDIRIZZO**.
-

2 – IEC 61131-3: elementi comuni

TIPI DI DATO

- ❑ DATI DI TIPO **ELEMENTARE**: chiaramente definiti dallo standard con nome, descrizione e occupazione di memoria.
- ❑ DATI DI TIPO **DERIVATO**: definiti dall'utente.

2 – IEC 61131-3: elementi comuni

DATI DI TIPO ELEMENTARE

BOOL (0÷1), occupa 1 bit.

USINT (0÷255), occupa 1 byte.

SINT (-128÷127), occupa 1 byte.

UINT (0÷65535), occupa 2 bytes.

INT (-32768÷32767), occupa 2 bytes.

UDINT (0÷4294967295), occupa 4 bytes.

DINT (-2147483648÷2147483647), occupa 4 bytes.

REAL (-3.4e+38÷3.4e+38), occupa 4 bytes.

STRING, 1 carattere occupa 1 byte.

DATE_AND_TIME (min÷max), occupa 4 bytes.

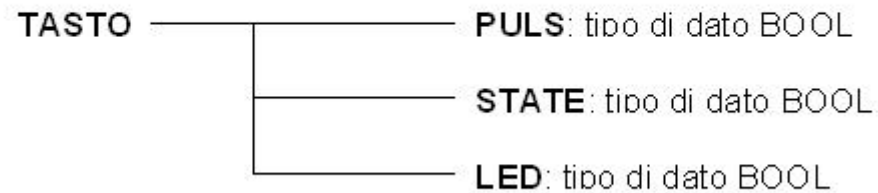
TIME (1 msec÷±24 days), occupa 4 bytes.

ARRAY DI DATI, dipende dal numero e dal tipo di elementi.

2 – IEC 61131-3: elementi comuni

DATI DI TIPO DERIVATO: ESEMPIO

Tipo di dato: **TASTO**



2 – IEC 61131-3: elementi comuni

DATI DI TIPO DERIVATO: ESEMPIO

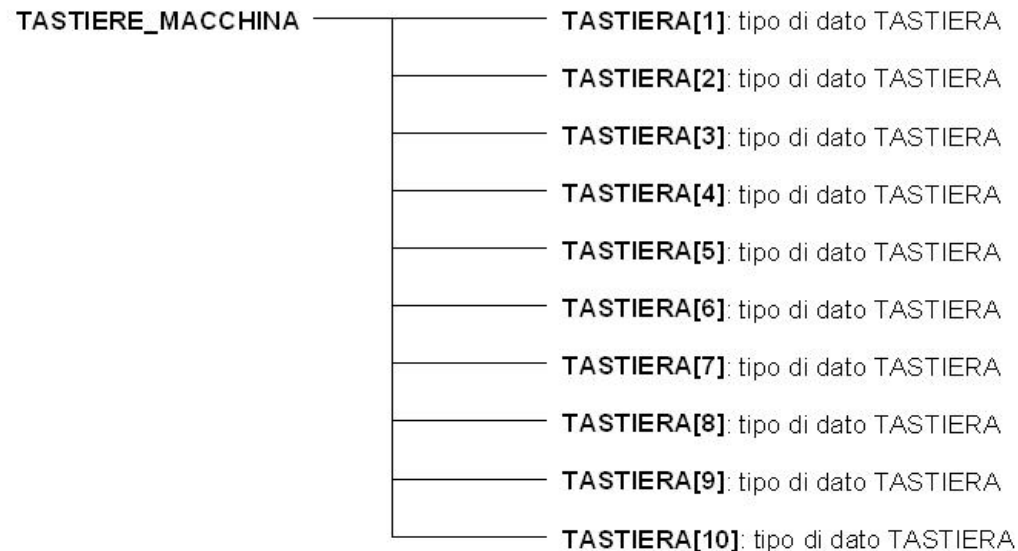
Dato: **"TASTIERA"**, tipo di dato: **ARRAY[6] di dati di tipo "TASTO"**
(deve essere dichiarato anche l'elemento "[0]")



2 – IEC 61131-3: elementi comuni

DATI DI TIPO DERIVATO: ESEMPIO

Dato: **"TASTIERE_MACCHINA"**, tipo di dato: **ARRAY[11]** di dati di tipo **"TASTIERA"**
(deve essere dichiarato anche l'elemento "[0]")



2 – IEC 61131-3: elementi comuni

DATI DI TIPO DERIVATO: ESEMPIO

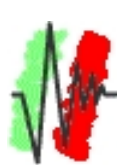
Dichiarazione variabile con struttura dati appena creata ...

Variabile: **"TASTIERE_COMMESSA_1"**

Tipo di dato: **"TASTIERE_MACCHINA"**

Tipo di memoria variabile: **"A RIMANENZA"**

Tipo di variabile: **"GLOBALE"**



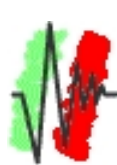
2 – IEC 61131-3: elementi comuni

DATI DI TIPO DERIVATO: ESEMPIO

... HO CABLATO IL FILO **"TASTIERA 4 TASTO 3 PREMUTO"**
ALL' **"INPUT 5"** DEL PLC ...

... ED IL FILO **"ATTIVA LED 2 TASTIERA 2"** ALL' **"OUTPUT 3"**
DEL PLC ...

???



2 – IEC 61131-3: elementi comuni

DATI DI TIPO DERIVATO: ESEMPIO

```
TASTIERA_COMMESSA_1.TASTIERA[4]. TASTO[3].PULS := INPUT5;
```

```
OUTPUT3 := TASTIERA_COMMESSA_1.TASTIERA[2]. TASTO[2].LED;
```

2 – IEC 61131-3: elementi comuni

FUNZIONI E BLOCCHI FUNZIONE PREDEFINITI: FUNZIONI DI CONVERSIONE

CONVERSIONE GENERICA

VAR2 := dato1_**TO**_dato2(VAR1);

ARROTONDAMENTO E TRONCAMENTO DI VALORI "REAL"

VAR2 := **TRUNC**(VAR1);

2 – IEC 61131-3: elementi comuni

FUNZIONI E BLOCCHI FUNZIONE PREDEFINITI: FUNZIONI ARITMETICHE

ABS: VALORE ASSOLUTO
SQRT: RADICE QUADRATA
LN: LOGARITMO IN BASE NATURALE
LOG: LOGARITMO IN BASE 10
EXP: ESPONENZIALE NATURALE
SIN: SENO
COS: COSENO
TAN: TANGENTE
ASIN: ARCOSENO
ACOS : ARCOSENO
ATAN : ARCOTANGENTE
ADD (+) : ADDIZIONE
SUB (-) : SOTTRAZIONE
MUL (*) : MOLTIPLICAZIONE
DIV (/) : DIVISIONE
MOD : MODULO (RESTO DELLA DIVISIONE TRA INTERI)
EXPT () :** ELEVAMENTO A POTENZA
MOVE (:=) : ASSEGNAZIONE

2 – IEC 61131-3: elementi comuni

FUNZIONI E BLOCCHI FUNZIONE PREDEFINITI: FUNZIONI DI SELEZIONE

SELEZIONE GENERICA

VAR3 := **SEL**(index_bool(VAR1 , VAR2));

SELEZIONE PER MASSIMO VALORE

VAR3 := **MAX**(VAR1 , VAR2);

SELEZIONE PER MINIMO VALORE

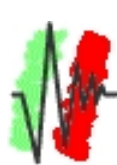
VAR3 := **MIN**(VAR1 , VAR2);

SELEZIONE PER VALORI COMPRESI TRA LIMITI

VAR2 := **LIMIT**(num_min , VAR1 , num_max);

SELEZIONE CON INDICE (Multiplexer)

VAR4 := **MUX**(index_integer(VAR1 , VAR2 , VAR3));



2 – IEC 61131-3: elementi comuni

FUNZIONI E BLOCCHI FUNZIONE PREDEFINITI: FUNZIONI BOOLEANE

AND: PRODOTTO LOGICO

OR: SOMMA LOGICA

XOR: SOMMA LOGICA CON COMPLEMENTO A DUE

SHL: SHIFT DI UNA STRINGA DI BIT VERSO SINISTRA

SHR: SHIFT DI UNA STRINGA DI BIT VERSO DESTRA

ROL: ROTAZIONE DI UNA STRINGA DI BIT VERSO SINISTRA

ROR: ROTAZIONE DI UNA STRINGA DI BIT VERSO DESTRA

2 – IEC 61131-3: elementi comuni

FUNZIONI E BLOCCHI FUNZIONE PREDEFINITI: FUNZIONI DI COMPARAZIONE

GT (>): MAGGIORE DI

GE (>=): MAGGIORE O UGUALE DI

EQ (=): UGUALE

LT (<): MINORE DI

LE (<=): MINORE O UGUALE DI

NE (<>): DIVERSO DA

2 – IEC 61131-3: elementi comuni

FUNZIONI E BLOCCHI FUNZIONE PREDEFINITI: FUNZIONI DI TEMPORIZZAZIONE E CONTEGGIO

TON: TIMER CON RITARDO SU ATTIVAZIONE

TOFF: TIMER CON RITARDO SU DISATTIVAZIONE

TP: TIMER AD IMPULSO

CTU: CONTATORE IN AVANTI

CTD: CONTATORE ALL'INDIETRO

CTUD: CONTATORE IN AVANTI O ALL' INDIETRO

3 – IEC 61131-3: linguaggi di programmazione

- LINGUAGGI DI TIPO TESTUALE.
Instruction List (IL) e Structured Text (ST).
 - LINGUAGGI DI TIPO GRAFICO.
Ladder Diagram (LD), Function Block Diagram (FBD) e Sequential Functional Chart (SFC).
-

3 – IEC 61131-3: linguaggi di programmazione

IL (Instruction List)

- ❑ LINGUAGGIO LETTERALE DI LIVELLO INFERIORE NELLO STANDARD.
 - ❑ UTILIZZO DI ACCUMULATORI.
 - ❑ LA SINTASSI E' COMPOSTA DA UN'EVENTUALE ETICHETTA INIZIALE, DA UN OPERATORE, DA UN EVENTUALE MODIFICATORE E DALL'OPERANDO.
-

3 – IEC 61131-3: linguaggi di programmazione

IL (Instruction List): esempio

START: LD Input1
ANDN Input2
ST Output1

3 – IEC 61131-3: linguaggi di programmazione

OPERATORI E MODIFICATORI AMMESSI

LD, carica valore dall'operando all'accumulatore (modificatori ammessi: **N**).

ST, assegna valore dall'accumulatore all'operando (modificatori ammessi: **N**).

S, se l'accumulatore ha valore 1 pone l'operando a 1 altrimenti lo lascia inalterato.

R, se l'accumulatore ha valore 0 pone l'operando a 0 altrimenti lo lascia inalterato.

NOT, negazione logica dell'accumulatore, non richiede operando.

AND-OR-XOR, (modificatori ammessi: **N** e "(").

ADD-SUB-MUL-DIV-MOD-GT-GE-EQ-NE-LE-LT, (modificatori ammessi: "(").

JMP, salta all'etichetta posta prima dell'operando (modificatori ammessi: **C** e **N**).

CAL, chiama il FUNCTION BLOCK indicato nell'operando (modificatori ammessi: **C** e **N**).

RET, non richiede operando, forza l'uscita di una FUNCTION, FUNCTION BLOCK o PROGRAM (modificatori ammessi: **C** e **N**).

MODIFICATORE "**C**": CONDIZIONA L'ISTRUZIONE PER VALORE "1" DELL'ACCUMULATORE

MODIFICATORE "**N**": CONDIZIONA L'ISTRUZIONE PER VALORE "0" DELL'ACCUMULATORE

3 – IEC 61131-3: linguaggi di programmazione

ST (Structured Text)

- ❑ LINGUAGGIO LETTERALE DI ALTO LIVELLO NELLO STANDARD.
 - ❑ PROGRAMMAZIONE STRUTTURATA GRAZIE A COSTRUTTI DI SELEZIONE E ITERAZIONE.
 - ❑ LA SINTASSI PRESENTA LE STESSE REGOLE DELLE ESPRESSIONI ALGEBRICHE, LOGICHE E MATEMATICHE.
 - ❑ ANCHE GLI OPERATORI NON SONO DI TIPO LETTERALE MA HANNO LA CLASSICA RAPPRESENTAZIONE ALGEBRICA (" $<>$ ", " $<=$ ", " $+$ " etc.)
-

3 – IEC 61131-3: linguaggi di programmazione

ST (Structured Text): esempio

```
Output1 := Input1 AND NOT Input2;
```

3 – IEC 61131-3: linguaggi di programmazione

ST (Structured Text). Costrutti di selezione: "CASE"

```
CASE VAR1 OF
    1: VAR2 := 1;
    2: VAR2 := 2;
ELSE
    VAR2 := 0;
END_CASE;
```

3 – IEC 61131-3: linguaggi di programmazione

ST (Structured Text). Costrutti di selezione: “IF ... THEN
... ELSE ...”

```
IF VAR1 = 1 THEN
    VAR2 := 1;
ELSIF VAR1 = 2 THEN
    VAR2 := 2;
ELSE
    VAR2 := 0;
END_IF;
```

3 – IEC 61131-3: linguaggi di programmazione

ST (Structured Text). Costrutti di iterazione:
"WHILE"

```
Counter := 0;  
VAR      := 10;  
WHILE Counter < 10 DO  
    VAR := VAR - 1;  
    Counter := Counter + 1;  
END_WHILE;
```

3 – IEC 61131-3: linguaggi di programmazione

ST (Structured Text). Costrutti di iterazione: "REPEAT"

```
Counter := 0;  
VAR      := 10;  
REPEAT  
    VAR := VAR - 1;  
    Counter := Counter + 1;  
UNTIL Counter >= 10;  
END_REPEAT;
```

3 – IEC 61131-3: linguaggi di programmazione

ST (Structured Text). Costrutti di iterazione:
"FOR"

```
VAR := 10;  
FOR Counter := 0 TO 9 BY 1 DO  
    VAR := VAR - 1;  
    IF VAR_BOOL THEN  
        EXIT;  
END_FOR;
```

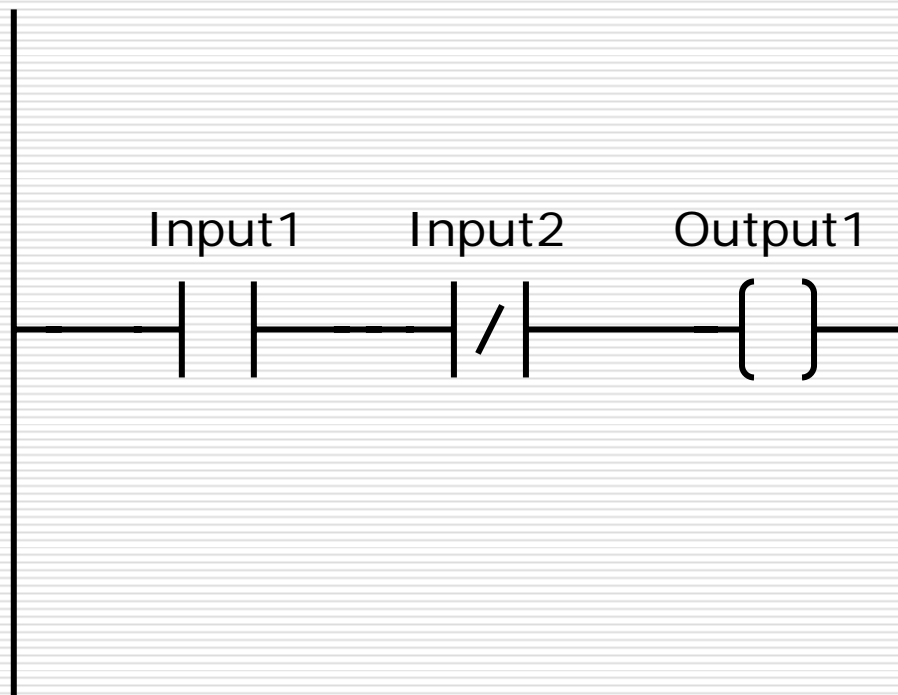
3 – IEC 61131-3: linguaggi di programmazione

LD (Ladder Diagram)

- ❑ PRIMO EDITOR GRAFICO NATO COME RAPPRESENTAZIONE DEL LINGUAGGIO IL.
- ❑ RAPPRESENTATO DA UN FLUSSO VIRTUALE DI CORRENTE TRA DUE BARRE DI POTENZIALE.
- ❑ LA SINTASSI PRESENTA REGOLE DI TIPO ELETTROTECHNICO CON CONTATTI, INTERRUTTORI E BOBINE.
- ❑ FUNZIONI E BLOCCHI FUNZIONE (STANDARD O NO) SONO RICHIAMATI TRAMITE CABLAGGI DI BOX.

3 – IEC 61131-3: linguaggi di programmazione

(Ladder Diagram): esempio



3 – IEC 61131-3: linguaggi di programmazione

LD (Ladder diagram): Operatori.

 Contatto N.O.

 Contatto N.C.

 Contatto fronte pos.

 Contatto fronte neg.

 Salto condizionato

 Salto incondizionato

 Bobina momentanea

 Bobina momentanea negata

 Bobina con memoria

 Bobina con memoria negata

 Bobina per fronte di salita

 Bobina per fronte di discesa

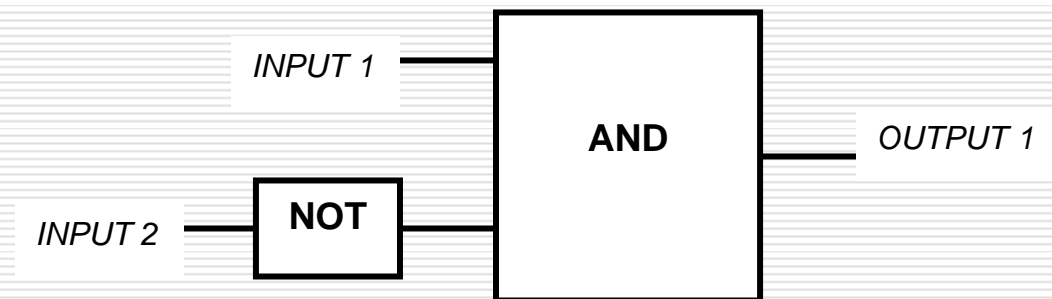
3 – IEC 61131-3: linguaggi di programmazione

FBD (Funcional Block Diagram)

- EDITOR GRAFICO RAPPRESENTATO DA FLUSSO DI SEGNALI ATTRAVERSO BLOCCHI GRAFICI.
- I BLOCCHI SONO RAPPRESENTATI IN MODO ANALOGO AI CIRCUITI ELETTRONICI

3 – IEC 61131-3: linguaggi di programmazione

FBD (Function Block Diagram): esempio



3 – IEC 61131-3: linguaggi di programmazione

SFC (Sequential Funcional Chart)

- ❑ EDITOR GRAFICO RAPPRESENTATO DA FLUSSO DI SEGNALI ATTRAVERSO BLOCCHI GRAFICI.
 - ❑ I BLOCCHI GRAFICI COSTITUENTI LA RETE SONO PROGRAMMABILI E RAPPRESENTANO LO STATO DELLE VARIABILI.
 - ❑ I BLOCCHI GRAFICI SONO CONNESSI TRA LORO ATTRAVERSO DELLE TRANSIZIONI, ANCH'ESSE PROGRAMMABILI. ESSE RAPPRESENTANO LE TRANSIZIONI DI STATO DELLE VARIABILI.
 - ❑ RAPPRESENTAZIONE TIPICA DELL'AUTOMA A STATI FINITI
-

3 – IEC 61131-3: linguaggi di programmazione SFC (Sequential Functional Chart): esempio

