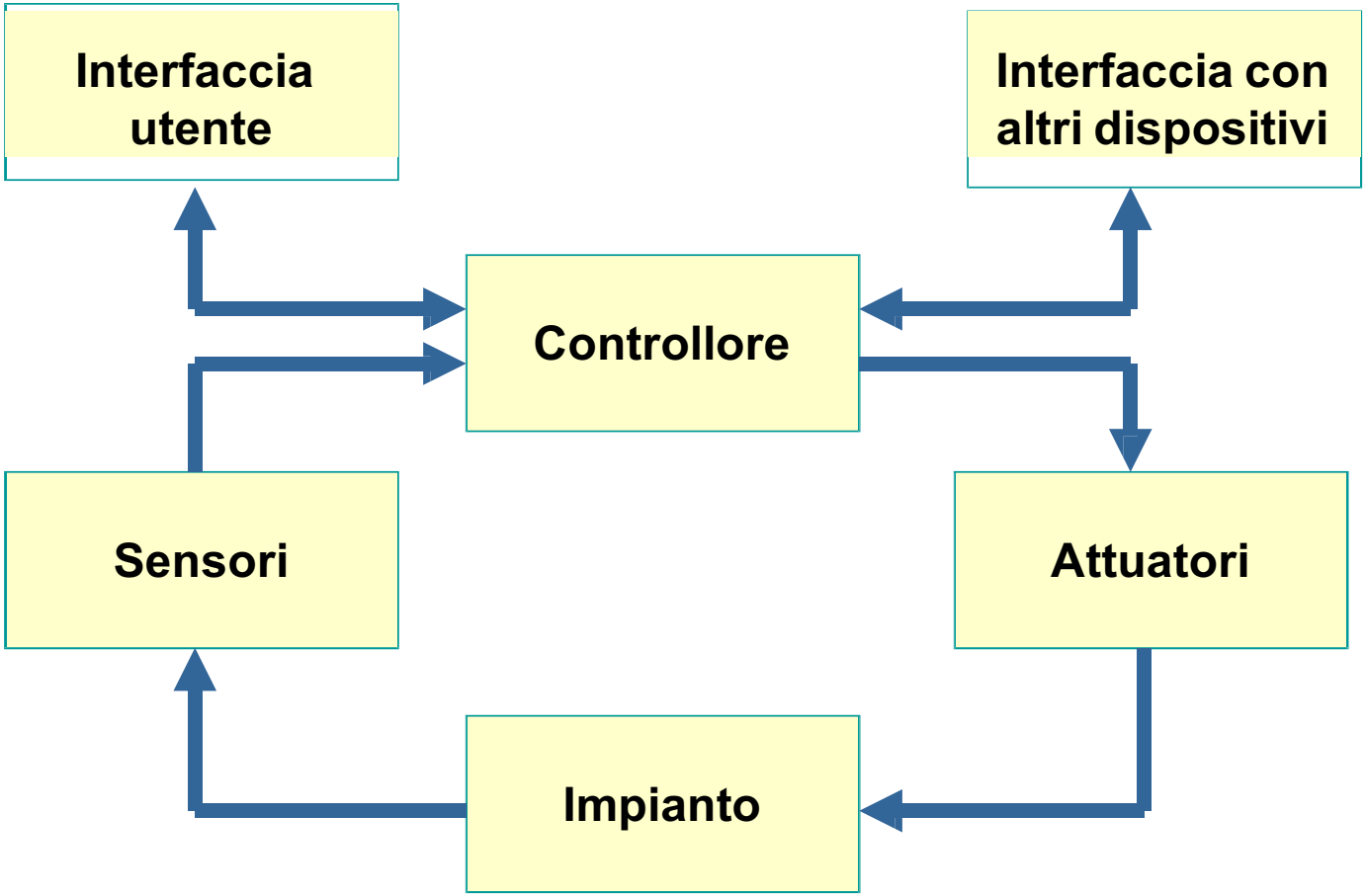
Sistemi per l'automazione di fabbrica

Introduzione

- In queste slide si fornirà un quadro, necessariamente sommario, del ruolo dei sistemi informatici in un sistema di automazione industriale, con riferimento in particolare alle architetture di controllo, alle reti di comunicazione, ai PLC e ai sistemi in tempo reale ed embedded.
- È importante essere a conoscenza almeno di un quadro di insieme di queste problematiche, peraltro in rapida e continua evoluzione.
- Studiare l'architettura (hardware e software) dei sistemi di controllo è un compito che va al di là, per sua stessa natura, dell'ambito dei sistemi meccatronici. Si farà quindi nel seguito riferimento a sistemi di controllo ed automazione non limitati strettamente al campo meccatronico.
- Verranno fornite agli studenti durante la lezione slides sui sistemi di automazione di fabbrica del corso di Automazione industriale del EPFL

Architettura generale di un sistema di automazione/controllo



Sistemi di controllo

.

Si deve fare una distinzione preliminare tra due modalità di sistemi di controllo:

- Controllo di regolazione(feed back)
  - controllo eseguito (in tecnologia analogica o digitale) da dispositivi che ad ogni istante si pongono come obiettivo che le variabili controllate inseguano i rispettivi riferimenti.
- Controllo logico-sequenziale
  - controllo eseguito da dispositivi che devono assicurare lo svolgimento di una o più sequenze di attività, la cui evoluzione è dettata dal verificarsi di eventi (come la conclusione di un'attività, l'insorgenza di anomalie, l'interazione con l'operatore).

Spesso il controllo logico opera a livelli più alti in un sistema di controllo, il controllo modulante a livelli più bassi.

Controllo logico-sequenziale

Alcuni esempi di problemi di controllo logico:

- controllo di un ascensore
- distributore automatico di bibite
- sistema di illuminazione
- sistema semaforico

In tutti questi esempi sono previste delle azioni che devono essere eseguite al verificarsi di determinati eventi.

La sequenza delle azioni è gestita dal controllo logico, l'esecuzione delle singole azioni dal controllo modulante.

Un sistema costituito da un robot che preleva pezzi da un nastro trasportatore e li manipola comporta per esempio problematiche sia di controllo di regolazione sia di controllo logico.

Controllo logico

Le specifiche di un sistema di controllo logico sono costituite da sequenze di azioni, descritte generalmente in linguaggio naturale.

Si pongono due problemi:

- formalizzare queste specifiche
- determinare metodologie di sintesi di un controllore che ne garantisca il soddisfacimento

Per quanto riguarda il primo aspetto, esistono formalismi di progressiva diffusione.

Per quanto riguarda il secondo aspetto, si fa generalmente ricorso alla teoria dei sistemi ad eventi discreti.

Sistemi a eventi discreti

Un sistema ad eventi discreti è un sistema dinamico (quindi dotato di stato) in cui:

- Lo stato assume valori in un insieme discreto
- L'evoluzione dello stato è dettata esclusivamente dall'occorrenza di eventi asincroni (ossia la cui temporizzazione non rispetta caratteristiche di regolarità).

Un esempio classico è costituito dalla coda di persone ad uno sportello, in cui lo stato è costituito dal numero di persone in coda e gli eventi sono l'arrivo di nuove persone in coda o l'uscita di altre.

Tra gli strumenti matematici di più largo utilizzo per la modellazione dei sistemi ad eventi discreti vi sono gli automi a stati finiti e le reti di Petri.

Pur sottolineandone l'importanza ai fini della progettazione e della comprensione di un sistema di automazione industriale, queste problematiche non saranno trattate nelle presenti lezioni.

Architettura di un sistema di controllo

Il sistema di controllo elabora informazione (acquisendola dai sensori e trasferendola agli attuatori) e come tale il suo progetto, nei casi più complessi, va condotto con le tecniche proprie dell'ICT (Information and Communication Technology).

Vi sono vari livelli in cui tale progetto si articola (si usa la terminologia standard IEEE):

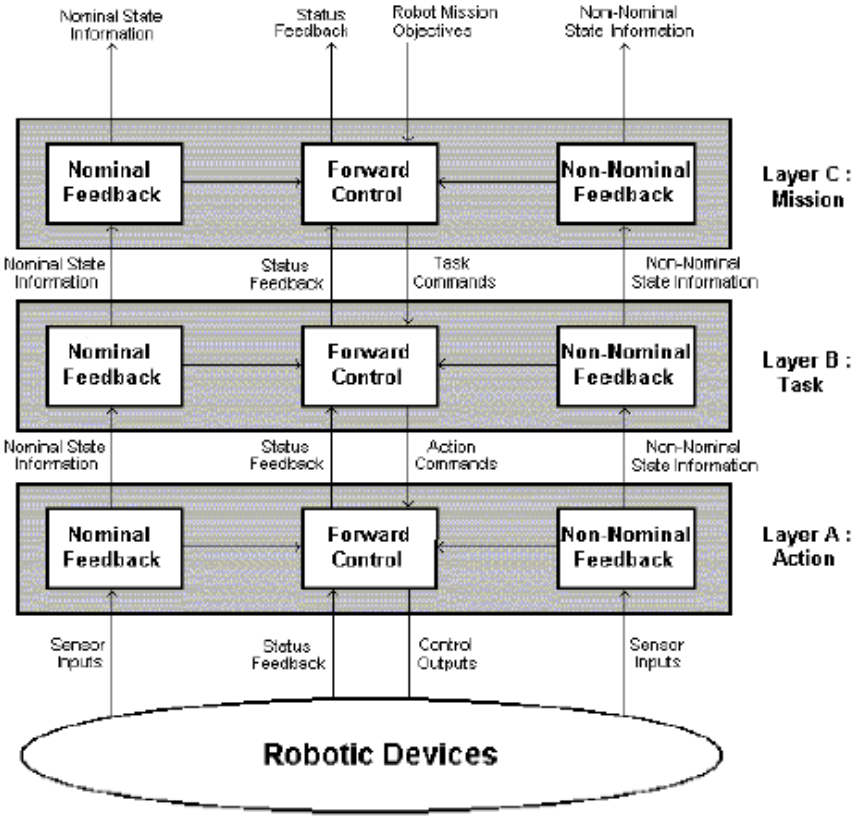
- Preliminary Design
  - si analizzano alternative di progetto e si definiscono preliminarmente l'architettura, i componenti e le interfacce
- Functional Design
  - si definiscono le funzionalità e le interfacce tra i componenti del sistema
- Detailed Design
  - si rifinisce ed espande il progetto preliminare del sistema e dei componenti a livello tale che il progetto è sufficientemente completo per cominciarne l'implementazione
- Architectural Design
  - si definiscono i componenti hardware e software del sistema e le loro interfacce

Progetto preliminare

- Una prima fase progettuale consiste nella raccolta e nell'analisi dei requisiti d'uso del sistema
- Occorre specificare sia il comportamento atteso del sistema (che cosa deve fare) sia il comportamento escluso (che cosa è bene che il sistema non faccia)
- Successivamente si passa alla definizione di massima delle funzionalità del sistema
- In generale si può dire che le funzionalità del sistema di controllo sono scomposte in funzionalità più semplici (divide et impera)
- Si ottiene una gerarchia di funzionalità da implementare con gli opportuni componenti
- Si trasferiscono metodologie di progetto tipiche del mondo informatico all'ambito controllistico o di automazione più in generale

Un esempio: CDM dell'ESA

Un esempio di scomposizione gerarchica di funzionalità è la CDM (Control Development Methodology) dell'ESA (European Space Agency), articolata nei livelli Mission, Task e Action



Es.: REPAIR a satellite

Es.: OPEN a door

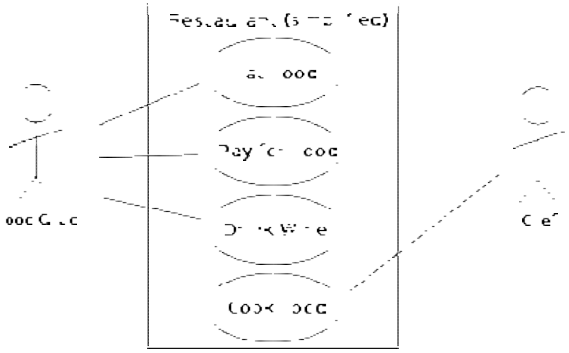
Es.: TIGHTEN a screw

Diagrammi UML

Nella definizione delle funzionalità di un sistema informatico sono molto usati formalismi grafici.

Uno dei più noti è UML (Unified Modelling Language)

- UML fornisce una serie di strumenti grafici per creare modelli visuali di sistemi software orientati agli oggetti
- È oggi uno standard nel progetto di questi sistemi



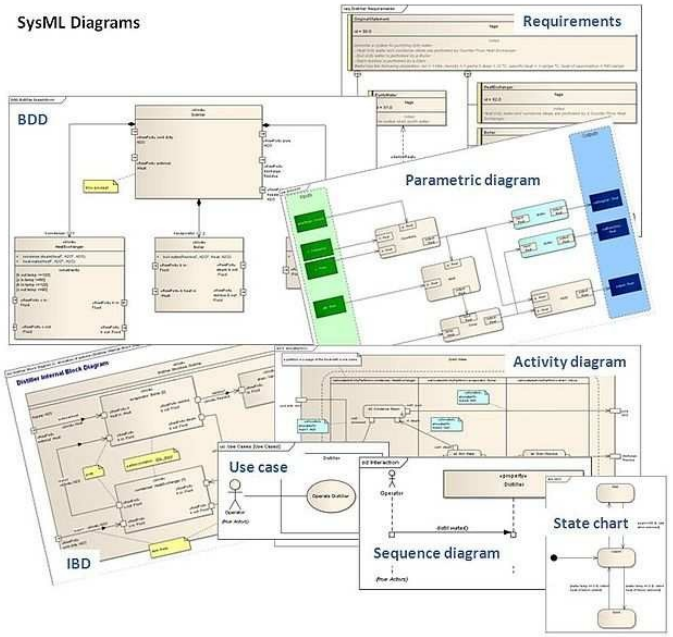
Esempio di “use case diagram” in UML

Progetto funzionale

L'architettura funzionale del sistema di controllo:

- definisce i singoli componenti per mezzo delle loro interfacce
- definisce le interazioni tra i componenti ed in particolare i canali di comunicazione ed i dati
- definisce per ciascun componente le funzioni, i requisiti in termini di capacità di memoria e le loro responsabilità

- Anche lo sviluppo dell'architettura funzionale si può condurre con strumenti grafici di ausilio alla progettazione.
- Ad esempio si usano diagrammi SysML (evoluzione del linguaggio UML).



Progetto hardware/software

L'implementazione finale del sistema informatico, dopo il progetto di dettaglio, passa dalla realizzazione dell'architettura hardware e software:

Architettura hardware

- Componenti hardware del sistema di controllo
  - Sensori
  - Attuatori
  - Organi di controllo
  - Canali di comunicazione (cavi…)

Architettura software

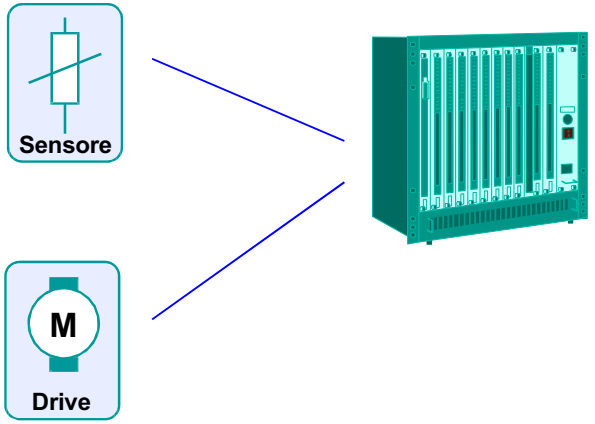
- Componenti software del sistema di controllo
- Connessioni logiche tra i componenti
- Dati che fluiscono tra i componenti

Alcune problematiche architetturali

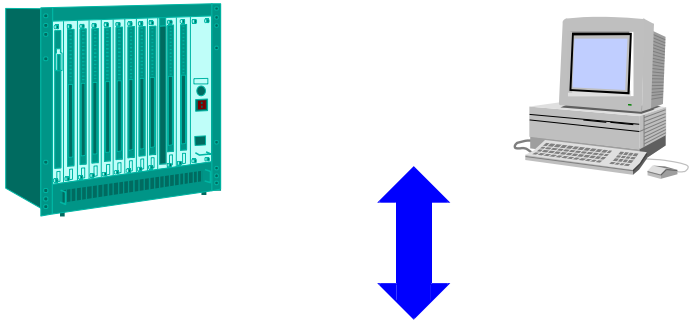
Lo sviluppo dell'architettura hardware/software di un sistema di controllo (e in particolare di uno per applicazioni meccatroniche) deve tener conto di alcune peculiarità che lo differenziano dallo sviluppo di un generico sistema informatico:

- Grandi quantità di dati e segnali da gestire con diverse scale temporali, d'importanza e di criticità
- Determinismo temporale (particolarmente critico in meccatronica).
- Semplicità e comodità nella costruzione e nella (ri)configurazione del sistema (mettere le mani su un sistema di controllo dentro un impianto non è come farlo su una rete di PC in un ufficio).
- Tolleranza ai guasti ed ai malfunzionamenti (sia del software sia dell'hardware).
- Interfaccia operatore (che coinvolge anche l'hardware) adatta a personale con grado di cultura ed addestramento molto variabile.

Connessioni di componenti di campo

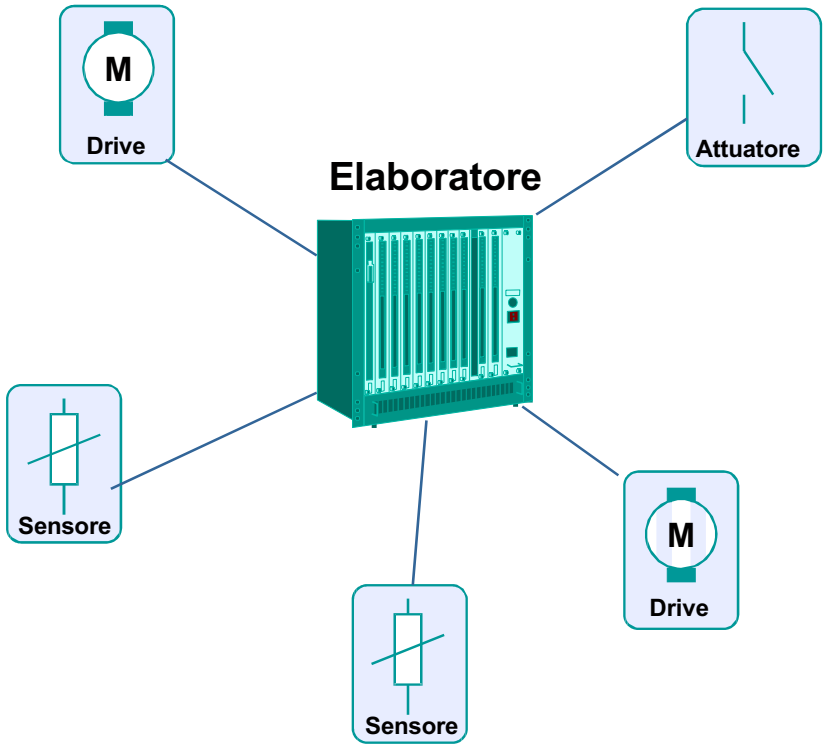


Nel caso più semplice di architettura hardware di sistema di controllo, due componenti “di campo”, un trasduttore e un attuatore sono collegati al controllore, con collegamenti analogici (in corrente o tensione).



Come sono organizzate le connessioni se vi sono numerosi componenti di campo?

Architettura tradizionale



Architettura centralizzata con collegamenti di tipo analogico punto – punto

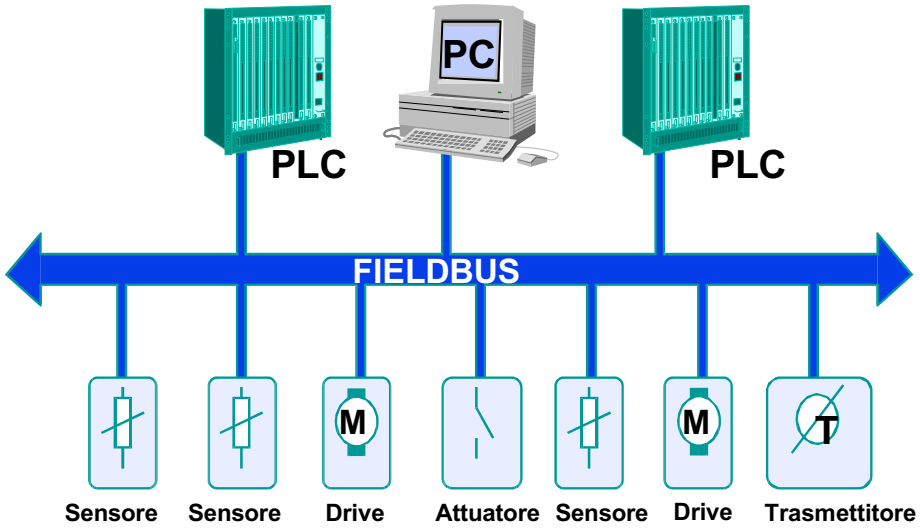
Vantaggi

- Affidabilità, sistema collaudato

Svantaggi

- Elevato numero collegamenti
- Costo dei cablaggi
- Elevata sensibilità ai disturbi (necessità di schermatura)
- Scarsa flessibilità e scalabilità

Architettura a bus



Architettura a bus con trasmissione digitale dei segnali

Vantaggi

- Risparmio sui costi di cablaggio
- Facilità di aggiunta e rimozione di dispositivi
- Condivisione delle risorse
- Flessibilità
- Decentramento di funzioni
- Intelligenza distribuita (funzionalità diagnostiche locali)

Svantaggi

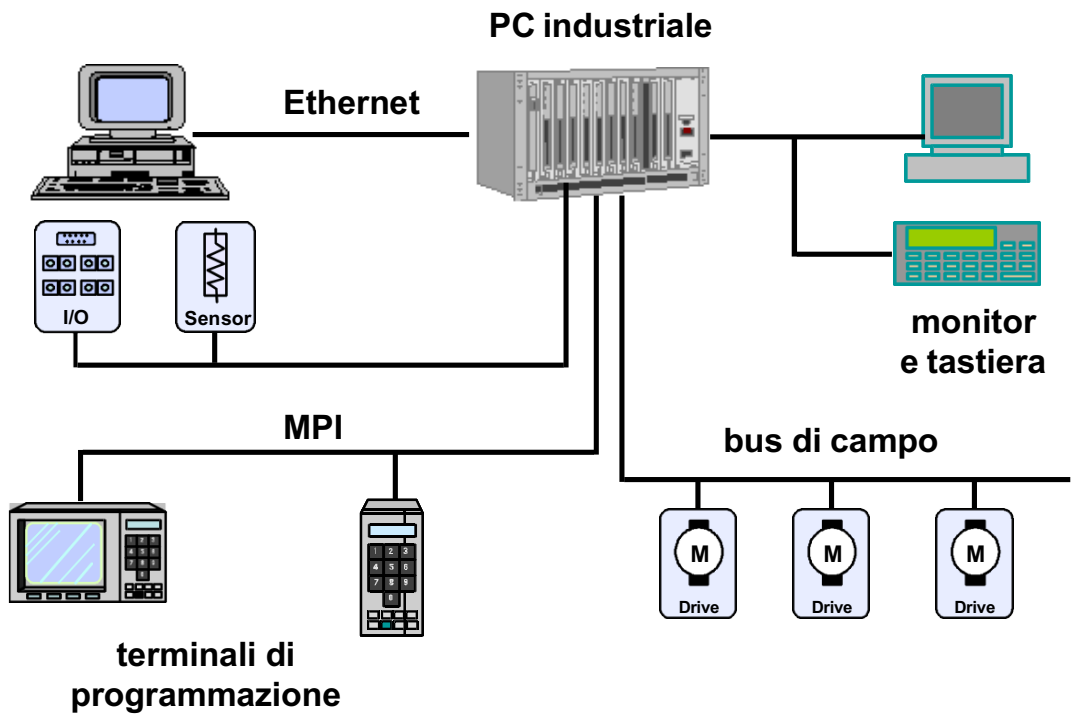
- Maggior costo dei dispositivi
- Diverse metodologie di progettazione
- Problemi di interoperabilità (far funzionare insieme dispositivi di produttori diversi)

Un esempio di architettura hw: CNC

- I CNC (Computer Numerical Control) sono dispositivi utilizzati nel controllo di macchine utensili e altre macchine di produzione.
- Il compito primario del CNC è quello della pianificazione o generazione delle traiettorie dell'utensile e conseguentemente degli assi di macchina.
- Il valore di un CNC è per la maggior parte costituito dal software estremamente sofisticato necessario per la pianificazione e l'esecuzione dei programmi di lavoro (cosiddetti *part program*)
- L'architettura software è quindi estremamente complessa

Nel seguito si accennerà all'architettura hardware del Simumerik 840Di di Siemens, come esempio di struttura complessa, articolata in più componenti connessi da opportune comunicazioni digitali.

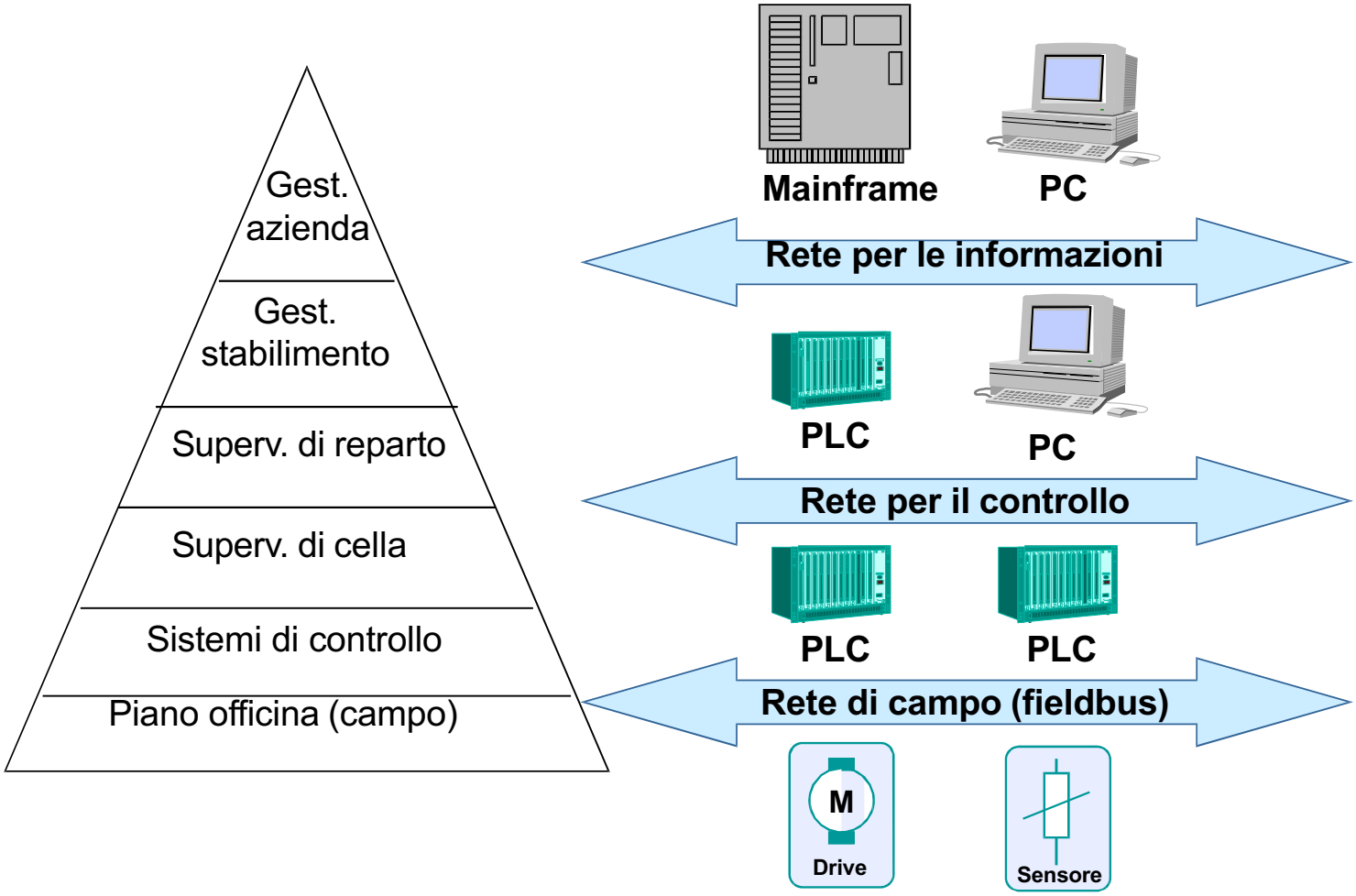
Un esempio di architettura hw: CNC



- PC industriale: PC con caratteristiche costruttive adatte ad uso in ambito industriale
- Scheda motion control sul PC industriale con funzioni PLC e interfaccia a bus di campo
- MPI: Multi Point Interface: interfaccia veloce verso terminali di comando

Nel CNC si chiude l'anello di controllo di posizione, mentre quelli di velocità e corrente si chiudono sugli azionamenti. La comunicazione verso gli azionamenti avviene tramite "bus di campo". Nel Sinumerik si usa il bus PROFIBUS DP.

Sistema CIM e reti



Sistema CIM e reti

- Reti per le informazioni
  - collegano i sistemi informativi di alto livello con altri elementi informativi di azienda (livelli 4, 5 e 6 della piramide CIM)
  - non vi sono specifiche di tempo reale
  - le informazioni sono di tipo complesso (file, ecc..)

- Reti per il controllo
  - collegano i dispositivi dedicati al controllo con quelli di supervisione (livelli CIM 2, 3 e 4)
  - vi sono specifiche di correttezza e vincoli temporali
  - le informazioni sono di tipo non molto complesso
  - si tratta in genere di reti proprietarie

- Reti di campo (fieldbus)
  - collegano i controllori (modulanti e logici) con sensori ed attuatori dotati di interfaccia digitale
  - specifiche stringenti di tempo reale
  - le informazioni sono di tipo semplice

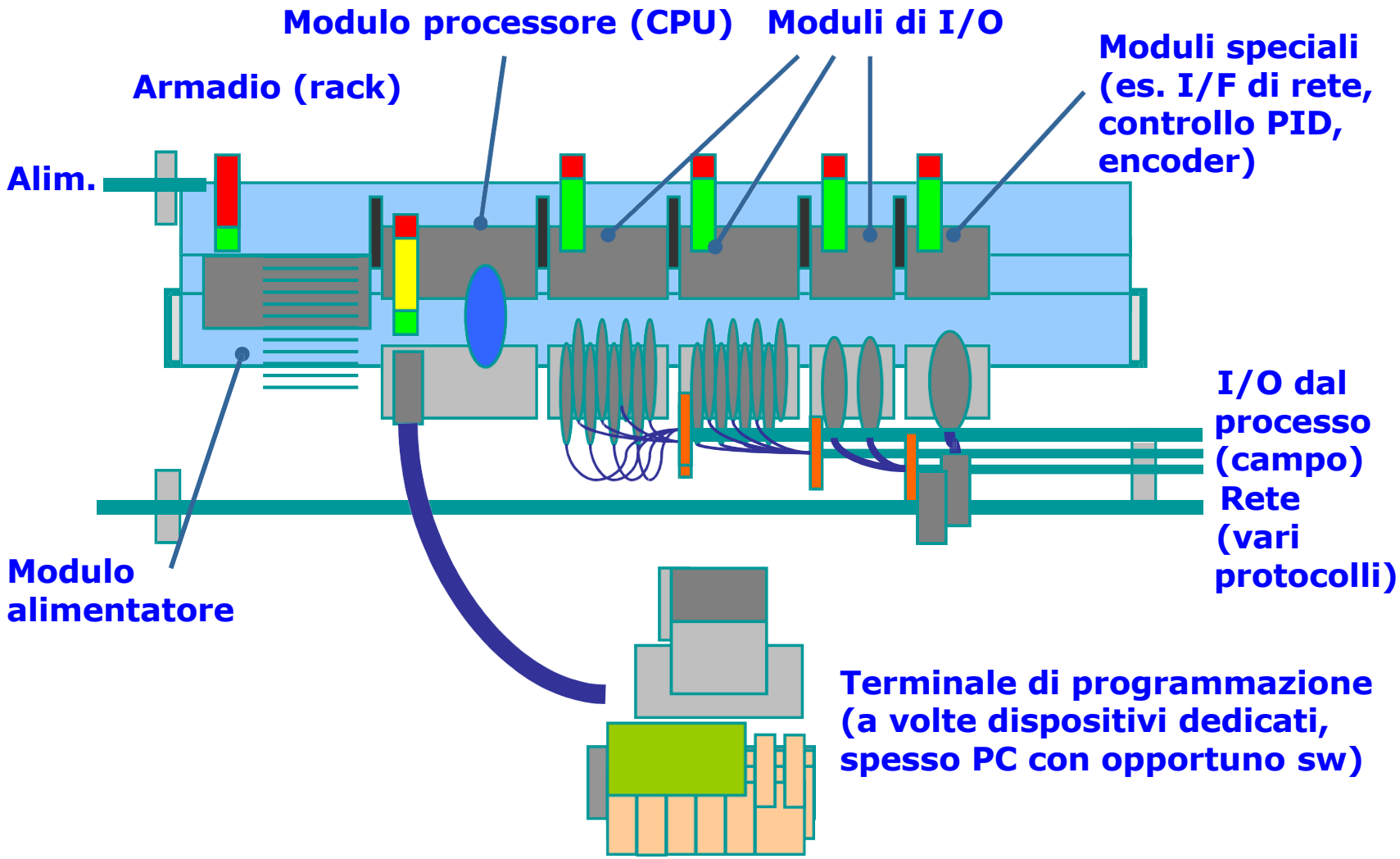
Che cos'è un PLC?

Un PLC (Programmable Logic Controller) è un dispositivo che fu introdotto nell'automazione per sostituire i circuiti sequenziali a relay, sostituendoli con un dispositivo in grado di acquisire ingressi ed eseguire su di essi operazioni logiche previa programmazione.

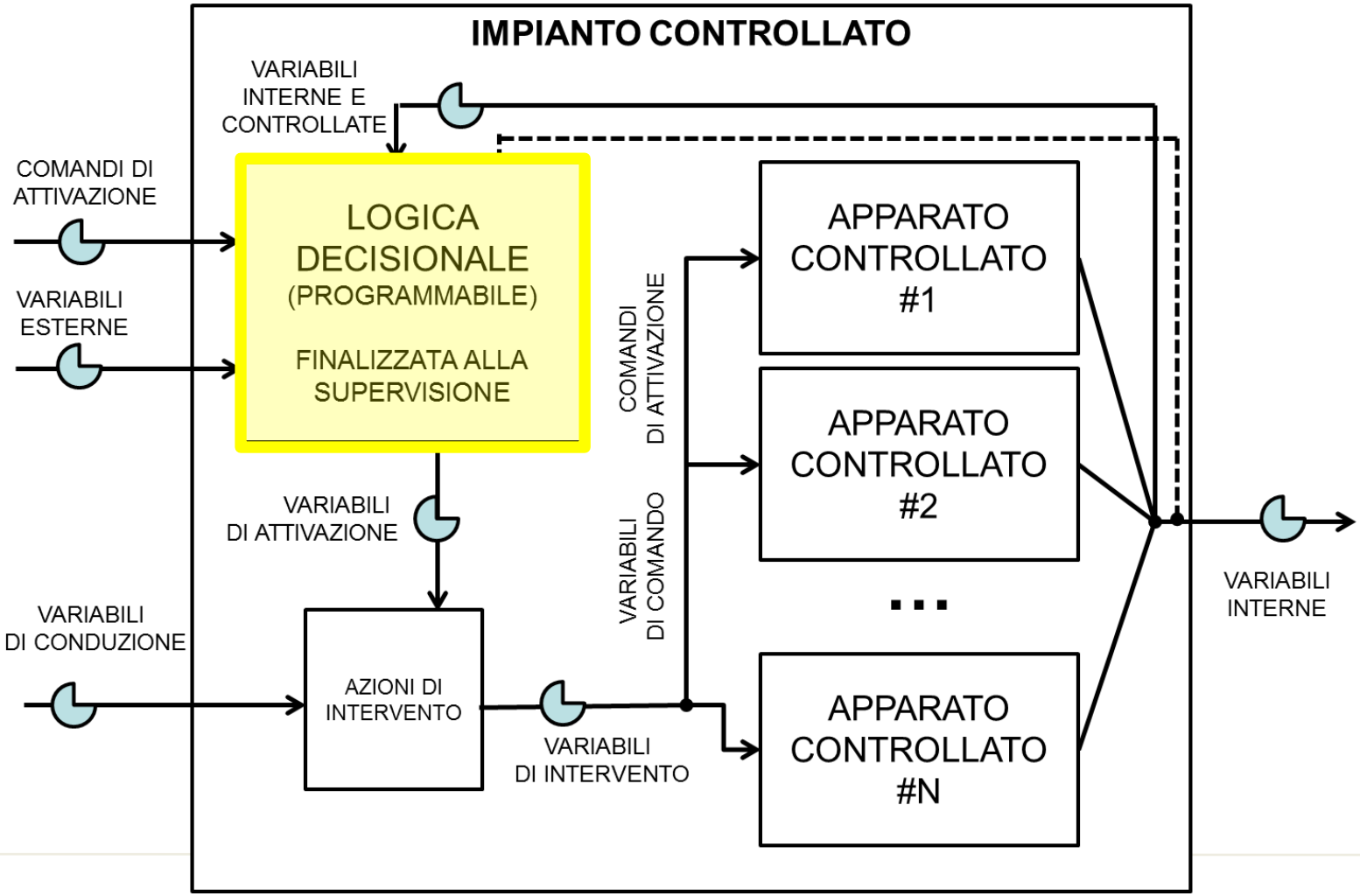
La diffusione dei PLC in automazione è enorme: essi infatti realizzano il "controllo logico" che insieme al controllo modulante svolge funzioni di basso livello essenziali per il processo produttivo.

Typical PLCs

Componenti fondamentali

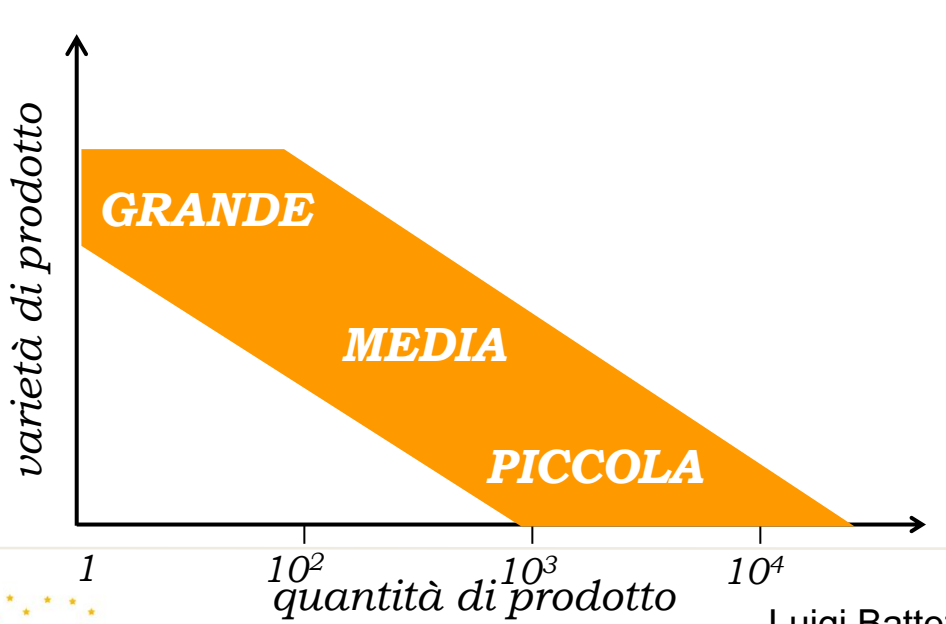


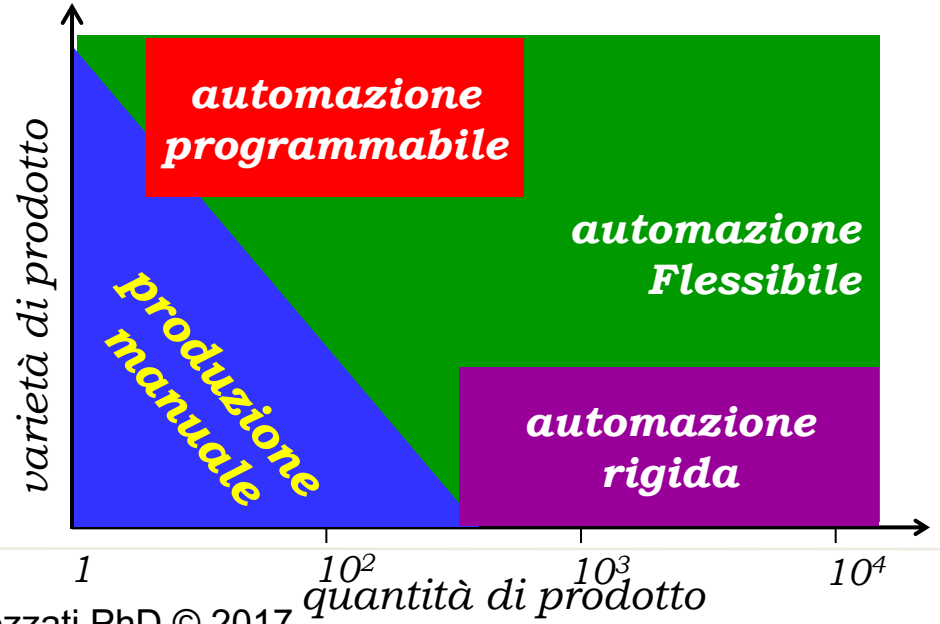
CONTROLLORI LOGICI



ETEROGENEITÀ E QUANTITÀ DI PRODOTTO

- La **produttività** di un sistema controllato dipende dalla **quantità** del prodotto realizzata **per unità di tempo**, che a sua volta è collegata alla **eterogeneità** della produzione del sistema controllato nonché alle modalità con cui è stata resa operativa l'automazione.





ORIGINI DEI CONTROLLORI LOGICI

I controllori logici furono realizzati per poter fare evolvere la **produzione di serie da manuale ad automatizzata**. La loro realizzazione dipendeva dalle tecnologie disponibili.

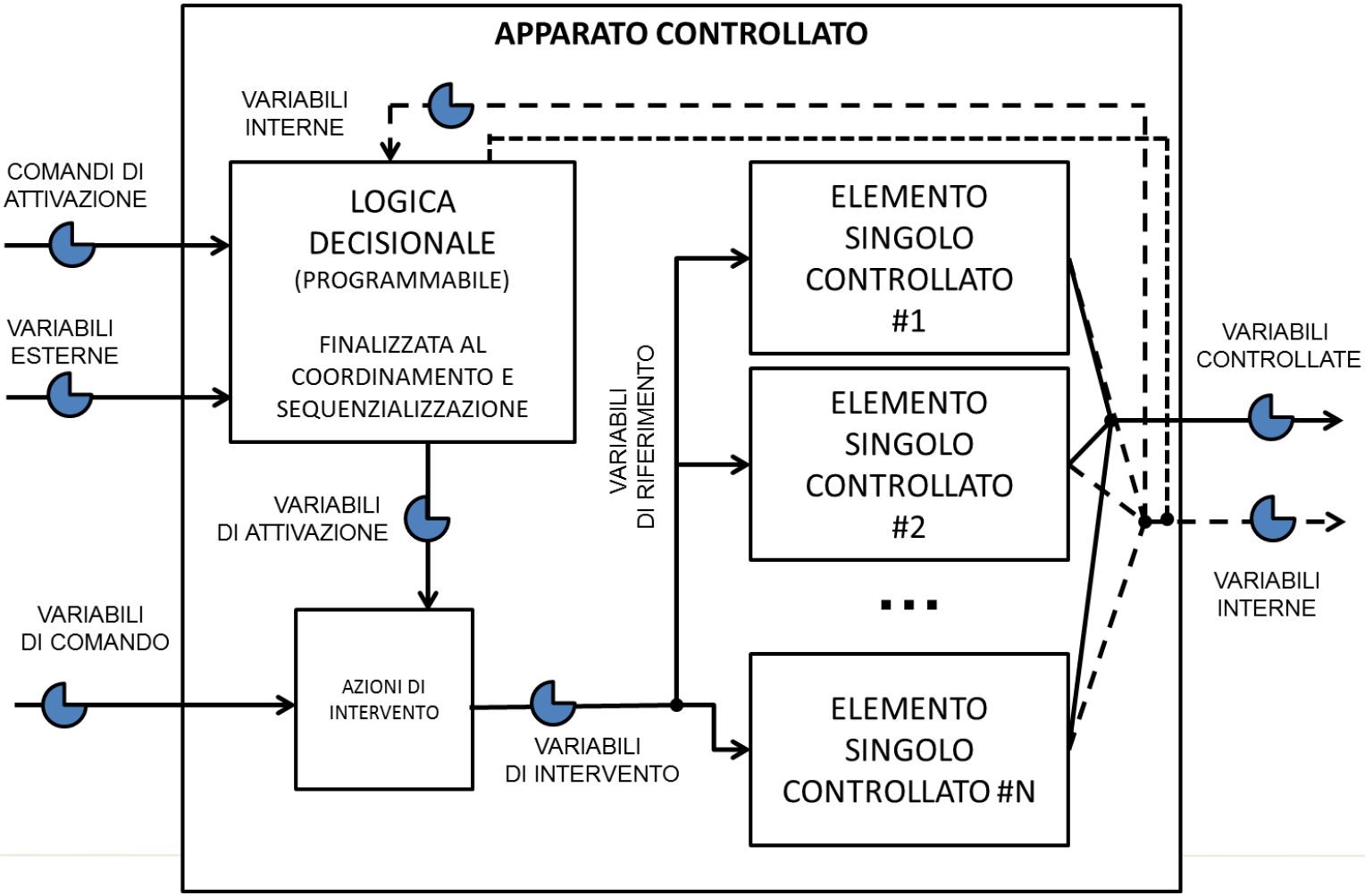
Attualmente, con lo sviluppo dei circuiti elettronici a larga integrazione e dei dispositivi di elaborazione digitale di tipo dedicato, i controllori logici sono realizzati con **tecnologie elettroniche**.

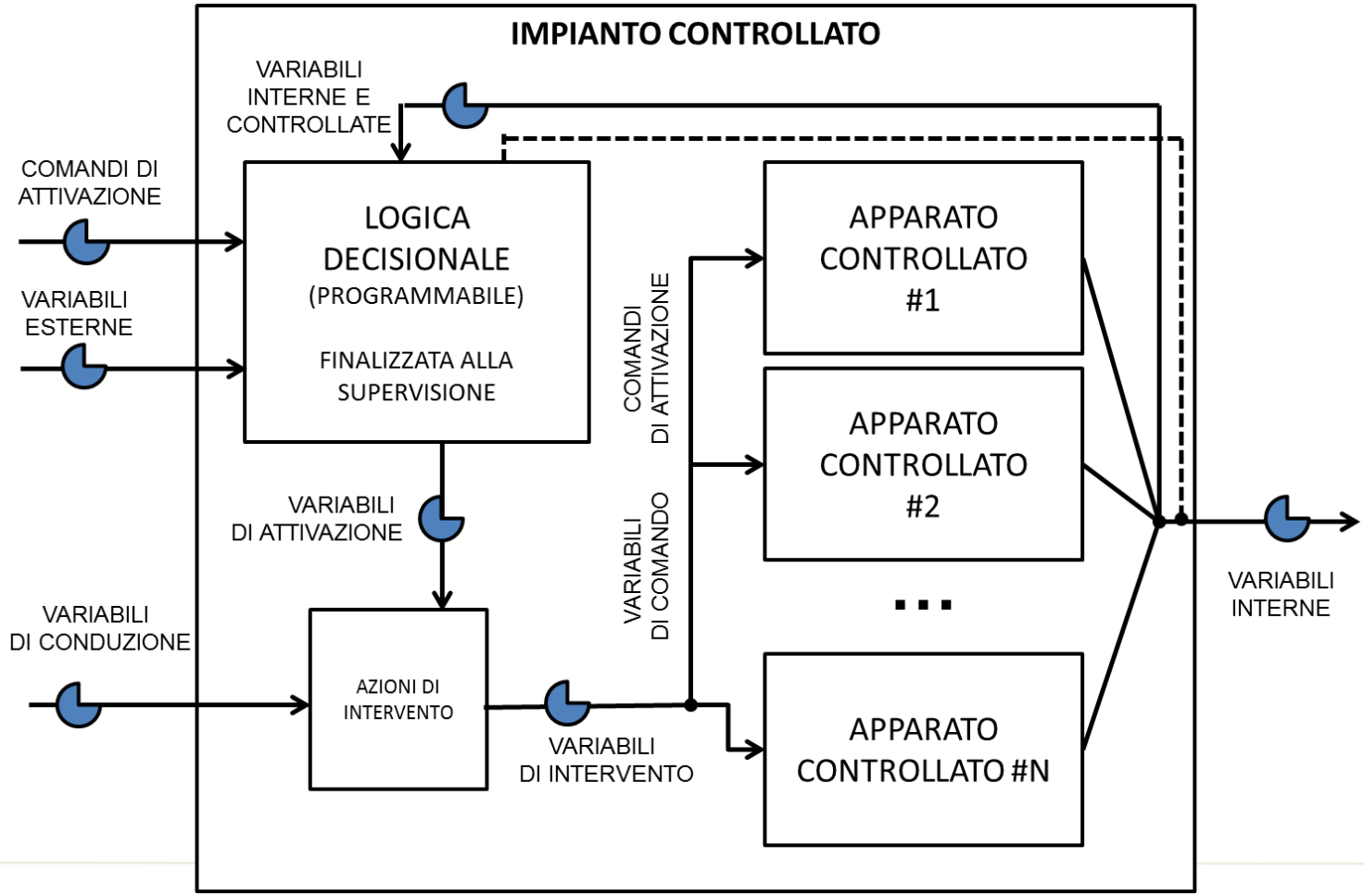
Sulla base della realizzazione i **controllori logici** possono essere di tipo:

- **cablato** quando l'elaborazione della logica di controllo è ottenuta impiegando relè e porte logiche opportunamente connesse (**reti logiche**)
- **programmabile** quando l'elaborazione è effettuata sulla base di un algoritmo di controllo espresso tramite un programma (**PLC**)

APPLICAZIONE DI RETI LOGICHE E DEI CONTROLLORI A LOGICA PROGRAMMABILE (PLC)

- Prima della applicazione delle azioni di intervento al sistema da controllare bisogna **verificare che sussistano tutte le condizioni che assicurino il corretto funzionamento** e il corretto impiego del sistema controllato.
- La verifica viene effettuata sulle **variabili di consenso**:
  - Comandi di attivazione
  - Variabili controllate
  - Variabili interne
  - Variabili esterne
- Sulla base delle informazioni ricevute, il programma deve fornire come risultato la decisione sotto forma di **variabili di attivazione**.





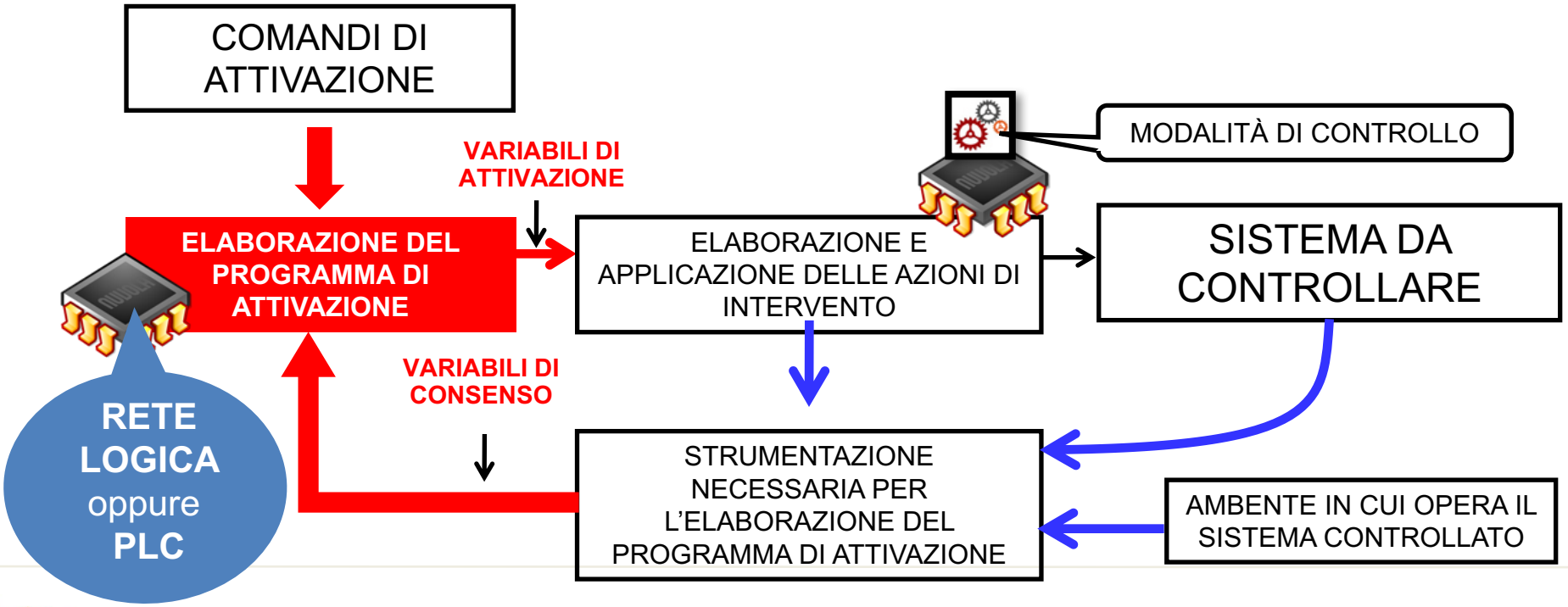
APPLICAZIONE DI RETI LOGICHE E DEI CONTROLLORI A LOGICA PROGRAMMABILE (PLC)

- Tutte le **informazioni** devono essere formulate in **logica binaria** e le elaborazioni collegate alle decisioni espresse in logica binaria.
- In molte applicazioni per la elaborazione, per la ricezione delle informazioni provenienti dalla strumentazione e per la trasmissione dei risultati della elaborazione agli attuatori vengono utilizzati o **reti logiche rigidamente programmate** oppure dispositivi in grado di effettuare le elaborazioni richiese sulla base di un **programma dedicato**. Questi ultimi dispositivi sono comunemente indicati come PLC, ossia **Controllori a Logica Programmabile**.
- La **differenza** sostanziale fra rete logica e controllori a logica programmabile sta nella **rapidità di elaborazione** e nella **flessibilità di programmazione**.

INSERIMENTO DI UNA RETE LOGICA o DI UN CONTROLLORE A LOGICA PROGRAMMABILE

Con il termine "*comandi*" sono indicati gli interventi:

- **diretti** effettuati da un operatore;
- **indiretti** ricavati da un programma di elaborazione.



DEFINIZIONE DI PLC - NORME IEC 61131.3

- Il PLC è un **sistema elettronico a funzionamento digitale**,
- destinato all'uso in **ambito industriale**,
- utilizza una **memoria programmabile** per l'**archiviazione** interna di istruzioni orientate all'utilizzatore;
- implementazione di funzioni **logiche**, di **sequenziamento**, di **temporizzazione**, di **conteggio** e **calcolo aritmetico**,
- **controlla**, mediante ingressi ed uscite sia digitali che analogici, vari tipi di sistemi semplici e/o complessi.

CARATTERISTICHE DI UN PLC

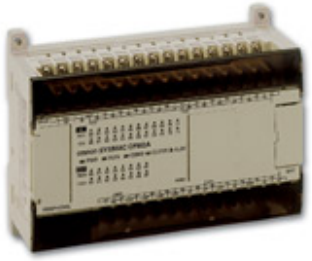
Il PLC è un controllore con **architettura general-purpose** dedicata alle elaborazioni di tipo logico e idonea ad un ambiente industriale.

Le principali caratteristiche:

- **Affidabilità** (ad es. 24/7, ridondanza x3, sicurezza certificata)
- **Espandibilità** (sostituzione/aggiunta nel rack di moduli)
- **Semplicità di programmazione** (tool sdk + gui, manuali)
- **Riusabilità della logica di programma** (linguaggi standard)
- **Interoperabilità tra dispositivi** di produttori diversi (i/f hw e sw)

CLASSIFICAZIONE DEI PLC

- Dal punto di vista costruttivo i plc sono classificati nella maniera seguente:
  - **μPLC**, quando gli **ingressi e le uscite** sono tutte digitali e **inferiori a 64** e la **memoria inferiore a 2 kbyte**;
  - **PLC** di medie dimensioni, quando gli **ingressi e le uscite** possono essere **digitali e analogiche**, in numero **inferiore a 512** e la memoria dell'ordine di **decine di kbyte**;
  - **PLC** di grandi dimensioni, quando i predetti limiti sono superati.
- I μPLC sono in genere monoblocco, gli altri componibili a moduli secondo le esigenze.



μPLC

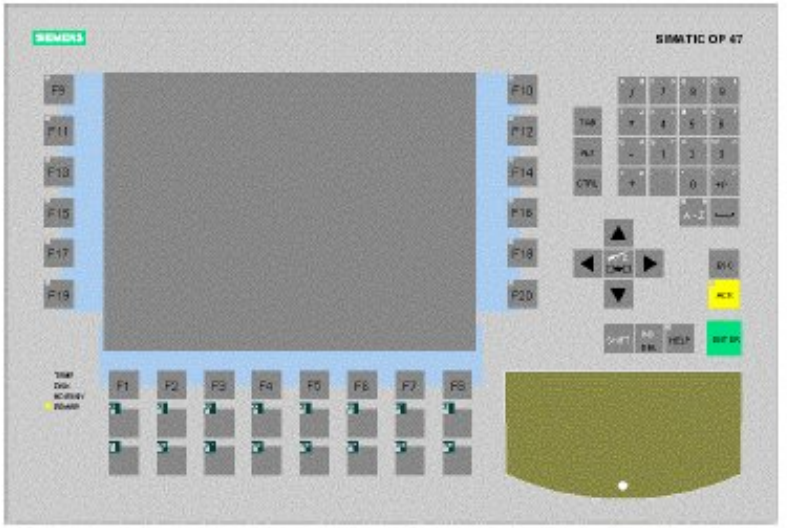
PLC MEDI

PLC GRANDI

STRUTTURA MODULARE DI UN PLC

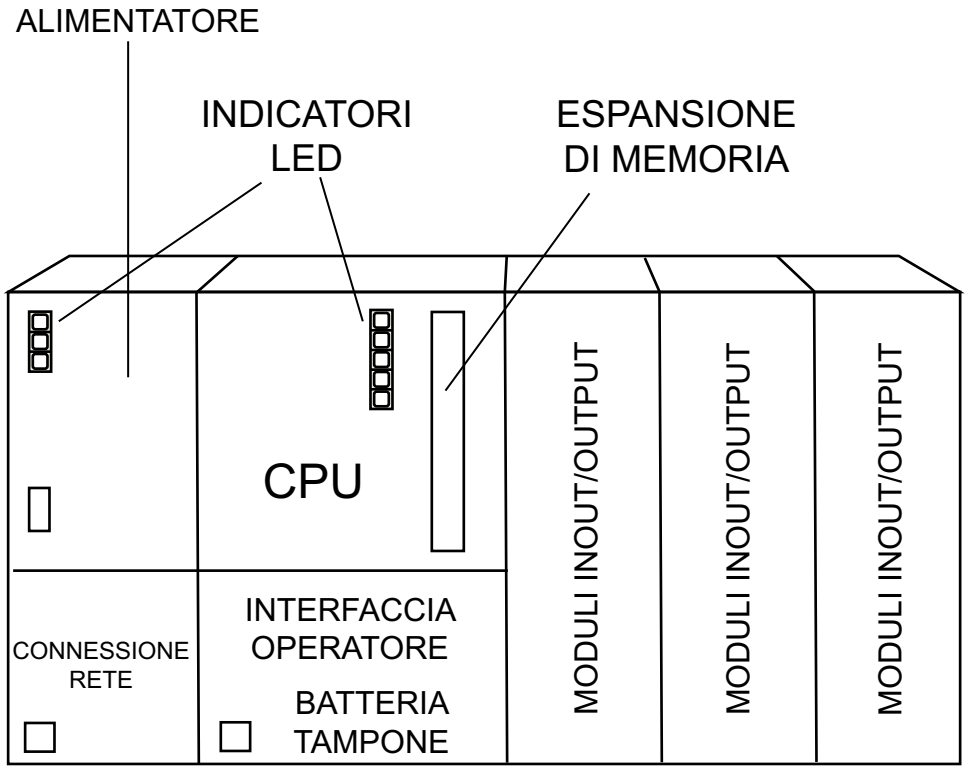
ESEMPIO DI INSERIMENTO DI UN PLC IN UN ARMADIO

ESEMPIO DI PANNELLO OPERATORE

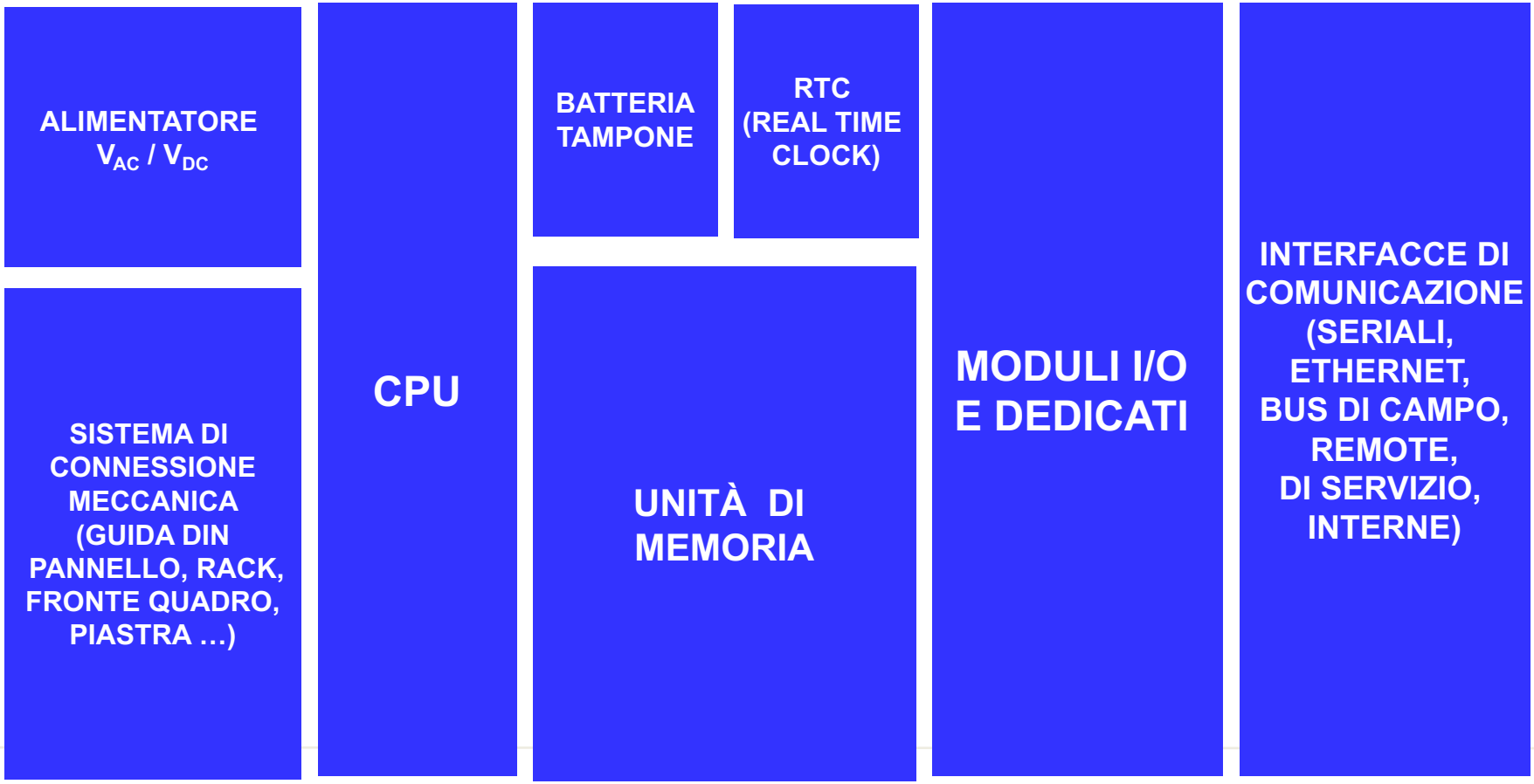


STRUTTURA DEI PLC

ASPETTO FISICO DI UN PLC



SCHEMA A BLOCCHI FUNZIONALI DI UN PLC



BLOCCHI FUNZIONALI DI UN P L C

- il **rack** con **alimentatore** e **bus di alimentazione e di trasmissione dei dati** relativi ai moduli che lo compongono + bus per l'alimentazione degli altri moduli e lo scambio di dati;
- il modulo di **elaborazione** (microprocessore + scheda madre)
- il modulo contenente la **batteria tampone** e la memoria *ram ad accesso rapido* necessaria per contenere i dati da elaborare;
- il modulo contenente la **memoria** non volatile (eeprom o similari) per contenere il programma e il salvataggio dei dati in condizioni di emergenza;
- i moduli di connessione ai **dispositivi di misura**;
- i moduli di connessione agli **attuatori**;
- i moduli di connessione alla **rete di telecomunicazione**;
- i moduli con **funzionalità prefissata** (contatori, regolatori PI+D, controlli assi)
- il modulo di **connessione con le periferiche operatore** (stato del PLC) →al PLC può essere collegato un pannello di visualizzazione per l'operatore

SEZIONE DI INGRESSO/USCITA

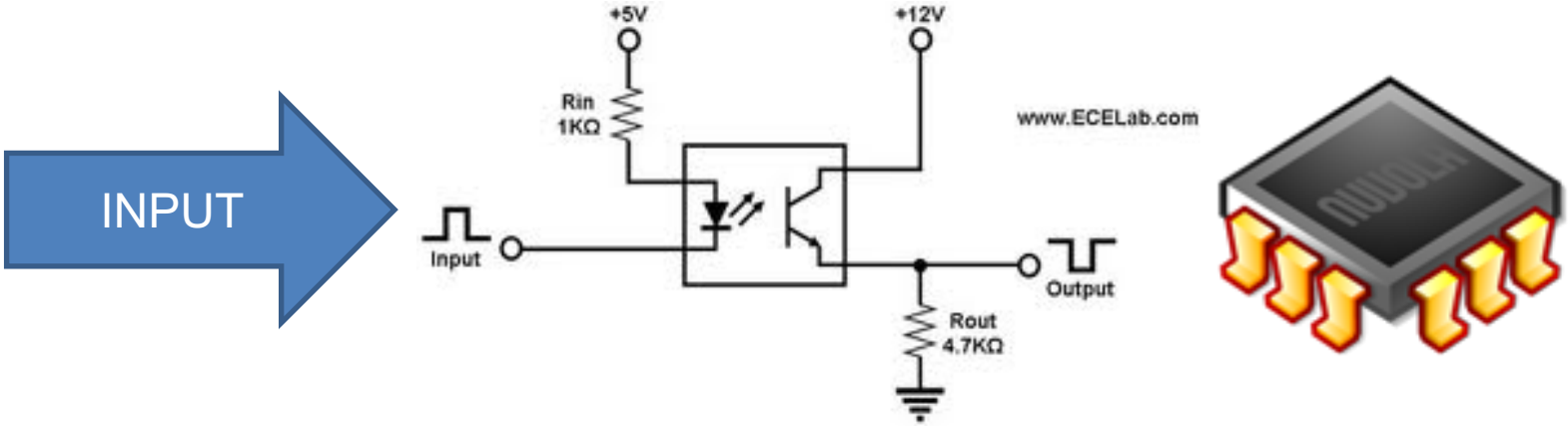
Le schede input/output specializzate sono:

- **regolatori** standard PI+D;
- schede per il **conteggio veloce** (lettura dell'uscita di un **encoder**);
- schede per la lettura e il controllo della **temperatura**;
- schede di lettura degli **estensimetri**;
- schede per il **controllo assi** - le schede controllo assi hanno la peculiarità che gli algoritmi da rendere operativi per realizzare una buona modalità di impiego del motore controllato sono in genere **sofisticati** e devono essere eseguiti con un **elevato passo di campionamento** e spesso hanno una **CPU dedicata**.

SEZIONE DI INGRESSO

Gli elementi utilizzati per un efficace interfacciamento con il sistema da controllare sono:

- stadio di isolamento realizzato in genere con **optoisolatori;**

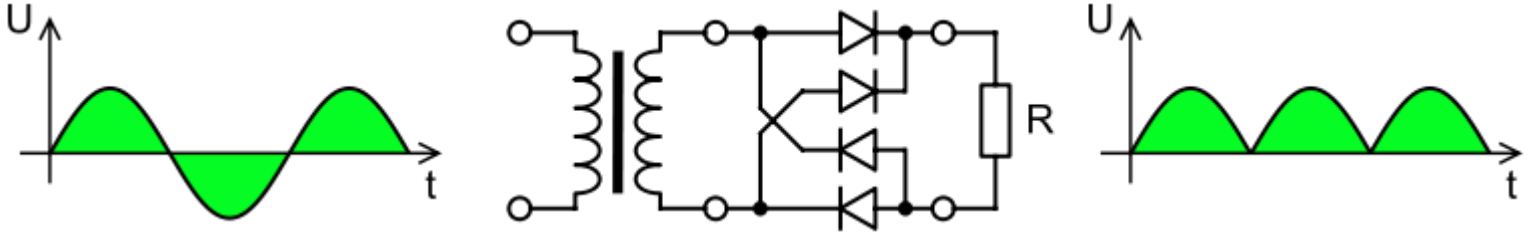


- utilizzazione di **contatti a vite** per velocizzare le procedure di installazione dei moduli;
- visualizzazione dello stato per il **debug** del programma di elaborazione (ad es. tramite **l'uso di LED**)

SEZIONE DI INGRESSO

- interfacciamento con i sensori adattato alle possibili caratteristiche del segnale di uscita, che può essere:
  - DC 5 - 12 - 24 - 48 v;
  - AC 110 - 280 v;

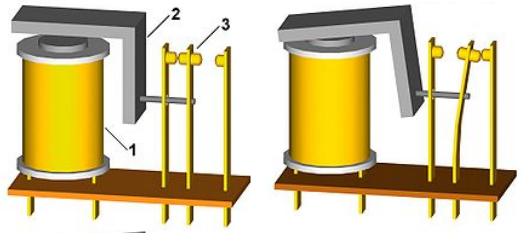
- per i segnali in alternata occorre uno stadio di **rettificazione** (ponte a diodi)



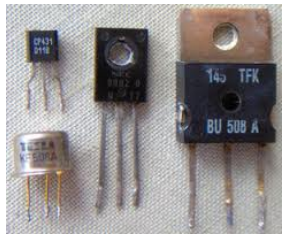
- e di **livellamento** (condensatori elettrolitici di livellamento);

SEZIONE DI USCITA

- **Relè** - componente elettromeccanico a solenoide, che permette di gestire elevate potenze con piccoli segnali di comando;



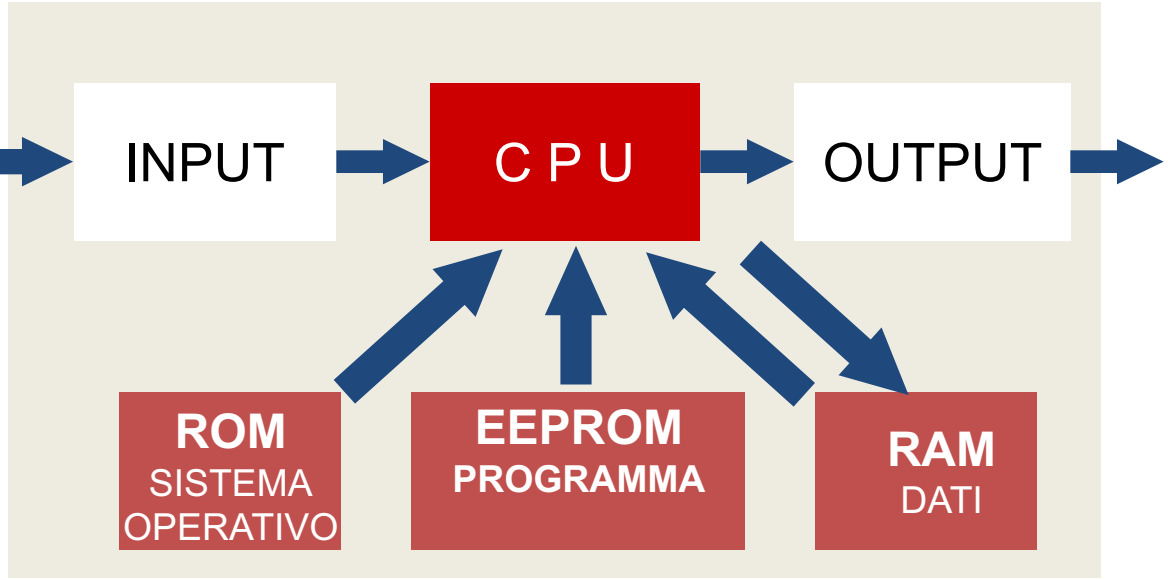
- **Triac** - è un relè allo stato solido che permette di gestire elevate potenze con piccoli segnali di comando e con limitata dissipazione di calore;



- **Transistor** – componente elettronico per amplificazione di piccoli segnali.

STRUTTURA DEI PLC
MEMORIA

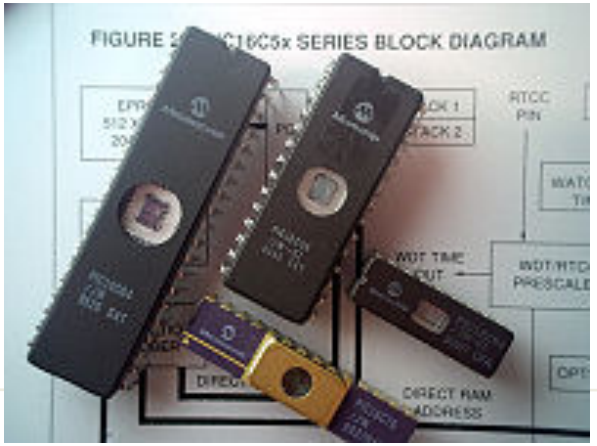
SCHEMA DELLA UNITÀ DI ELABORAZIONE DI UN PLC



ARCHITETTURA
CLASSICA

Esempio pratico

- PIC - Programmable Interface Controller



ORGANIZZAZIONE DELLA MEMORIA NEI PLC

La memoria RAM viene utilizzata per immagazzinare:

- i **dati** provenienti dal **sistema da controllare**;
- i **risultati intermedi** delle elaborazioni;
- i **dati** da inviare agli **attuatori**;
- **I dati dei programmi di supervisione** dedicati al controllo delle attività del PLC;
- i **dati** dei programmi di elaborazione dei **programmi utente**;
- i **dati** dei programmi di **comunicazione con altri PLC** o con l'apparato da controllare;
- i **dati** dei programmi di **diagnostica interna del PLC** stesso quali ad esempio il controllo di **parità** della memoria per la gestione degli errori e l'IRQ di **watchdog**.

ORGANIZZAZIONE DELLA MEMORIA NEI PLC

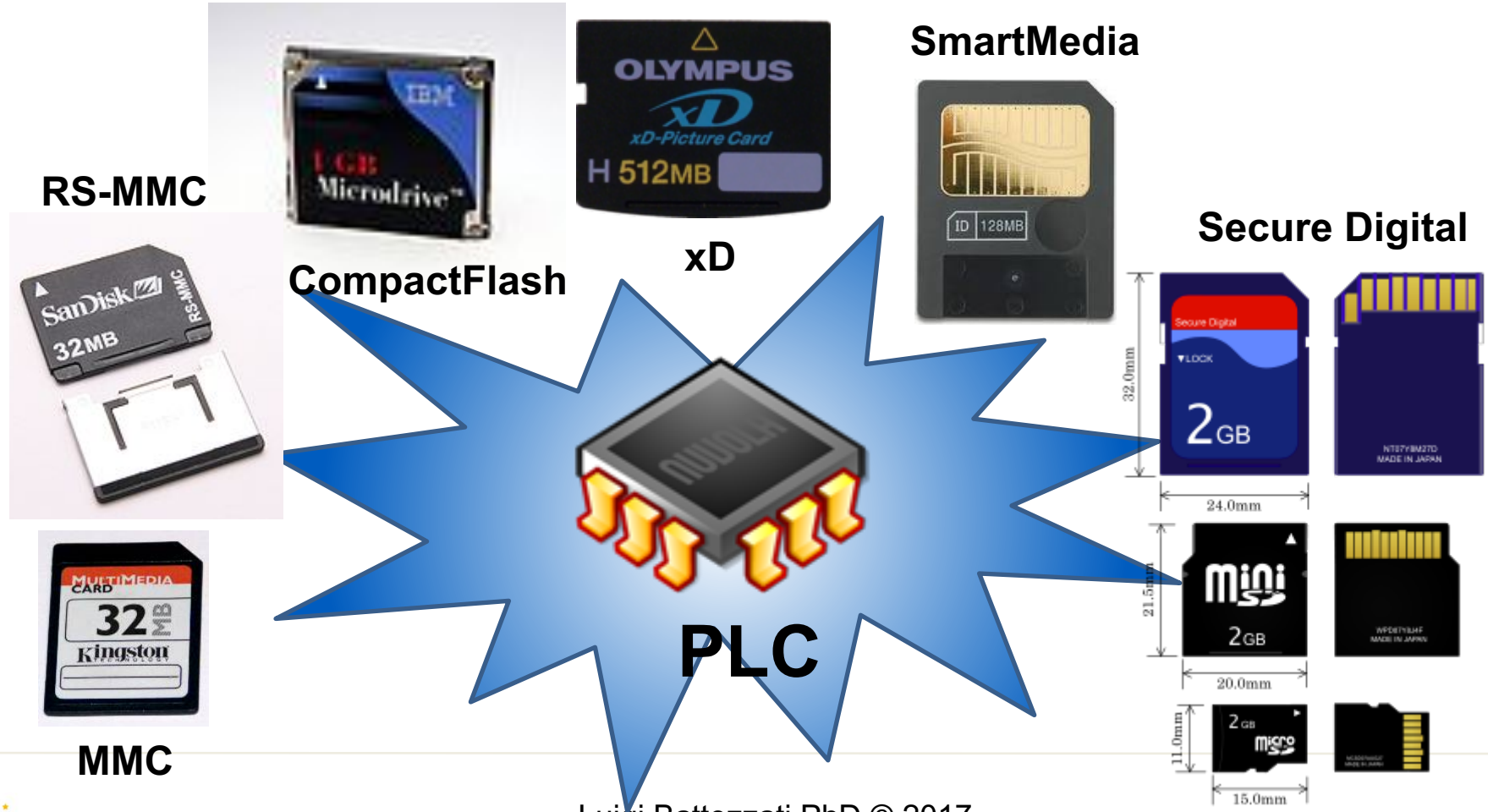
TIPOLOGIE DI MEMORIE **RAM** (RANDOM ACCESS MEMORY)

- La memoria RAM può essere letta e **scritta a blocchi** e non necessariamente in serie come la EEPROM.

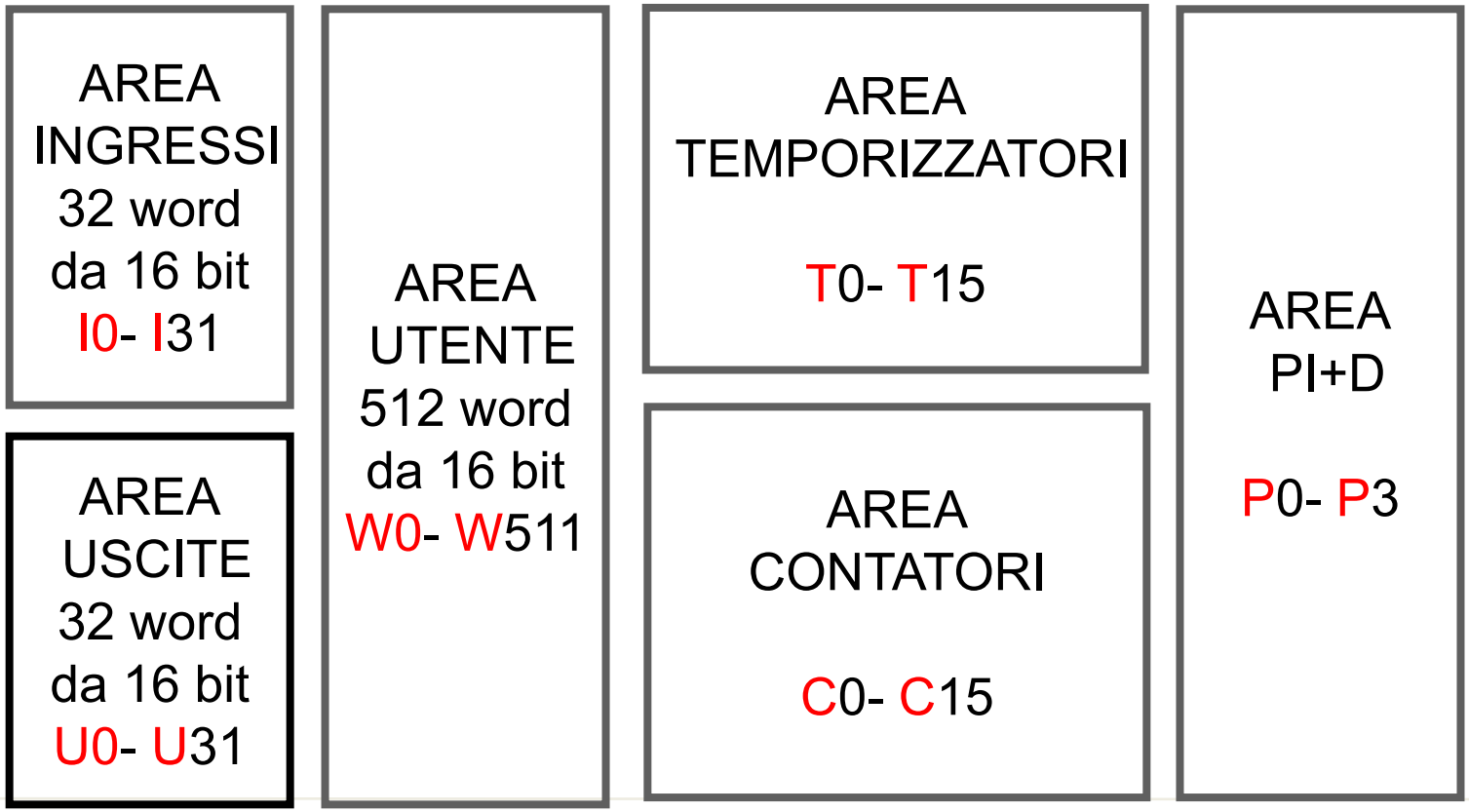
- La memoria **S**RAM (Static RAM) **una volta scritta entra in idle** e non richiede ulteriore alimentazione. Ma se non alimentata può perdere le informazioni immagazzinate pertanto NON è come una EEPROM. E' **molto veloce, consuma poco**, di semplice progettazione ma di bassa densità e quindi di **alto costo per Mbyte**.

- La memoria **D**RAM (Dynamic RAM) viene **alimentata periodicamente** per evitare la perdita di dati. E' discretamente veloce, è energivora, ma ha una **densità altissima** e quindi meno costosa per Mbyte della SRAM.

MEMORIE EEPROM RIMOVIBILI



ORGANIZZAZIONE DELLA MEMORIA RAM



ORGANIZZAZIONE DELLA MEMORIA RAM

AREA INGRESSI formata da 32 word da 16 bit indirizzabili da i0 a i31. Per ingressi di tipo digitale ad un bit, ciascun bit di ogni word può essere così indirizzato ix:y dove:

- i indica che la word è dell'area ingressi;
- x indica l'indirizzo della word (0-31)
- y indica il bit da indirizzare (0-15)

AREA USCITE formata da 32 word da 16 bit indirizzabili da u0 a u31 ciascun bit di ogni word può essere così indirizzato: ux:y dove:

- u indica che la word è dell'area ingressi;
- x indica l'indirizzo della word (0-31)
- y indica il bit da indirizzare (0-15)

N.B.: 1 nibble = 4 bit ; 1 byte = 8 bit ; 1 Word = 16 bit ; 1 Double Word = 32 bit

ORGANIZZAZIONE DELLA MEMORIA RAM

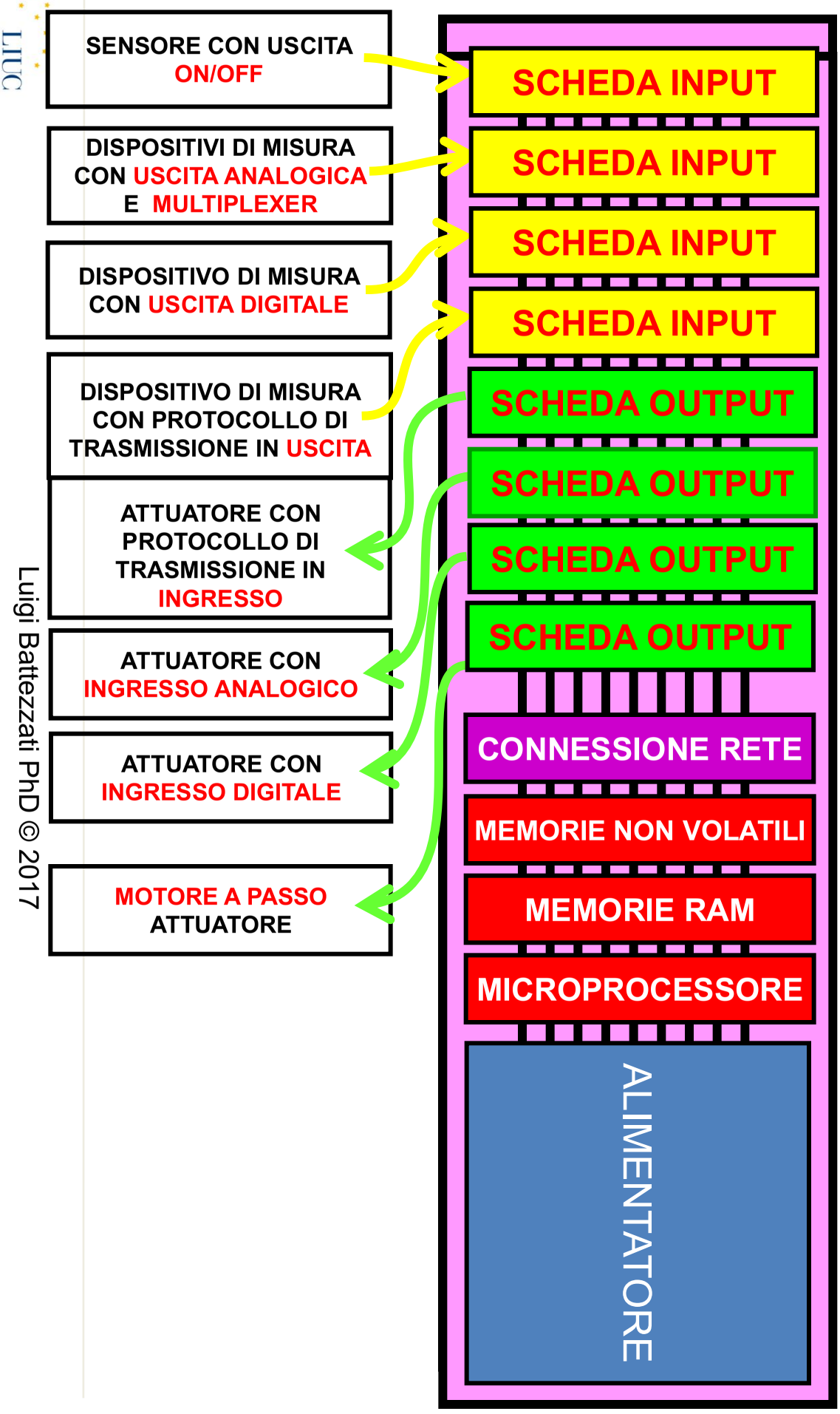
AREA UTENTE costituita da 512 word da 16 bit indirizzabili da W0 a W511
per ciascuna è possibile l'indirizzamento del singolo bit.

AREA TEMPORIZZATORI costituita da 16 word da 16 bit indirizzabili da T0 a T15

AREA CONTATORI costituita da 16 word da 16 bit indirizzabili da C0 a C15

AREA PI(D) riservata a 4 strutture PI+D indirizzabili da P0 a P3

STRUTTURA DEI PLC
SCHEDE DI I/O



ALIMENTATORE

L'alimentatore è utilizzato per fornire l'energia elettrica a tutte le schede del PLC. Fornisce le tensioni a +5V necessarie alle schede elettroniche, le tensioni a +/-12V, le altre tensioni necessarie, **sempre in corrente continua**. Può essere interno o esterno al PLC.

CPU

- La CPU è una scheda complessa basata su un **microprocessore con un sistema operativo proprietario**, e con una zona di memoria a disposizione del programma di automazione. Sono a 4 bit (logiche), 8 bit (somma), 16 bit (moltiplicazione) ma anche 32 e 64 bit (floating).
- La **memoria programmabile è spesso esterna** come ad esempio nel caso di memoria EEPROM. Il vantaggio di una memoria esterna è legata alla semplicità di programmazione o di modifica dello stesso.
- La CPU durante il funzionamento a regime, **colloquia con tutte le schede** connesse sul **BUS** del PLC, trasferendo dati e comandi sia verso il mondo esterno, sia dal mondo interno.

Una delle caratteristiche peculiari delle CPU dei PLC è la loro capacità di poter gestire le modifiche del programma di gestione del processo durante il normale funzionamento (**on-the-fly programming**). Questa possibilità è estremamente utile nel caso di impianti che devono essere sempre attivi.
All'interno della CPU sono varie parti, tra cui

- **unità di gestione**, ovvero informazioni di gestione del PLC stesso, impostate dal costruttore e trasparenti all'utente;
- archivio di **temporizzatori e contatori** funzionali all'operatività del PLC;
- **memorie di stato**, cioè le informazioni in ingresso ed i comandi in uscita dal processo;
- **memoria utente**, in cui vengono scritti i programmi che il PLC deve eseguire;
- **interfaccia** per il dispositivo di programmazione, che comunica con gli strumenti di programmazione;
- **bus dati**, (comando + indirizzi) per la veicolazione dei dati fra le varie parti e con l'esterno della CPU.

SCHEDE DI INGRESSO DIGITALI

Le schede di ingresso digitali sono utilizzate per il **monitoraggio di grandezze fisiche rappresentate in forma digitale**, cioè di **tensioni a due valori** (ad esempio 0-5V, oppure 0-24V). Ogni scheda può gestire 2^N (2,4,…,64,ecc.) ingressi digitali differenti. I segnali dal campo vengono fatti arrivare con **cavi elettrici** fino alla **morsettiera** della scheda ed ogni singolo canale è opportunamente protetto da **fusibili** di adeguato amperaggio.

SCHEDE DI USCITA DIGITALI

Le schede di uscita digitali sono utilizzate per i **comandi di attuatori digitali**. Ad esempio un **relè** è un attuatore digitale, in quanto può avere soltanto due stati stabili: diseccitato, o eccitato. Altro esempio di attuatore digitale è una **valvola a due stati**: aperta o chiusa (ad es. **elettrovalvola**). Anche nel caso di schede di uscita digitali, si possono gestire un numero 2^N di uscite.

SCHEDE DI INGRESSO ANALOGICHE

- Questo tipo di schede di ingresso permettono il monitoraggio di **grandezze elettriche** il cui valore può variare entro un **intervallo**.
- Le grandezze in gioco sono **in tensione o in corrente**. Ad esempio sono disponibili schede di ingresso analogiche in corrente, con un intervallo variabile tra 4 e 20 mA.
- Molti produttori di PLC rendono disponibili schede con ingressi analogici:
  - **Termoresistenze** Pt100/Pt1000

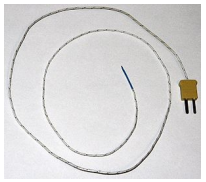
$$\rho(T) = \rho_0 \cdot [1 + \alpha(T - T_0)]$$

termo-resistenze in platino (Pt), in cui la resistenza alla temperatura di 0 °C è pari rispettivamente a 100 Ω e 1000 Ω.

EFFETTO Seebeck

CHROMEL
ALUMEN

  - **Termocoppie** T, J, K, ecc.



ECONOMICHE
NON LINEARI
POCO ACCURATE (+/- 1°C)

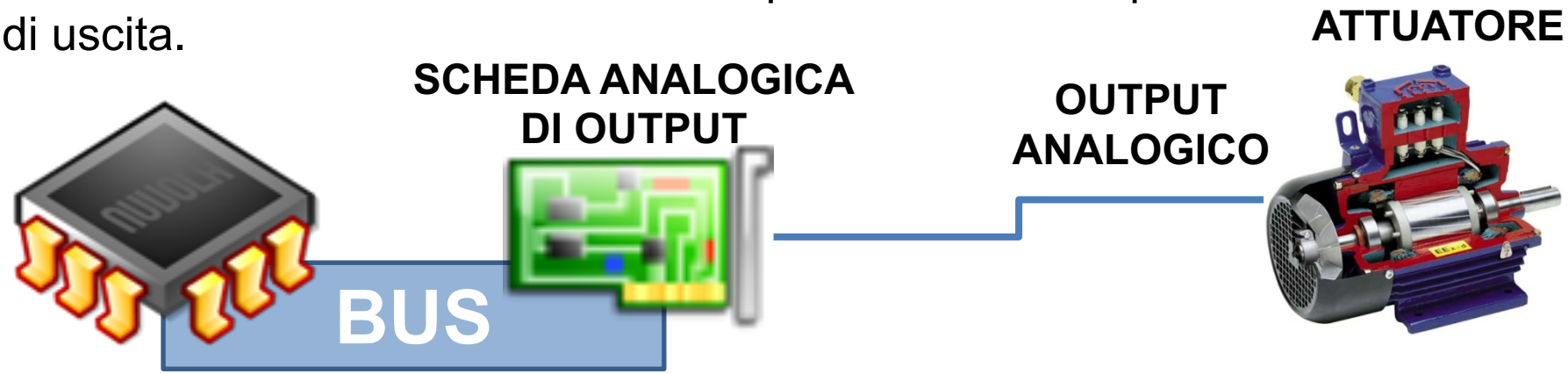
RAME COSTANTANA

FERRO COSTANTANA

- Queste schede sono disponibili con varie **risoluzioni** (8-12-16 bit) e con 1 o più (2^N) ingressi distinti disponibili in morsettiera o con connettore.

SCHEDE DI USCITA ANALOGICHE

- Le schede di uscita analogiche permettono di controllare degli **attuatori di tipo continuo**.
- Possono essere modulate **in corrente o in tensione** ed avere una determinata risoluzione esprimibile in bit.
- Ad esempio è possibile comandare la velocità di un motore elettrico tramite un **inverter** CA-CA operando sulla frequenza di uscita.



- Possono servire anche per regolazioni di temperatura o regolazioni di luce.

SCHEDE DI COMUNICAZIONE

- Il PLC durante il suo funzionamento può comunicare con altri PLC, computer o dispositivi CNC (**computer numerical control**) come presse piegatrici, punzonatrici, torni, fresatrici e macchine di taglio lamiera;
- La **comunicazione con computer** e altri dispositivi avviene tramite tipi di connessione standard come:
  - RS232 – Seriale
    Recommended Standard
  - RS422/RS485 – Seriale
    Recommended Standard
  - TCP/IP (RJ45) o USB

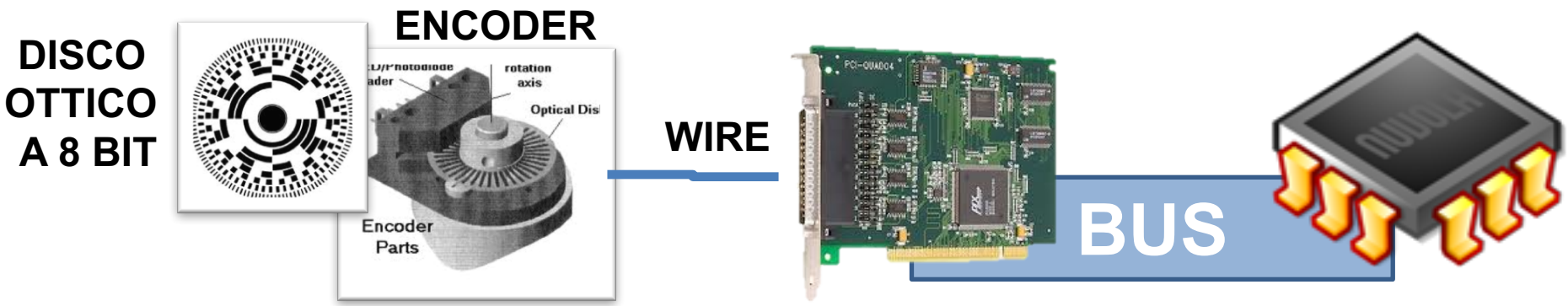
- La **comunicazione con altri PLC** avviene tramite protocolli standard, ad esempio:
  - Profibus - Modbus - CANBUS - ecc.

SCHEDE DEDICATE

- Qualunque PLC di livello medio alto, oltre le consuete schede di ingresso/uscita, analogiche/digitali, ha a catalogo moduli dedicati a **particolari compiti di automazione**.
- Il vantaggio nell'utilizzare tali schede è quello di avere il **controllo** di un'operazione/evento **indipendentemente dal ciclo del PLC**, relegando il PLC alla funzione di controllo/parametrizzazione.
- L'offerta è veramente vasta e ogni produttore propone a catalogo le più svariate soluzioni, fra cui si segnalano:
  - **Contatori;**
  - **PID;**
  - **Controllo assi.**

SCHEDE DI CONTEGGIO

- Accolgono il segnale di un **sensore di conteggio e direzione** più un canale di **azzeramento**. Il cablaggio funziona in **single ended** (Ground + Segnale, come nella RS232, non robusto al rumore) che in **differenziale** (normalmente secondo lo standard RS-422);



- Normalmente è possibile programmarle in modo che scatenino un evento (per esempio alzando un'uscita) al raggiungimento di una **soglia** o all'interno di un **intervallo** di valori.

SCHEDE PI+D

- Sono schede ad **un ingresso ed una uscita** ed applicano un anello di controreazione locale.

Sono utili per regolare temperature, pressioni, tensioni, correnti, etc.

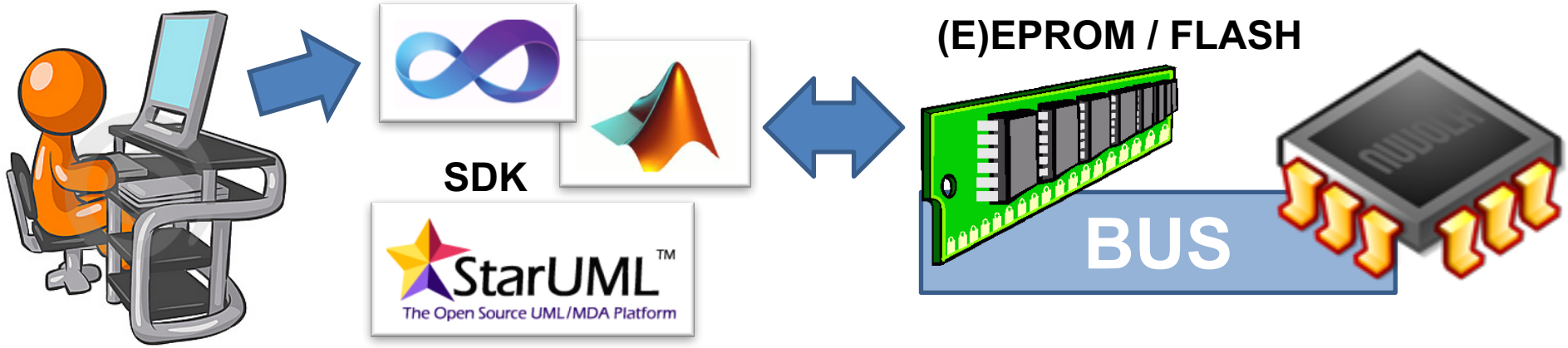
SCHEDE CONTROLLO ASSI

- Si impiegano ove sia necessario controllare il movimento di un organo meccanico tramite un **motore brushless o passo passo**. Alcune schede presentano un funzionamento particolarmente semplice permettendo di fissare una **quota di consegna** che l'asse deve raggiungere e un ingresso per il **feedback di posizione**. Altre permettono grandissima flessibilità e permettono di emulare diversi profili. Richiedono generalmente un modulo di potenza esterno (amplificatore di corrente) per il comando effettivo del profilo desiderato al motore.

PROGRAMMAZIONE DEI PLC

PROGRAMMAZIONE DI UN PLC

- Il PLC per ottemperare ai suoi compiti deve essere **programmato**.
- La programmazione del PLC è effettuata normalmente con un PC sul quale un software specializzato (**SDK - Software Development Kit**) permette di creare programmi da scaricare nella memoria della CPU del PLC.



- Tali software possono leggere il programma direttamente dalla memoria della CPU, e **visualizzare il programma sul PC**.
- Normalmente il programma può essere **testato** in un ambiente di **simulazione** o di **realtà virtuale** per essere validato.

MODALITÀ DI ESECUZIONE DI UN PROGRAMMA

Un PLC esegue secondo una **modalità ciclica un programma utente** scritto in uno dei linguaggi definiti dalle norme IEC.
La sequenza delle operazioni è la seguente:

- **lettura degli ingressi** e scrittura del loro contenuto in una particolare locazione di memoria (buffer di lettura). Le variabili di ingresso possono provenire da sensori di varia natura, (on/off, analogici, digitali) o da altri PLC;
- **esecuzione del programma**. Le istruzioni vengono eseguite una dopo l'altra, procedendo dall'alto verso il basso, con operandi prelevati dalla memoria e risultati conservati in locazioni di memoria riservate;
- **scrittura delle uscite**. I risultati delle elaborazioni vengono inviati a locazioni di memoria particolari.

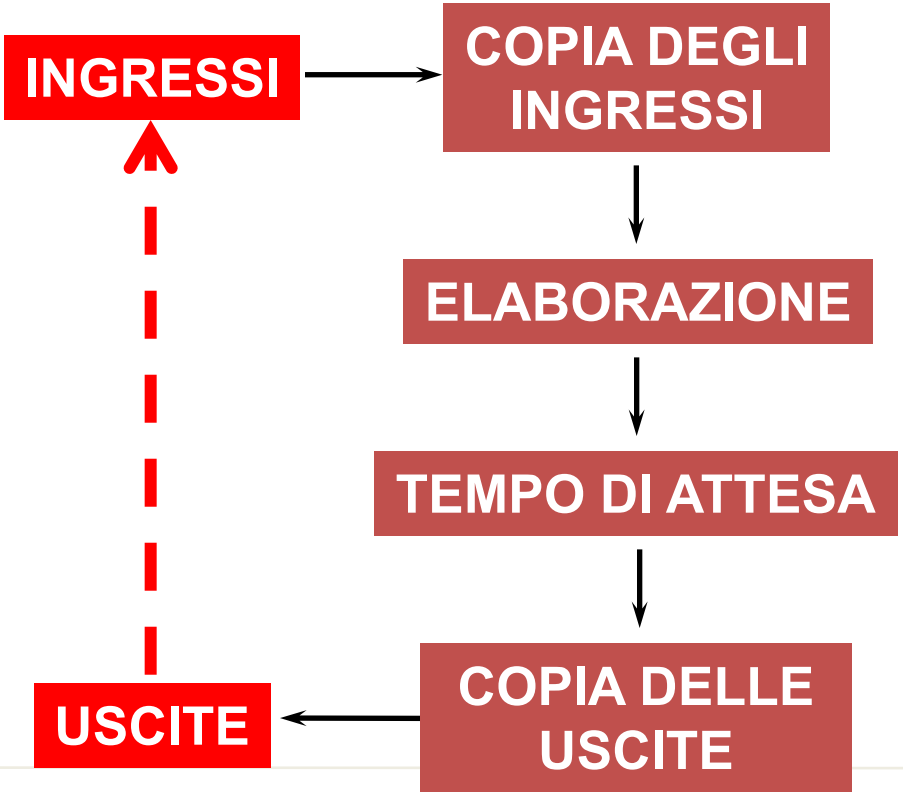
I dati in uscita possono essere **segnali di comando di un attuatore** o oppure **dati da scambiare** con altri PLC.

Il comando di inizio e di termine dell'esecuzione di un programma deve essere inviato dall'esterno.

ELABORAZIONE DI UN PROGRAMMA

La modalità di elaborazione di un programma è affidata al sistema operativo del PLC ed è effettuata in maniera **ciclica** e completamente **automatica**.
In ogni ciclo si susseguono i seguenti stati:

1) inizio di un ciclo di elaborazione
2) lettura degli ingressi e trasferimento del loro valore nell'area di memoria dedicata
3) elaborazione degli ingressi per il calcolo dei risultati, che costituiscono le variabili di uscita
4) inserimento di un opportuno tempo di attesa per ottenere che le variabili di uscita siano disponibili all'istante desiderato per realizzare la modalità di controllo real-time
5) trasferimento del valore delle variabili di uscite in una area di memoria dedicata
6) fine del ciclo di elaborazione e inizio di un nuovo ciclo



CICLO DI ESECUZIONE DI UN PROGRAMMA

TEMPO DI ESECUZIONE DI UN SINGOLO CICLO:

- lettura ingressi
- esecuzione programma
- tempo di attesa
- aggiornamento uscite
- Inoltro delle uscite in rete

SOFTWARE DI PROGRAMMAZIONE DEI P L C

SECONDO LE NORME IEC 61131-3

LINGUAGGI GRAFICI

LD - LADDER DIAGRAM
FBD - FUNCTION BLOCH DIAGRAM
SCF - SEQUENTIAL FUNCTION CHART

LINGUAGGI TESTUALI

IL - INSTRUCTION LIST
ST - STRUCTURATED TEXT

Nella stesura del programma in uno di tali linguaggi viene utilizzato un software dedicato allo sviluppo, alla validazione e alla rappresentazione grafica utilizzando i propri simboli grafici e il proprio insieme di istruzioni.

LINGUAGGI DI PROGRAMMAZIONE

SECONDO LE NORME IEC 61131

LINGUAGGI DI PROGRAMMAZIONE GRAFICI

LADDER DIAGRAM
SCHEMA A CONTATTI

ottenuto come **trasposizione** informatica dei **quadri a relè**.

FUNCTIONAL BLOCK DIAGRAM

ottenuto come **trasposizione** dei **diagrammi circuitali** in cui le interconnessioni rappresentano i **percorsi dei segnali** che collegano i vari componenti. I blocchi rappresentano le singole operazioni logiche.

SEQUENTIAL FUNCTIONAL CHART

ottenuto applicando un formalismo grafico per la descrizione di **operazioni logiche sequenziali** e formalismi grafici proprio di altri linguaggi di programmazione. utilizzato per descrivere in maniera orientata alla **progettazione sistemi complessi di automazione**.

LINGUAGGI DI PROGRAMMAZIONE

SECONDO LE NORME IEC 61131

LINGUAGGI DI PROGRAMMAZIONE TESTUALI

INSTRUCTION LIST

linguaggio di programmazione di **basso livello** molto simile all'**ASSEMBLER**. Le istruzioni sono costituite da un operatore e da un solo operando e fanno riferimento ad un registro di memoria. I formalismi adottati possono essere molto differenti in quando fissati dal produttore dell'hardware per il PLC.

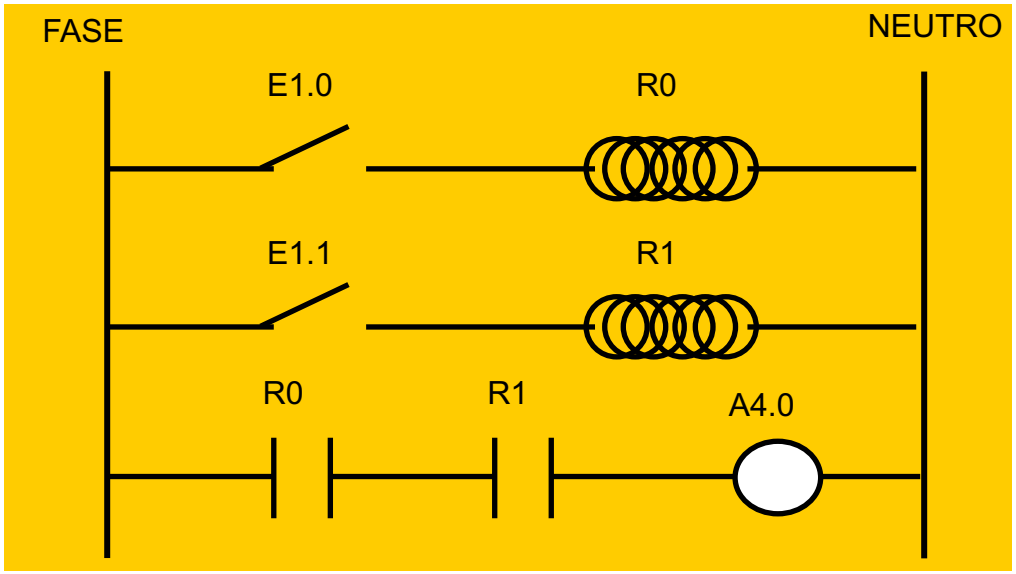
STRUCTURED TEXT

linguaggio di programmazione strutturato ad **alto livello** con un formalismo che si ispira al **BASIC** e al **PASCAL**. È adatto alla rappresentazione di procedure complesse che non potrebbero essere descritte con i linguaggi grafici.

LINGUAGGIO LADDER

Linguaggio grafico per tecnici esperti di **quadri a relè**.

Esprimibile in termini di **relè** o di **porte logiche**;



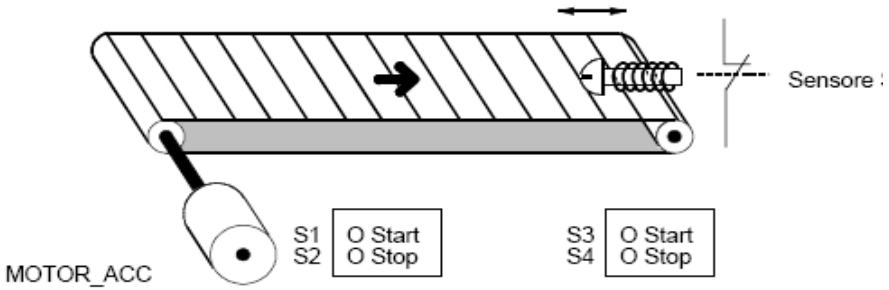
Nella figura in alto i primi due gradini realizzano la sezione di acquisizione dei segnali in ingresso sui relè R0, R1 mentre il terzo gradino realizza l'equazione logica di comando di una specifica uscita (in questo caso l'and)

DIAGRAMMA FUNZIONALE A BLOCCHI (FBD)

Il FUNCTION BLOCK DIAGRAM è un linguaggio grafico nel quale il programma di elaborazione è modellato come un **flusso di dati e di segnali** attraverso gli elementi di elaborazione (function block). Può essere visto come **l'analogo dei diagrammi circuitali,** in cui le connessioni rappresentano i percorsi dei segnali tra i componenti.

Un **blocco funzionale** ha due caratteristiche principali, la **definizione dei dati** (ingressi e uscite) e l'**algoritmo** che realizza le elaborazioni da effettuare sul valore degli ingressi e delle variabili interne (locali o globali) e produce i nuovi valori delle variabili di uscita.

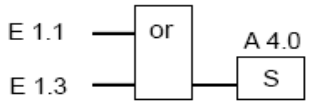
ESEMPIO DI DIAGRAMMA FUNZIONALE A BLOCCHI (FBD)



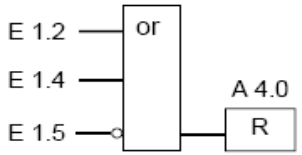
Componente del sistema	Indirizzo assoluto	Simbolo
Pulsante Start	E 1.1	S1
Pulsante Stop	E 1.2	S2
Pulsante Start	E 1.3	S3
Pulsante Stop	E 1.4	S4
Sensore	E 1.5	S5
Motore	A 4.0	MOTORE_ACC

Schema logico per il controllo del nastro trasportatore

Segmento 1: premendo uno dei due pulsanti Start si aziona il motore.



Segmento 2: premendo uno dei due pulsanti Stop o aprendo il contatto normalmente chiuso posto alla fine del nastro trasportatore si spegne il motore.

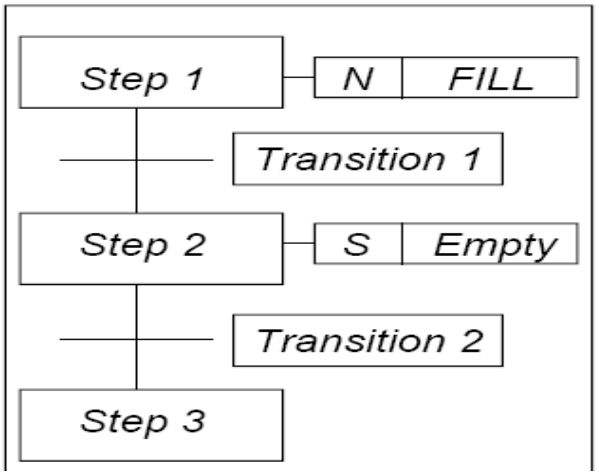


SFC - SEQUENTIAL FUNCTION CHART

Nel SEQUENZIAL FUNCTION CHART sono rappresentati gli elementi chiave di una procedura sequenziale, ossia le **condizioni per passare da uno stato ad un altro** e gli **effetti** (uscite fisiche) presenti mentre si trova in uno stato particolare.

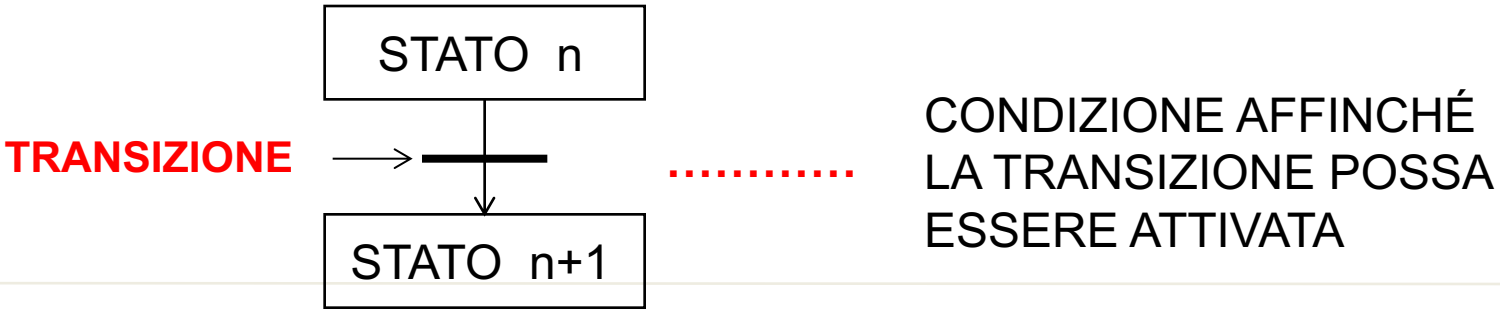
Risulta utile per partizionare un problema di controllo.

Mostra una visione di insieme utile per una rapida diagnostica.



DEFINIZIONE DI:

- **STATO** (FASE, TAPPA, PASSO)
  - descrive **l'evoluzione temporale** del funzionamento di un sistema complesso mediante una successione temporale di situazioni **operative più semplici (fasi)**, nelle quali è attivo solo un **sottoinsieme degli ingressi e delle uscite**
  - Associa ad ogni stato un **algoritmo di controllo** diverso da quelli associati agli altri stati
- **TRANSAZIONI**
  - rappresenta il **passaggio da uno stato ad un altro stato**
  - associa ad ogni transizione la **condizione che deve essere verificata affinché possa avvenire la transizione**



Ciclo di copia massiva

- Il PLC esegue periodicamente il ciclo di copia massiva degli ingressi e delle uscite, articolato nei seguenti passi:
  - Lettura degli ingressi fisici e aggiornamento coi valori così ottenuti di un'area specifica della memoria (registro immagine degli ingressi)
  - Esecuzione sequenziale del programma utente, che pone i risultati in un'altra area di memoria (registro immagine delle uscite)
  - Esecuzione dei programmi di gestione del sistema (ad esempio di diagnostica)
  - Scrittura sulle uscite fisiche dei valori corrispondenti conservati nell'area di memoria

- L'esecuzione è quindi ciclica: il tempo che il PLC impiega per una singola elaborazione viene chiamato tempo di ciclo (in genere da 10 a 100 ms).
- Il PLC non ha modo di accorgersi di variazioni degli ingressi durante il ciclo.
- Inoltre un ingresso non viene acquisito con un intervallo di campionamento costante.

Linguaggi di programmazione

La normativa IEC 1131-3 definisce cinque linguaggi di programmazione per i PLC, di cui tre grafici e due testuali.

Linguaggi grafici:

diagramma funzionale sequenziale	(**SFC**, **S**equential **F**unctional **C**hart),
linguaggio a contatti	(**LD**, **L**adder **D**iagram),
diagramma a blocchi funzionali	(**FBD**, **F**unction **B**lock **D**iagram).

Linguaggi testuali:

lista di istruzioni	(**IL**, **I**nstruction **L**ist),
testo strutturato	(**ST**, **S**tructured **T**ext).

Vi sono oggi diversi ambienti di sviluppo che supportano più di un linguaggio di programmazione.

Ladder diagram

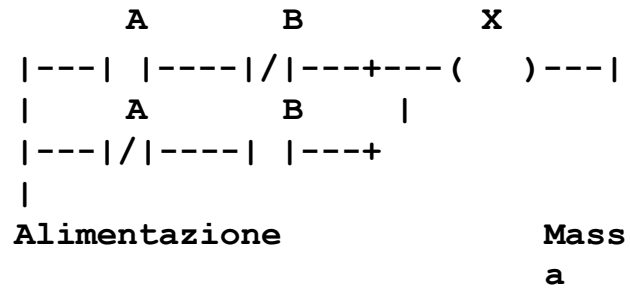
- Il Linguaggio a Contatti o Diagramma a Scala (Ladder Diagram, LD) è derivato dai disegni dei sistemi di controllo realizzati con relé elettromeccanici

- Si basa sui concetti di contatto e bobina ed è stato inizialmente pensato per funzioni di logica binaria; poi è stato esteso per trattare anche numeri interi e/o reali.

- È un linguaggio di basso livello e poco strutturato, non molto adatto a sistemi complessi. Tuttavia è importante perché è stato il primo linguaggio grafico per PLC, è presente in tutti i PLC industriali ed è uno standard di fatto del mercato americano.

Ladder diagram: concetti fondamentali

- Un ladder diagram si compone di:
  - due linee verticali dette “montanti”: il montante di sinistra si intende connesso all’alimentazione, quello di destra a massa
  - alcuni collegamenti orizzontali tra i montanti, detti “pioli” o “rung” che contengono a sinistra dei contatti e a destra delle bobine
  - la “corrente” può fluire solo da sinistra verso destra
  - a ciascun contatto e a ciascuna bobina sono associate delle variabili logiche
  - i pioli vengono esplorati dal primo in alto fino all’ultimo in basso
  - se c’è continuità elettrica tra il montante di sinistra e la bobina, la variabile logica associata alla bobina risulta 1 (true), altrimenti è 0 (false)



Ladder diagram: elementi fondamentali

`--| |--`: contatto normalmente aperto (se il bit associato è 1 il contatto è chiuso)

`--|/|--` : contatto normalmente chiuso (se il bit associato è 1il contatto è aperto)

`--( )--`: bobina normale (se alimentata, assegna al bit associato il valore 1)

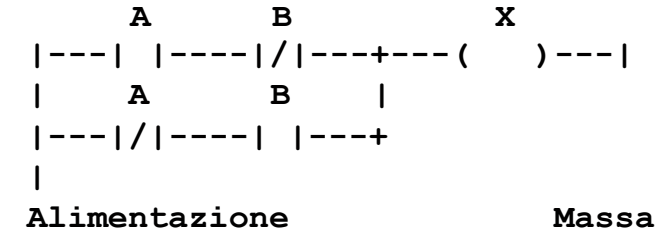
`--(L)--`: bobina latch (se alimentata, assegna al bit associato il valore 1, che rimane tale finché non si alimenta una bobina unlatch associata allo stesso bit)

`--(U)--`: bobina unlatch (se alimentata, assegna al bit associato il valore 0, che rimane tale finché non si alimenta una bobina latch associata allo stesso bit)

`--|P|--`: contatto a riconoscimento di fronte positivo (se il bit associato passa da 0 a 1 il contatto si chiude per un ciclo)

`--|N|--`: contatto a riconoscimento di fronte negativo (se il bit associato passa da 1 a 0 il contatto si chiude per un ciclo)

Esempio di codice LD (funzione **`X = A xor B`**)



Ladder diagram: altri elementi

Temporizzatore (normale):

```
   +-----------+
   | T         |
---| NomeTemp  |---
   | Durata    |
   +-----------+
```

- Se al temporizzatore giunge corrente (da sinistra) esso conta il tempo fino a raggiungere la durata impostata
- A questo punto la variabile NomeTemp va a 1 e così rimane fino al reset del temporizzatore, che si ha quando cessa la continuità elettrica verso di esso.
- In ogni altro caso, NomeTemp vale 0.

Temporizzatore (a ritenuta):

```
   +-----------+
   | TR        |
---| NomeTemp  |---
   | Durata    |
   +-----------+
```

- Analogo al precedente
- Quando cessa la continuità elettrica verso di esso non si resetta, bensì ferma il conteggio del tempo al valore raggiunto in quel momento.
- Quando la continuità ritorna, il conteggio riprende quindi dal valore raggiunto in precedenza
- Per resettarlo, si usa un apposito comando

```
NomeTemp
---(RES)---
```

Contatore:

```
   +-----------+
   | CU        |
---| NomeCont  |---
   | NumFronti |
   +-----------+
```

- Analogo al temporizzatore a ritenuta
- Vengono contati i fronti di salita, da 0 a 1, dell'ingresso, fino al numero NumFronti
- Per resettarlo, si usa un apposito comando:

```
NomeCont
---(RES)---
```

Ladder diagram: salti

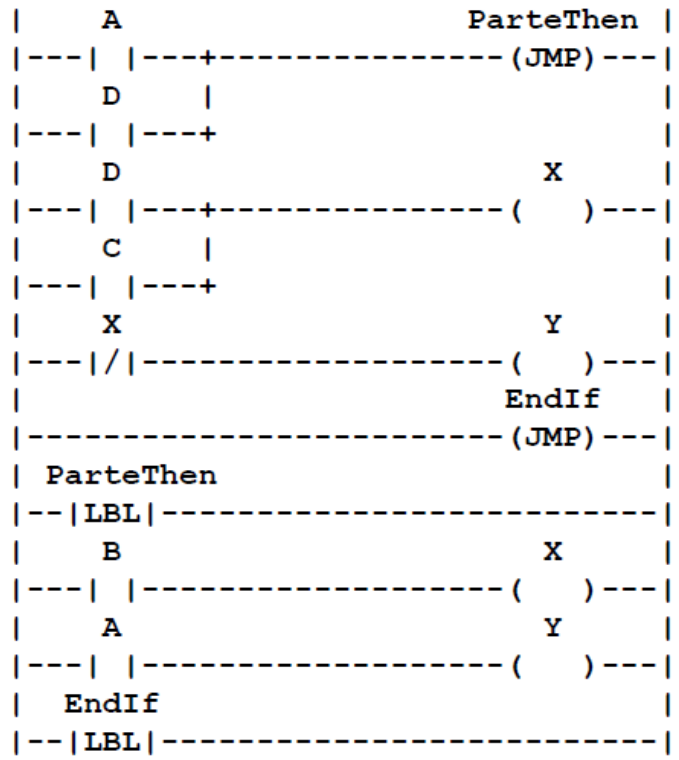
```
---(JMP)---
```

```
---|LBL|---
```

Se la bobina JMP è alimentata, il PLC salta al rung successivo a quello che contiene il solo elemento LBL

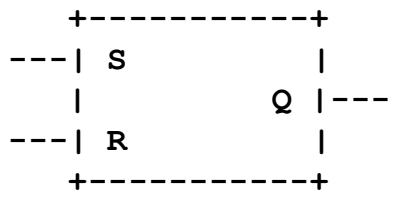
Esempio di codice LD:

```
if (A or D) {
    X = B; Y = A;}
else {
    X = D or C; Y = not X;}
```



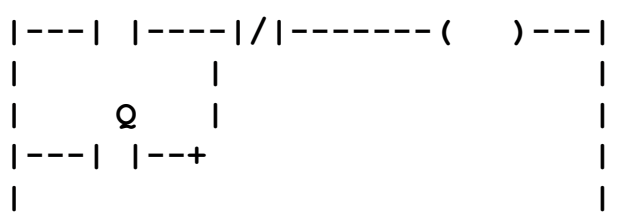
Ladder diagram: esempio

Flip-Flop SET RESET



L'uscita Q viene posta a 1 se l'ingresso S (Set) passa a 1 e tale rimane finché non viene posto a 1 l'ingresso R (Reset) che porta a 0 l'uscita (l'ingresso di RESET è prioritario)

```
Q = (NOT R) AND (Q OR S)
```



Per esempio questo programma potrebbe costituire la logica di abilitazione di un motore:

- S: interruttore di abilitazione
- R: interruttore di disabilitazione
- Q: motore in moto

Sistemi real time

- Un sistema di elaborazione real time (ovvero in tempo reale) deve essere in grado di eseguire tutti i suoi compiti senza violare vincoli temporali specificati.
- Deve quindi rispondere in un modo certo ed entro tempi fissati a eventi esterni, prevedibili o no.
- La correttezza complessiva di un'operazione dipende quindi non solo dalla correttezza logica ma anche dal momento in cui l'operazione è eseguita

Esempio

I sistemi di controllo devono reagire entro tempi rigidamente vincolati a certi segnali di ingresso, pena danneggiamenti anche gravi dell'impianto controllato o dell'ambiente in cui l'impianto opera.

Sistemi real time

- Tempo reale non vuol dire necessariamente veloce: il fallimento nel calcolare una risposta entro un certo tempo può essere altrettanto scorretto, e avere effetti altrettanto gravi, che calcolarla prima del tempo dovuto.
- Si può quindi dire che tempo reale non vuol dire essere veloci, ma essere sempre puntuali.
- A volte si definiscono “in tempo reale” sistemi che in realtà non lo sono (come i sistemi interattivi in tempo reale).

Hard real time e soft real time

I sistemi real time sono classificati in base alla conseguenza del mancato soddisfacimento dei vincoli temporali:

- Hard real time:
  - La classe dei sistemi real time per cui il fallimento nel rispettare i vincoli temporali (spesso molto stringenti) può portare a gravi danni o addirittura disastri
  - Esempi: sistemi di controllo, sistemi medicali

- Soft real time:
  - La classe dei sistemi real time per cui il fallimento nel rispettare i vincoli temporali (più o meno stringenti) non porta a danni immediati, ma al più a una diminuzione dei benefici, che sono decrescenti col tempo
  - Esempi: trasmissioni audio/video, aggiornamento di display

Requisiti hardware di un sistema real time

Il requisito di tempo reale richiede, dal punto di vista hardware:

- L'utilizzo di uno o più processori di adeguata velocità di elaborazione
- Che il tempo di esecuzione delle istruzioni sia noto (almeno nei valori massimi)
- Che l'accesso alla memoria ed ai dispositivi di I/O sia veloce, affidabile e deterministico
- La garanzia di una tempificazione di riferimento certa
- La presenza di funzioni di autodiagnostica
- La presenza di ridondanze strutturali, in modo da poter continuare ad operare anche in presenza di malfunzionamenti

Sistemi operativi real time

- Se l'applicazione di controllo è molto semplice, il modo in cui i vari compiti (*task*) vengono eseguiti in parallelo e in cui sono gestite le risorse è totalmente determinato dal programma utente
- In tutti gli altri casi il dispositivo di controllo dovrà prevedere un (minimo di) sistema operativo che si occupi almeno della pianificazione dell'esecuzione dei processi (scheduling) e della gestione della comunicazione tra i processi
- Il requisito di tempo reale si traduce quindi in requisiti sul sistema operativo

Requisiti di un sistema operativo real time

- Un sistema operativo real time deve avere le seguenti caratteristiche:
  - Avere una pianificazione dell'esecuzione che preveda un meccansimo di assegnazione di priorità ai processi
  - Essere multitasking pre-emptive, ossia essere in grado di interrompere in qualsiasi istante un processo per trasferire le risorse ad un processo più prioritario
  - Supportare il cambio di contesto e l'allocazione della memoria
  - Evitare situazioni di stallo (deadlock)
  - Realizzare un meccanismo di sincronizzazione e comunicazione tra processi
  - Gestire le interruzioni (interrupt)

Sistemi operativi real time

I sistemi operativi real time presentano difficoltà di progettazione e verifica a posteriori.

- Il progetto è complicato: operazioni ritenute sicure nella pratica di progetto possono non essere più corrette per i sistemi operativi real time
- Il sistema implementato non può spesso essere provato sul campo (si pensi al controllore di una centrale nucleare): serve un simulatore dell'ambiente;
- È molto importante che il loro sviluppo sia fin dall'inizio rigoroso, basato su tecniche precise e assistito da strumenti opportuni.
- Occorre fare uso di metodologie consolidate di ingegneria del software.

Sistemi operativi real time

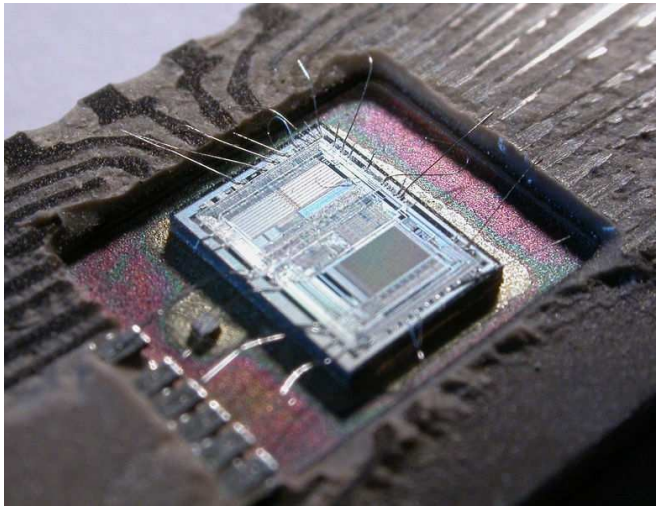
Alcuni sistemi operativi real time di interesse sono i seguenti:

- Proprietari:
  - QNX
  - VxWorks
  - Windows CE
- Open source:
  - RTAI Linux
  - Xenomai

Sistemi embedded

Un sistema embedded :

- È inserito in altri prodotti (es. in automobile)
- È “single purpose”: fa una cosa sola in modo efficiente ed economico (contro i PC che sono “general purpose”)
- Avendo dei compiti noti già durante lo sviluppo può essere progettato con hardware/software specificamente studiati per la tale applicazione.



- L’hardware può essere ridotto ai minimi termini per ridurne lo spazio occupato, riducendo così anche i consumi, i tempi di elaborazione (maggiore efficienza) e il costo di fabbricazione.
- Inoltre l'esecuzione del software è spesso in real-time per permettere un controllo deterministico dei tempi di esecuzione
- Ha un’interfaccia utente primitiva o inesistente

Sono sistemi embedded quelli installati nei telefonini, nelle automobili, negli elettrodomestici, nei ricevitori GPS, nei satelliti, nei giocattoli, ecc.

Sistemi embedded

Alcune caratteristiche dei sistemi embedded

- Dal punto di vista software sono dotati di sistema operativo real time
- Hanno piccola occupazione di memoria (tipico 1÷2 KB di memoria RAM/ROM), e devono essere eseguiti da CPU di piccola potenza e consumo
- Devono funzionare sempre senza interventi manuali
- È bene che non abbiano parti in movimento, che possono rompersi, consumano molta energia, sono lente, richiedono driver complessi, occupano spazio
- Devono gestire e recuperare, eventualmente con funzionalità ridotte, situazioni di errore
- Devono ripartire (reboot) automaticamente in caso di arresto, eventualmente in configurazioni di sicurezza e “istantaneamente”
- Devono fare autotest di funzionamento (watchdog, ecc)