



## **Automazione industriale dispense del corso 15. PLC e standard IEC 61131**

Luigi Piroddi  
[piroddi@elet.polimi.it](mailto:piroddi@elet.polimi.it)

## Introduzione

Un controllore logico è un dispositivo che mette in relazione delle variabili (logiche) in ingresso con variabili (logiche) di uscita, mediante un insieme di algoritmi combinatori e/o sequenziali.

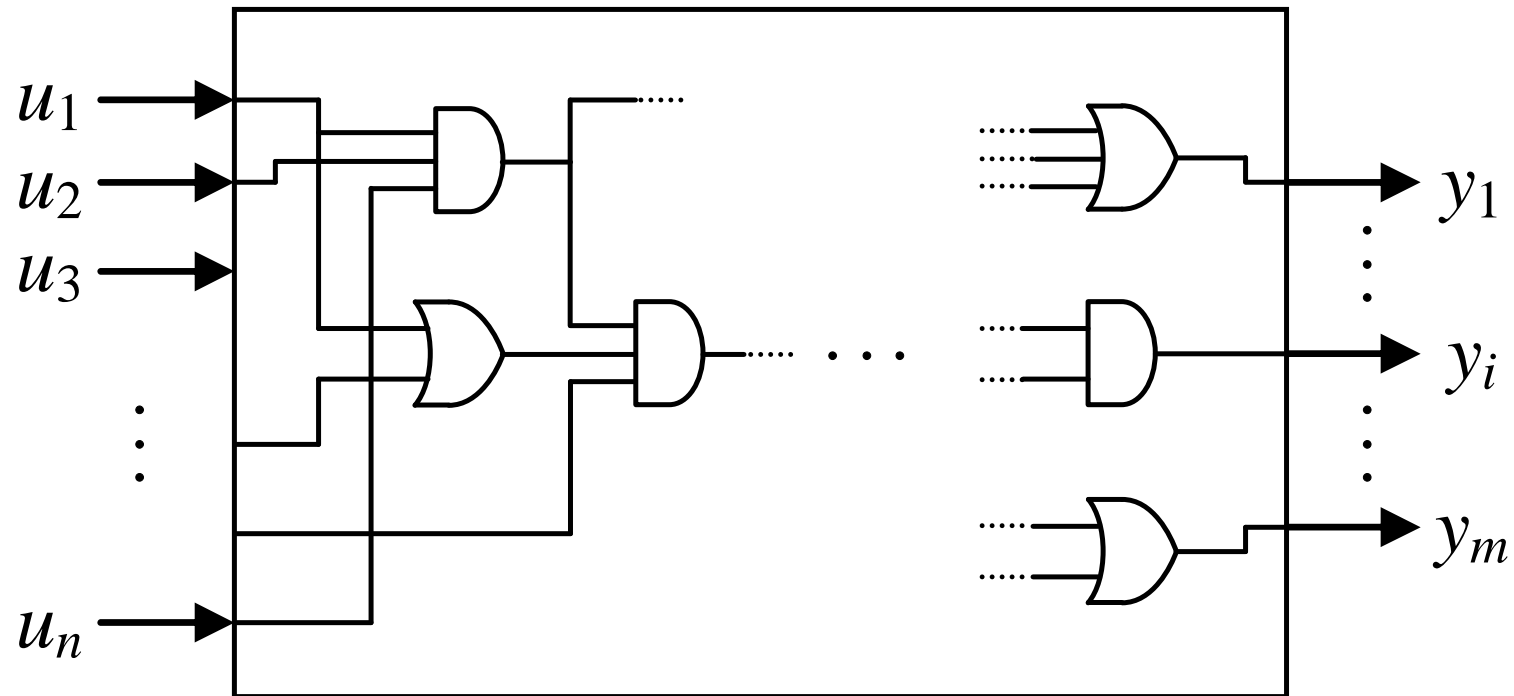
Il controllore logico è detto

- ▶ *statico*, se le uscite del sistema ad un certo istante dipendono dal valore degli ingressi allo stesso istante (le equazioni che legano ingressi e uscite sono *statiche*)
- ▶ *dinamico*, se le uscite correnti dipendono anche dai valori passati degli ingressi (equazioni di tipo *sequenziale*)

Dal punto di vista implementativo qualsiasi algoritmo logico/sequenziale può essere realizzato mediante un'opportuna combinazione di

- ▶ porte logiche fondamentali (AND, OR, NOT)
- ▶ elementi di ritardo

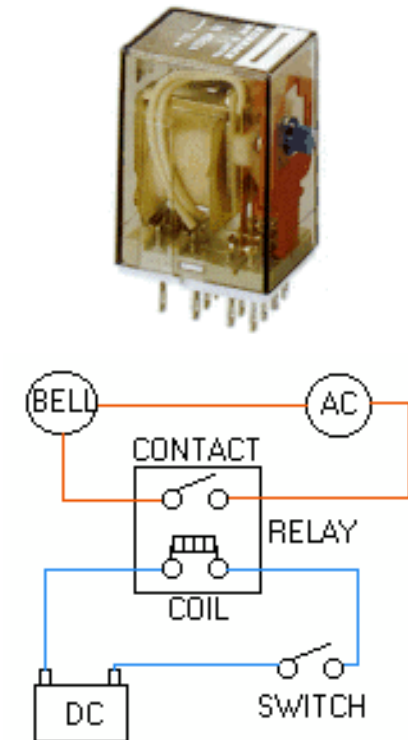
Un controllore logico può essere quindi realizzato in modo *cablato*, ovvero mediante un insieme di dispositivi fisici che realizzano le porte logiche e la loro interconnessione che definisce l’algoritmo logico/sequenziale.



Fino agli anni '60 i sistemi di automazione e controllo erano implementati tramite cablaggio serie/parallelo di dispositivi elettromeccanici, come relé, temporizzatori, contatori (quadri a relé).

Un relé è un interruttore elettromagnetico: se si applica una tensione all'avvolgimento si genera un campo magnetico che attira i contatti del relé, e ne determina la connessione, lasciando passare la corrente e chiudendo così il circuito.

Esempio: nel circuito rappresentato a lato, quando l'interruttore viene chiuso passa corrente nell'avvolgimento del relay generando un campo magnetico che fa chiudere i contatti del relé facendo suonare la campanella.

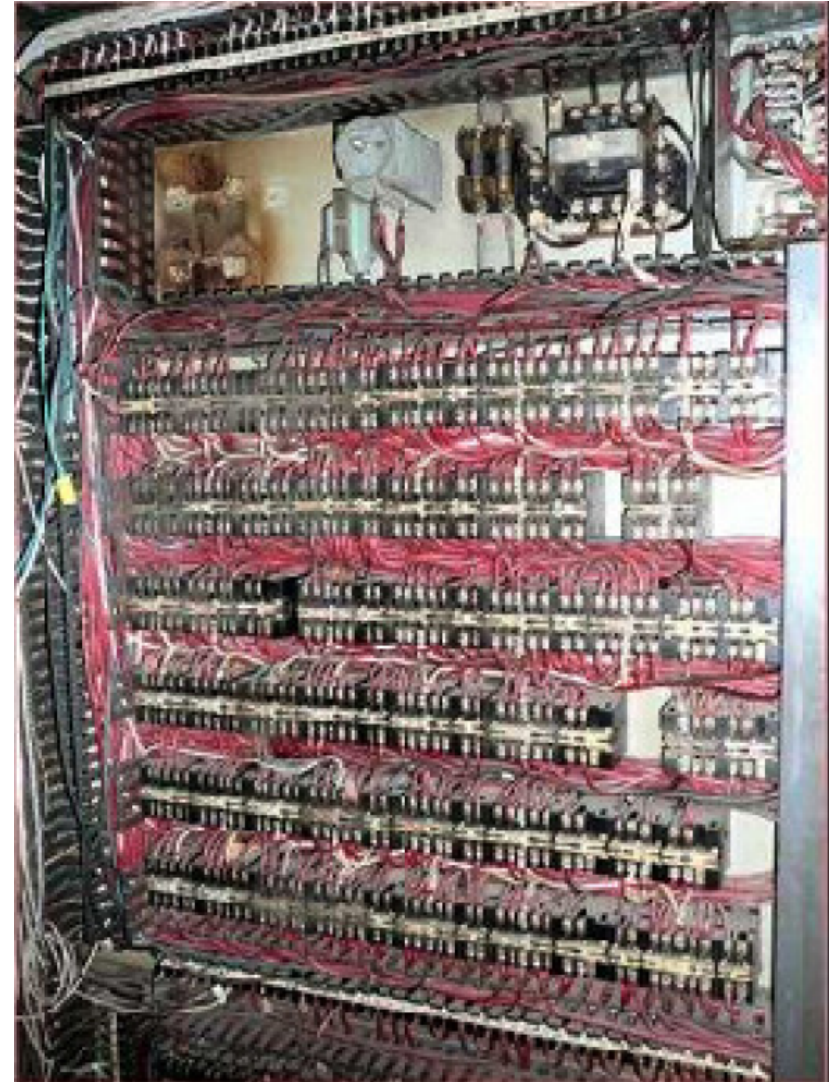


Mediante i relé è perciò possibile definire delle porte logiche (stati logici = relé eccitato e non eccitato), e connettendoli opportunamente (cablaggio) è possibile definire una qualunque rete logico/sequenziale.

## Quadri a relé

Con i quadri a relé era possibile implementare controlli anche di elevata complessità, ma c'erano una serie di inconvenienti:

- ▶ elevato costo di progettazione
- ▶ complessità e ingombro (100÷1000 relé, cablatura)
- ▶ modifica difficile a progettazione completata (rigidezza della logica *cablata*)
- ▶ difficile integrazione con dispositivi di controllo modulante
- ▶ scarsa affidabilità nel tempo (i relé sono dispositivi meccanici con una durata di vita limitata)
- ▶ manutenzione difficile
- ▶ velocità di elaborazione limitata

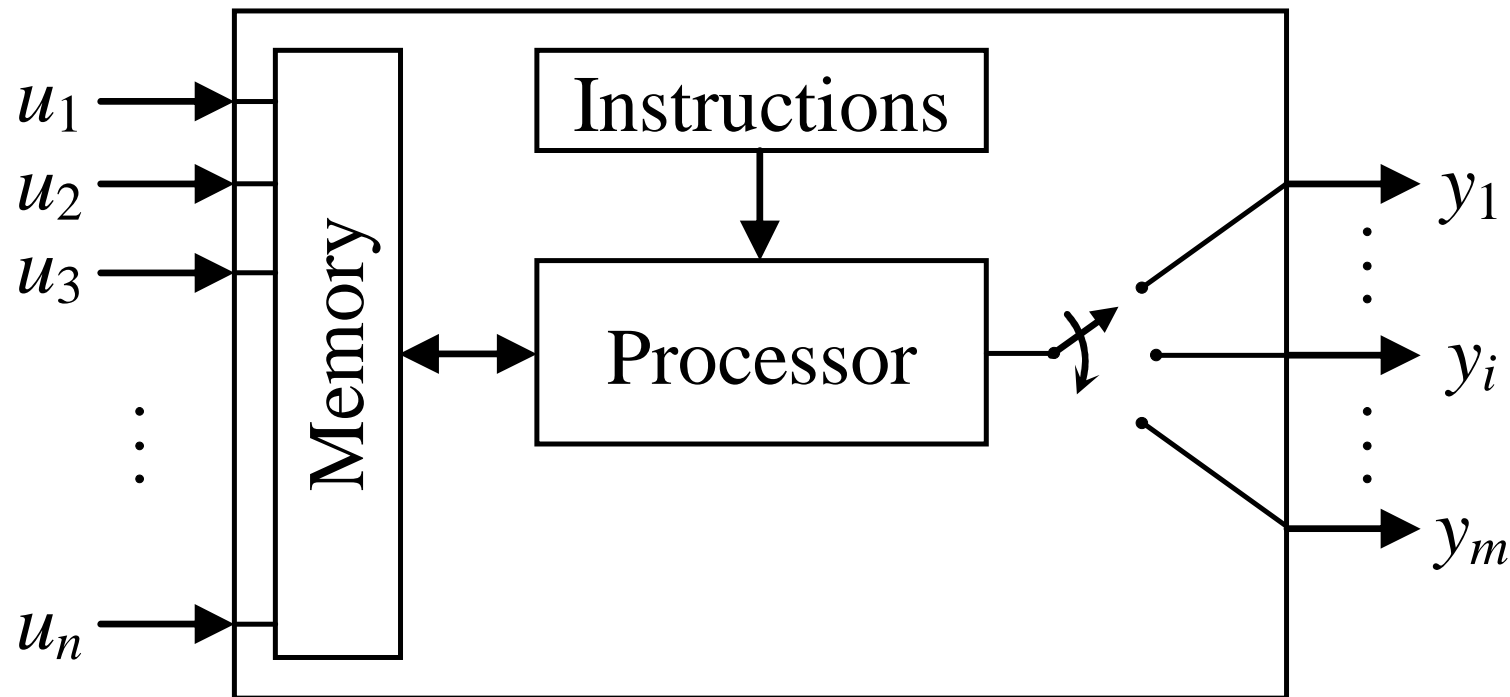


C’era quindi l’esigenza di disporre di una nuova generazione di controllori, che fossero:

- ▶ facilmente programmabili e, soprattutto,
- ▶ *riprogrammabili*, anche sul luogo stesso di funzionamento con tempi di interruzione minimi
- ▶ robusti e adatti ad operare in un ambiente industriale
- ▶ caratterizzati da una modularità tale da rendere semplici le azioni di manutenzione e riparazione
- ▶ facilmente espandibili
- ▶ facilmente integrabili e interfacciabili con sensori, attuatori, sistemi di raccolta dati
- ▶ di dimensioni ridotte
- ▶ a basso consumo energetico
- ▶ a basso costo
- ▶ standardizzabili
- ▶ dotati di una memoria interna espandibile per programmi e dati

Un controllore logico *programmato* è realizzato mediante un sistema elettronico programmabile ed un opportuno programma di controllo memorizzato.

Richiede un processore per eseguire le istruzioni del programma e una memoria per conservare dati e programmi.



## Differenze tra logica cablata e logica programmata

### Logica cablata:

- ▶ la sequenzialità dell'algoritmo logico è ottenuta mediante l'utilizzo di anelli di retroazione
- ▶ tutti i segnali presenti ad un certo istante vengono considerati *simultaneamente* dal dispositivo di calcolo
- ▶ la propagazione dei segnali tra strati circuitali connessi in modo sequenziale può avvenire in modo totalmente asincrono (possibili *instabilità* dovute ai diversi tempi di propagazione dei segnali nel circuito), oppure regolato da un clock
- ▶ nel secondo caso, gli elementi del circuito devono essere dotati di un ingresso di abilitazione per ricevere gli impulsi di clock
- ▶ normalmente il processo di propagazione è *quasi-simultaneo*, perchè avviene strato per strato in modo sequenziale, mentre in ogni strato la propagazione è parallela e simultanea.

## Logica programmata:

- ▶ i diversi strati di logica sequenziale sincrona sono sostituiti da *un unico processore universale*, che simula le funzioni svolte dai vari strati, eseguendo selettivamente e sequenzialmente le istruzioni appropriate sui segnali appropriati
- ▶ il processore è esso stesso un dispositivo logico sequenziale *sincrono*
- ▶ per operare correttamente, è necessario utilizzare una memoria, dove conservare i dati (così come i risultati delle operazioni intermedie) affinché possano essere processati al momento opportuno.

Il primo *controllore logico programmabile* (o *PLC*) con queste caratteristiche fu prodotto nel '68 dalla General Motors (Bedford Associates, MODICON 084).

Pochi anni dopo, a metà degli anni '70 la Allen Bradley introdusse sul mercato il primo PLC basato su microprocessore 8080.

Ai giorni nostri non si parla più di logica programmabile, quanto di *logica distribuita*, visto che vengono impiegati più PLC basati su microprocessori connessi in rete, a vantaggio della modularità sia “fisica” che dei costi.



La struttura HW/SW dei PLC è regolata dalla normativa industriale IEC 61131 (del 1993, recepita in Italia nel 1996).

## Perché è necessario uno standard come l’IEC 61131?

In generale, l’aderenza ad uno standard industriale serve a proteggere l’investimento, garantendo che le soluzioni adottate siano supportate anche nel futuro e anche se si decidesse di cambiare i fornitori.

Questa esigenza è particolarmente sentita nell’automazione industriale, dato che:

- ▶ un sistema di automazione industriale deve tipicamente integrare diversi dispositivi di misura, controllo e attuazione
- ▶ la struttura “fisica” del sistema di automazione può essere parzialmente vincolata
  - ▼ quando alcune sue componenti sono già installate nelle macchine dai loro produttori
  - ▼ quando si affronta il revamping di un impianto già esistente
- ▶ a volte è necessario cambiare fornitore di tutto o parte del sistema HW/SW usato
- ▶ tipicamente il sistema di automazione deve comunicare con dispositivi che non sono PLC, ad esempio per integrarsi col sistema informativo
- ▶ spesso i programmatori non sono specialisti del controllo, per cui nel fornire loro le specifiche occorre essere assolutamente univoci
- ▶ spesso occorre iniziare a verificare la funzionalità del sistema quando la struttura fisica non è ancora del tutto definita, oppure lo è ma potrebbe cambiare

## Normativa IEC 61131

La normativa IEC 61131 si applica ai PLC e alle periferiche associate, come p.es. strumenti di programmazione e debugging e interfacce uomo-macchina, adibiti al controllo di macchine e processi industriali.

Tale normativa si occupa di vari aspetti relativi ai PLC, come le specifiche dei dispositivi, i linguaggi di programmazione, i protocolli per la comunicazione, ecc.

La normativa 61131:

- ▶ fornisce modelli, concetti e terminologia comuni ai tecnici del settore
- ▶ costituisce un riferimento per la realizzazione di strumenti di sviluppo, verifica e simulazione
- ▶ facilita l'interazione tra progettisti e il riuso di elementi del progetto
- ▶ consente la sopravvivenza a diverse generazioni tecnologiche
- ▶ favorisce correttezza, qualità e basso costo del software

In particolare, la parte 3 introduce il modello software e i linguaggi di programmazione utilizzabili per programmare un PLC.

## Il controllore logico programmabile (PLC)

Definizione di *PLC* secondo la normativa IEC 61131:

- il PLC è un sistema elettronico a funzionamento digitale, destinato all'uso in ambito industriale, che utilizza una memoria programmabile per l'archiviazione interna di istruzioni orientate all'utilizzatore per l'implementazione di funzioni specifiche, come quelle logiche, di sequenziamento, di temporizzazione, di conteggio e di calcolo aritmetico, e per controllare, mediante ingressi ed uscite sia digitali che analogici, vari tipi di macchine e processi

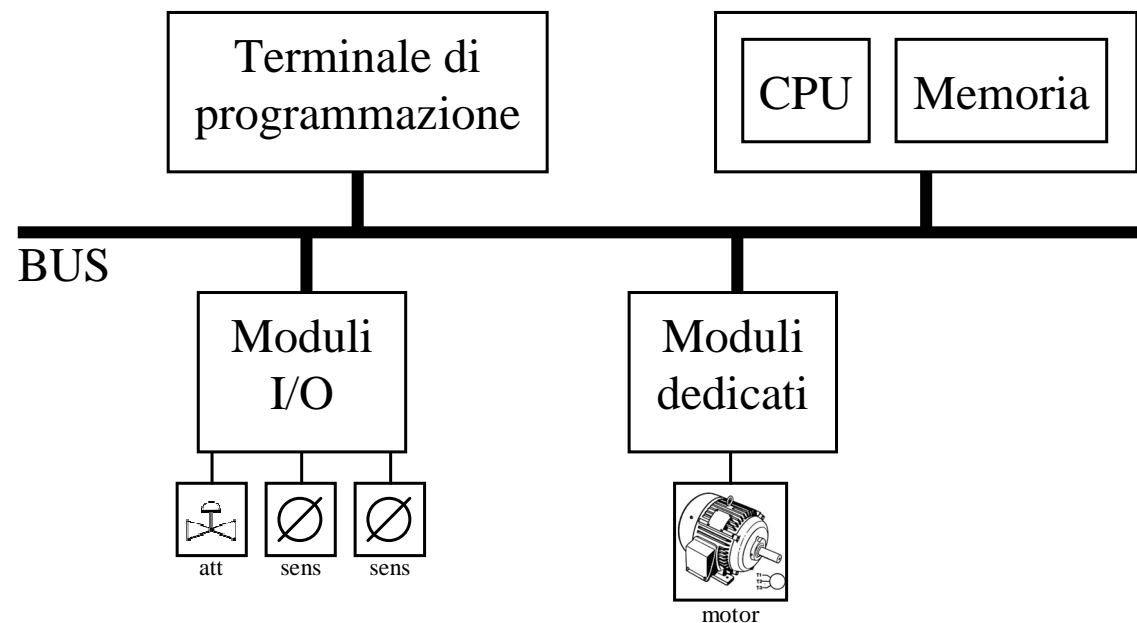


Definizione di *sistema PLC* secondo la normativa IEC 61131:

- configurazione realizzata dall'utilizzatore, formata da un PLC (con i suoi processori specializzati e non e i moduli di ingresso/uscita industriali) e dalle periferiche associate, necessaria al sistema automatizzato previsto

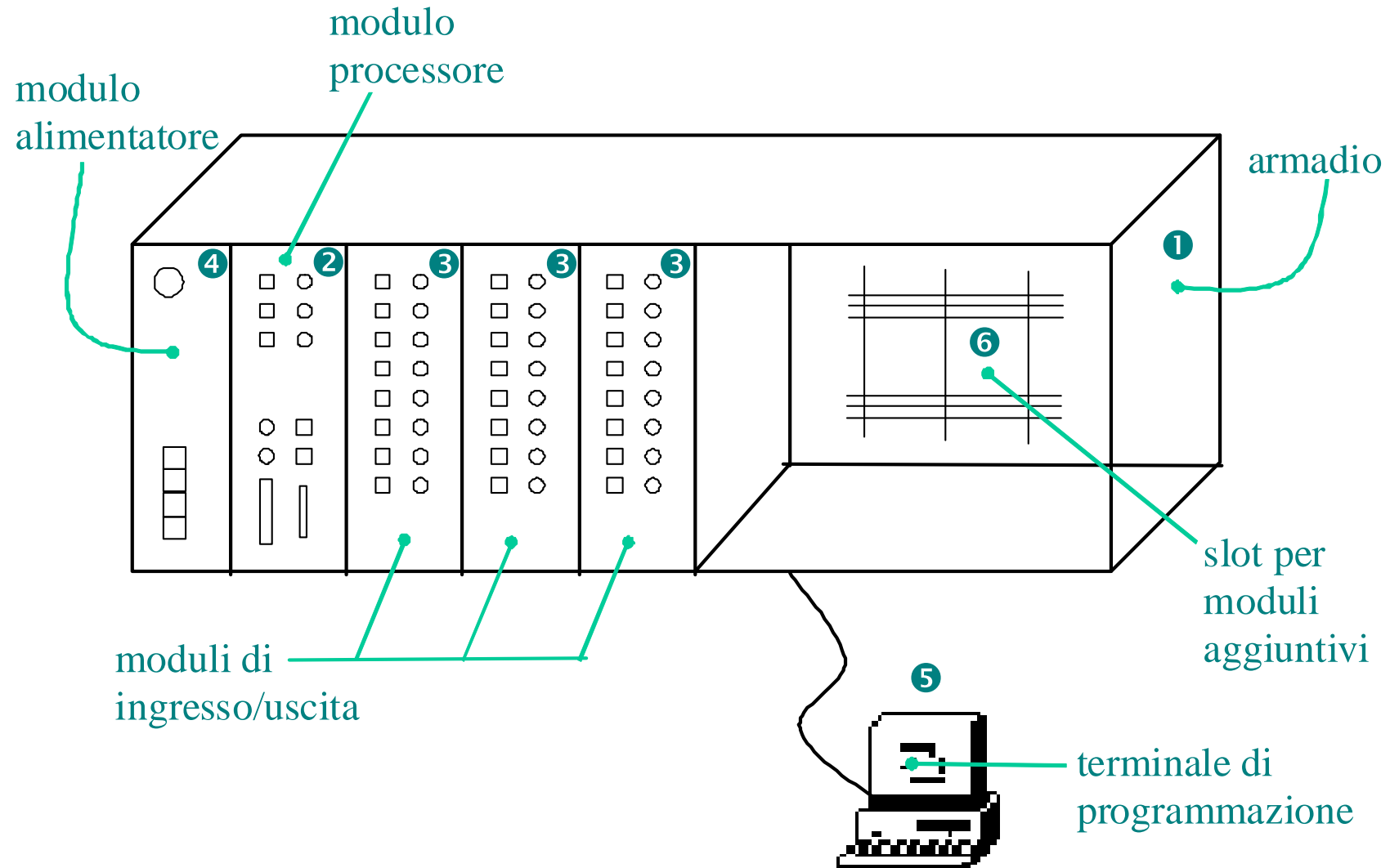
## Architettura HW di un PLC

Il PLC è un controllore a bus con un architettura simile a quello di un generico calcolatore elettronico. La struttura a bus è fondamentale per la modularità.



E' presente un'unità centrale (CPU), dotata di memoria, e collegata tramite il bus a moduli di interfaccia di input/output, connessi a loro volta con i sensori e gli attuatori del livello di campo. Sempre tramite il bus si possono connettere un terminale di programmazione (PC) e moduli dedicati.

## Componenti fondamentali di un PLC



## ① Armadio

L'*armadio* (o *cestello* o *rack*) è l'involucro che contiene e racchiude i vari moduli, e ospita fisicamente il bus a cui essi vanno agganciati meccanicamente ed elettricamente.

Normalmente contiene degli slot supplementari per eventuali moduli aggiunti in un secondo tempo.

Il *bus* connette tutte le unità funzionali (memoria, processore e moduli ingresso/uscita) tra di loro e permette un flusso continuo di informazioni da e verso la CPU.

E' un insieme di linee elettriche suddivise per funzioni (linee per i dati, linee per gli indirizzi, linee per l'alimentazione).

Generalmente, i bus dei PLC sono di tipo proprietario (connessioni fisiche e protocolli di comunicazione).

L'information rate del bus (tipicamente dell'ordine di 10 Mbytes/s) è un fattore determinante nelle prestazioni del PLC.

Caratteristiche fondamentali (essenzialmente di tipo meccanico) dell'armadio:

- ▶ numero di slot (espandibilità)
- ▶ dimensioni di ingombro
- ▶ modalità di fissaggio all'impianto di automazione

L'armadio deve garantire la schermatura elettromagnetica dei moduli nonché un'adequata robustezza meccanica. Più in generale, deve proteggere il sistema da vari fenomeni potenzialmente dannosi, spesso riscontrati in ambiente industriale:

- ▶ effetti ambientali (temperatura, umidità)  
l'elevata umidità può causare condensazione e accelera corrosione  
la bassa umidità può generare dei potenziali elettrostatici
- ▶ sollecitazioni meccaniche  
vibrazioni e urti possono danneggiare i contatti, le saldature e i componenti in genere
- ▶ corrosione chimica  
attenzione a gas corrosivi, vapori di idrocarburi, polvere metallica o minerale
- ▶ interferenze elettriche  
generazione di tensioni per effetto termico (Peltier), differenze di potenziale su giunzioni tra metalli diversi, effetti elettrostatici, interferenze elettromagnetiche

## ② Modulo processore

Il modulo processore racchiude una scheda con:

- ▶ uno o più microprocessori, che eseguono i programmi del sistema operativo e quelli sviluppati dall'utente
- ▶ la memoria dove questi programmi sono conservati
- ▶ i dispositivi per il controllo del bus di comunicazione con gli altri moduli
- ▶ tutti i componenti necessari al funzionamento

Una configurazione tipica a 3 microprocessori è la seguente:

- ▶ un microprocessore specializzato sulle operazioni a 1 bit
- ▶ un microprocessore per le istruzioni aritmetico/logiche
- ▶ un microprocessore per la comunicazione con i moduli I/O e dispositivi esterni

Caratteristiche principali:

- ▶ gestione automatica dei segnali di ingresso/uscita
- ▶ scansione *ciclica* del programma utente (questa è la caratteristica principale che contraddistingue il PLC rispetto ad altri sistemi informatici)

## Il ciclo di copia massiva

La modalità di funzionamento standard consiste nell'eseguire periodicamente il *ciclo di copia massiva degli ingressi e delle uscite*:

- ❶ lettura (sincrona) degli ingressi e loro caricamento in un'area specifica della memoria (che contiene quindi un'immagine del processo in quel momento)
- ❷ esecuzione programma utente, operando sui valori in memoria e conservando i risultati in memoria
- ❸ esecuzione programmi di gestione sistema (p.es. diagnostica)
- ❹ scrittura (sincrona) sulle uscite fisiche dei valori corrispondenti conservati nell'area di memoria riservata a questo scopo

Caratteristiche del ciclo di scansione:

- ▶ consente una gestione ottimale delle comunicazioni con i moduli I/O
- ▶ garantisce che i valori memorizzati degli ingressi rimangano inalterati durante l'esecuzione dei programmi
- ▶ la lettura di ingressi e uscite è gestita interamente e in modo trasparente dal sistema operativo, consentendo all'utente di concentrarsi sul programma applicativo

## Avviamento (start-up) e spegnimento

Per il corretto funzionamento del sistema il programma di controllo dovrà prevedere delle parti apposite dedicate all'accensione e allo spegnimento del sistema.

All'accensione del sistema lo stato del processo è generalmente non noto.

Prima di iniziare la normale elaborazione ciclica del controllore il sistema va portato in uno stato noto.

- ▶ *avviamento a freddo*  
il processo deve essere completamente inizializzato e portato in una fase adatta all'inizio della produzione vera e propria
- ▶ *avviamento a caldo*  
il processo deve ripartire dopo una breve interruzione

Allo spegnimento del sistema occorre prevedere una procedura per la fermata ordinata del sistema.

- ▶ di norma, si completano le sequenze operative del processo ancora in corso, in modo da svuotare in sequenza le varie componenti dell'impianto e lasciarlo in uno stato di riposo

## Funzioni real-time

Le fasi di acquisizione ingressi, elaborazione e aggiornamento uscite sono da considerarsi *hard real-time*, in quanto la precisione nella esecuzione periodica degli algoritmi è critica per il funzionamento corretto del sistema.

Il funzionamento del PLC è essenzialmente time-driven, per via dell'esecuzione ripetuta del ciclo di copia massiva, ma in casi di urgenza particolare (guasti, emergenze) si possono eseguire operazioni asincrone con accesso immediato ai punti di ingresso/uscita (per attuare immediatamente dei comandi) o gestire interrupt.

Ad esempio, se cade l'alimentazione, il sistema operativo salva lo stato dell'elaborazione su una memoria apposita (alimentata con una batteria tampone).

Altre attività sono classificabili come *soft real-time*, come per esempio la comunicazione con altri sistemi per lo scambio di comandi, parametri, stati del sistema controllato (ad esempio per memorizzare serie storiche a scopo di monitoraggio e controllo qualità).

## Prestazioni del PLC

Le prestazioni del PLC sono misurate in termini di:

- ▶ *tempo di scansione*
  - ▼ misura la *velocità di elaborazione* del modulo
  - ▼ tempo che intercorre tra due attivazioni successive della stessa porzione del programma applicativo (dipende da quanti ingressi e uscite bisogna aggiornare e dalle dimensioni e complessità del programma utente)
  - ▼ definito in millisecondi per *KWord* di programma (con word di 8 o 16 bit)
  - ▼ il valore indicato per un PLC è quello corrispondente al valor medio per programmi di media complessità e tipicamente varia da qualche unità a qualche decina di *ms/KWord*
- ▶ *tempo di risposta*
  - ▼ massimo intervallo di tempo che passa tra la rilevazione di un certo evento e l'esecuzione dell'azione di risposta programmata (tiene conto anche dei ritardi introdotti dai moduli I/O)

## Sistema operativo di un PLC

Il sistema operativo di un PLC è costituito da un insieme di programmi di supervisione memorizzati in modo permanente, con le seguenti funzioni:

- ▶ controllo delle attività del sistema complessivo
- ▶ elaborazione dei programmi utente
- ▶ acquisizione ingressi e attuazione uscite
- ▶ gestione della comunicazione tra i moduli
- ▶ diagnostica interna
  - ▼ watchdog timer (temporizzatore che controlla il tempo di esecuzione di una funzionalità, generando un allarme se esso supera una determinata soglia)
  - ▼ controlli di parità sulla memoria e sulle linee di comunicazione
  - ▼ controllo della tensione di alimentazione e dello stato di carica delle batterie tampone
- ▶ gestione software dell'interfaccia con l'operatore

## Modalità operative di un PLC

### Modalità operative:

- ▶ *modalità di programmazione*
  - ▼ modalità utilizzata per caricare il codice nella memoria del PLC
  - ▼ in passato il PLC era programmato direttamente in memoria e il programma era convertito in codice macchina da un *interprete*
  - ▼ attualmente è normalmente previsto l'uso di terminali PC per la programmazione e questo consente di *compilare* il codice prima di scaricarlo sul PLC
- ▶ *modalità di validazione/debugging*
  - ▼ si eseguono i programmi con l'aggiornamento delle uscite disabilitato, per testare la correttezza del codice
  - ▼ debugging avanzato (esecuzione passo-passo, uso di break-point, ecc.)
  - ▼ debugging con uscite abilitate e ingressi imposti via software per controllare la risposta dell'impianto in condizioni particolari
- ▶ *modalità di esecuzione*
  - ▼ si eseguono i programmi utente e si aggiornano gli ingressi e le uscite
  - ▼ in tale modalità il sistema operativo garantisce il funzionamento real-time

## Memoria di un PLC

La memoria di un PLC è organizzata per aree distinte:

- ▶ area sistema operativo (ROM)
- ▶ area di lavoro del sistema operativo (RAM)
- ▶ area I/O (RAM), divisa in due parti per evitare la sovrapposizione tra dati di ingresso e di uscita
- ▶ area programmi utente (RAM durante lo sviluppo, poi PROM; ora anche memorie flash EEPROM)
- ▶ area dati utente (RAM)

La memoria per i programmi utente è dell'ordine di 1 – 100 *KWord*.

### ③ Moduli di ingresso/uscita

Il PLC comunica con il processo fisico attraverso moduli di I/O, sia digitali che analogici, rilevando eventi e dati dai sensori e comandando azioni agli attuatori.

I moduli I/O devono:

- ▶ realizzare l'*interfaccia elettrica* con la strumentazione di campo, ovvero l'adattamento e il condizionamento tra i livelli di tensione e corrente propri del PLC (elettronica di bassa potenza) e quelli (più elevati) usati per la trasmissione dei segnali
- ▶ assicurare l'*isolamento galvanico* tra il campo, il bus e l'elettronica interna del PLC (in modo da salvaguardarla da sovratensioni dovute a collegamenti sbagliati, cortocircuiti o guasti)
- ▶ realizzare le *conversioni D/A* e *A/D* necessarie per interfacciare direttamente il processo fisico (continuo) con il PLC (*moduli I/O analogici*)

L'indirizzamento delle parole di memoria dove vengono memorizzati gli stati degli ingressi e delle uscite dipende da dove viene fisicamente posto il modulo I/O nell'armadio.

I moduli di ingresso/uscita digitali sono caratterizzati da:

- ▶ livelli di tensioni e correnti gestibili (0-24V, 0-220V, 0-5V in DC, 0-50V in DC)
- ▶ ritardo introdotto dai circuiti di filtraggio utilizzati contro rumore e interferenze

Sono inoltre provvisti di indicatore a led per visualizzare lo stato del segnale.

I moduli I/O analogici sono dotati di circuiti per la conversione A/D e D/A (eventualmente con MUX e DEMUX per risparmiare sul numero di convertitori).

Essi sono caratterizzati da:

- ▶ livelli di tensioni e correnti gestibili ( $\pm 5V$ ,  $\pm 10V$ , 0-5V, 4-20mA in DC)
- ▶ tipologia di segnali (single ended o differenziali)
- ▶ risoluzione e velocità di conversione
- ▶ rappresentazione numerica

I moduli di output analogici prevedono anche degli stadi finali di amplificazione di potenza per comandare i dispositivi che lo richiedono.

Esistono poi moduli di input speciali, interfacciabili con i sensori di uso più comune (termocoppie, termoresistenze, celle di carico, estensimetri) .

## ④ Modulo alimentatore

Il modulo alimentatore

- ▶ fornisce l'alimentazione necessaria al funzionamento di tutti i moduli
- ▶ deve erogare una tensione stabilizzata e priva di fluttuazioni e interferenze
- ▶ deve anche fornire protezione da sovratensioni e cortocircuiti

E' costituito da un trasformatore, un circuito raddrizzatore, un filtro, un circuito stabilizzatore, e un circuito per la protezione elettrica.

Caratteristiche principali:

- ▶ potenza massima erogabile
- ▶ possibilità di connessione di più moduli di alimentazione in parallelo (per aumentare potenza o per ridondanza)
- ▶ possibilità di inviare al PLC un segnale di shutdown nel caso che l'alimentazione scenda sotto un determinato limite (in modo che il PLC abbia il tempo di eseguire le opportune procedure)
- ▶ presenza di indicatori di stato

## ⑤ Terminale di programmazione

I terminali a tastiera si collegano direttamente al PLC (tramite una porta di comunicazione) e consistono di:

- ▶ una tastiera per l’inserimento delle istruzioni
- ▶ un display per il controllo del programma

In questo caso, la programmazione avviene direttamente nella memoria del PLC.

Più frequentemente si impegnano sistemi di sviluppo su PC per effettuare off-line la programmazione del codice da memorizzare sul PLC.

A questo scopo si utilizzano dei terminali PC:

- ▶ dotati di funzionalità di supporto alla programmazione
- ▶ in grado di memorizzare i programmi sviluppati
- ▶ connessi al PLC direttamente o via rete (per diminuire gli spostamenti del progettista e facilitare così il ciclo di programmazione/debugging/manutenzione del codice)
- ▶ utilizzabili anche per il monitoraggio del PLC durante il suo normale funzionamento

## ⑥ Moduli speciali

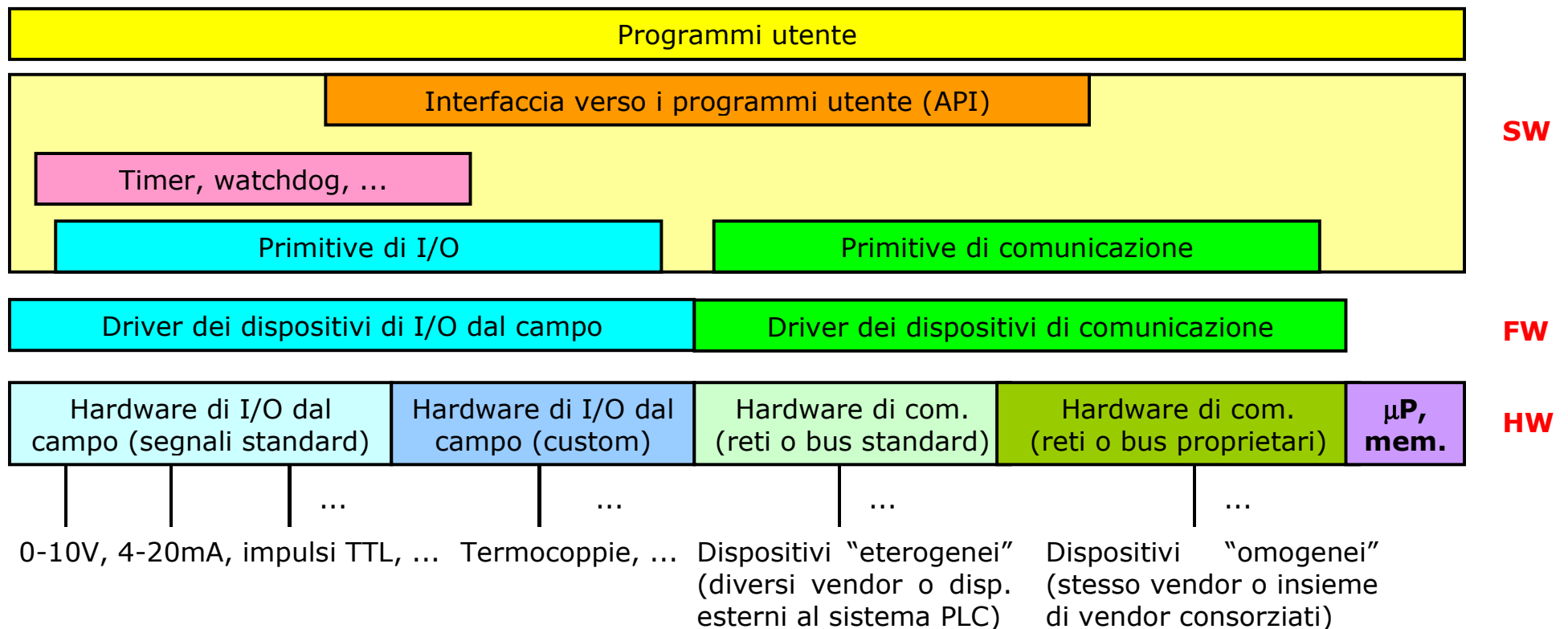
- ▶ moduli di I/O remoto
  - ▼ si utilizzano vari armadi di I/O distribuiti nell'impianto e collegati al PLC tramite un modulo di I/O remoto, che invia gli stati degli ingressi e delle uscite locali via linea seriale ad alta velocità
  - ▼ per impianti distribuiti su un'area di grande dimensione
- ▶ moduli per la connessione in rete
  - ▼ per bus di campo, ethernet, reti proprietarie, ecc.
  - ▼ per consentire la programmazione, il debugging, e la manutenzione del codice in remoto
  - ▼ per connettersi con il sistema informativo aziendale per la raccolta dati
- ▶ moduli encoder
  - ▼ essenzialmente contatori ad alta velocità
- ▶ moduli interfaccia operatore (MMI, Man Machine Interface)
  - ▼ schermi LCD touch-screen

- ▶ moduli di backup
  - ▼ un processore di riserva può essere sincronizzato con quello base e subentrare in caso di malfunzionamento
- ▶ moduli coprocessore
  - ▼ per elaborazioni complesse sui dati memorizzati sul PLC
  - ▼ possono interpretare SW programmato mediante linguaggi non specifici per l'automazione (C/C++, BASIC, ecc.)
- ▶ moduli PID
- ▶ moduli di servo
  - ▼ per l'asservimento di motori

## Struttura SW/FW/HW

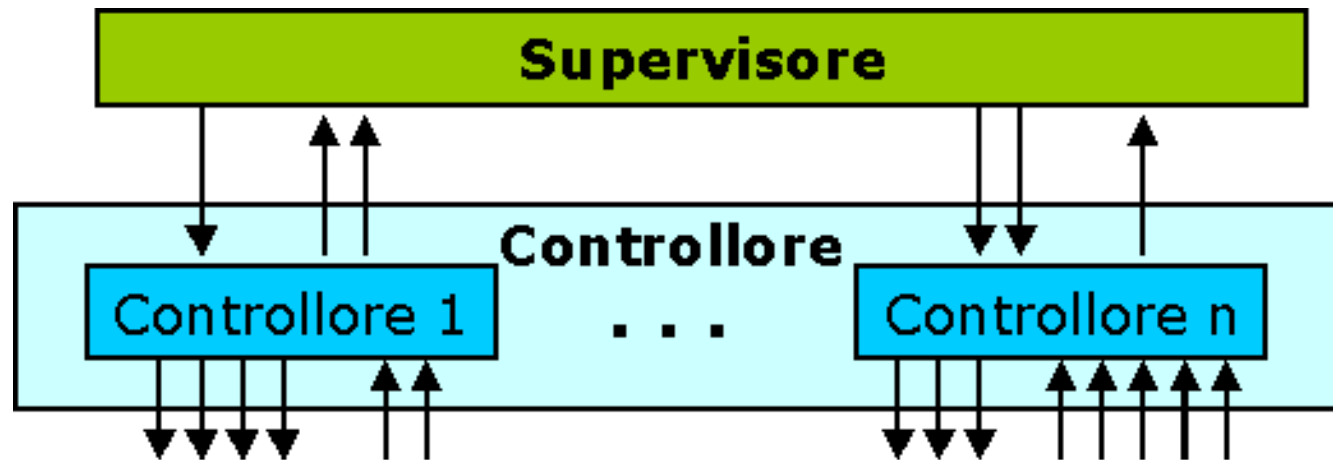
Nonostante abbiano una struttura apparentemente semplice, dal punto di vista hardware/firmware/software i PLC sono oggetti piuttosto complessi.

Schema semplificato (limitato al caso di una CPU):



## Struttura logica e fisica del controllore

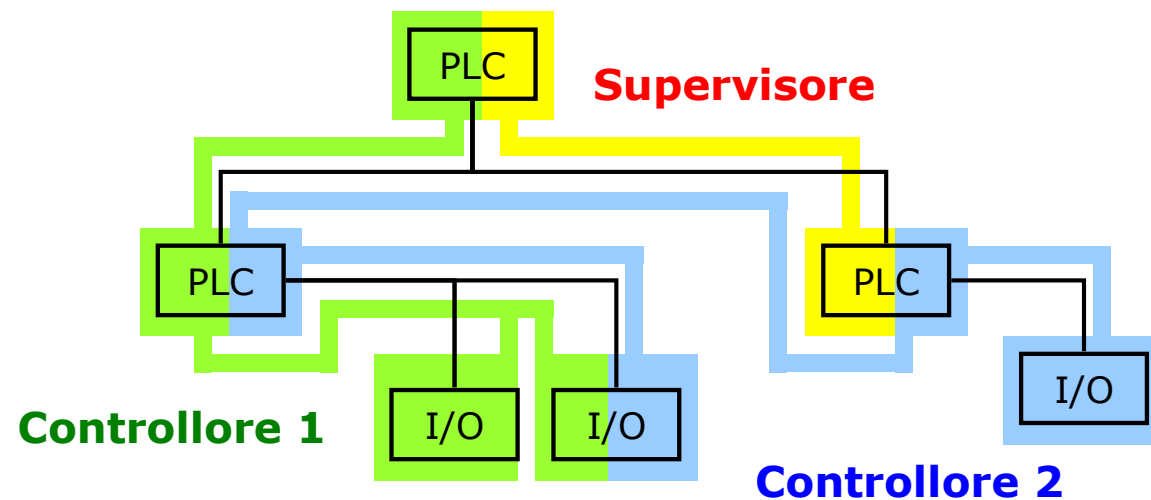
Abbiamo già delineato uno schema concettuale di com'è organizzato il controllo logico di un generico impianto, il che ci ha condotto in modo naturale ad una struttura gerarchica che d'ora in poi chiameremo *struttura logica* del controllo.



Parallelamente, però, esiste un'altra struttura, ovvero quella che connette tra di loro tutti i dispositivi formanti il controllore.

Chiameremo quest'ultima la *struttura fisica* del controllore.

Possiamo quindi dire che la struttura logica si mappa su quella fisica e viceversa.



Palesamente, la stessa struttura logica si può implementare con strutture fisiche differenti in tutto o quasi, dal numero di CPU al tipo di rete e così via.

## Come scegliere un PLC

### Caratteristiche importanti:

- ▶ espandibilità della memoria
- ▶ numero di moduli di I/O gestibili direttamente o in modalità remota
- ▶ numero e tipo di porte di comunicazione disponibili (seriali, parallele, di rete)
- ▶ linguaggi supportati e complessità dell'insieme delle istruzioni
- ▶ possibilità di multi-tasking
- ▶ possibilità di gestire interrupt

### Classificazione di PLC:

- ▶ micro PLC: fino a 64 punti I/O (solo digitali), memorie da 1-2 K  
impiegati in sostituzione di logiche a relé (macchine operatrici, ascensori, lavatrici)
- ▶ piccoli PLC: 64-512 punti I/O (anche analogici), memorie fino a 4 K
- ▶ medi PLC: 256-2048 punti I/O, memorie fino a qualche decina di K
- ▶ grandi PLC: migliaia di punti I/O, memorie da qualche centinaio di K  
utilizzati come supervisori di celle automatizzate o per gestire PLC più piccoli

## **Principali aziende produttrici di PLC operanti sul mercato italiano**

In ordine di fatturato (complessivo):

- ▶ Siemens
- ▶ Allen-Bradley
- ▶ Omron
- ▶ Telemécanique
- ▶ GE Fanuc
- ▶ Toshiba
- ▶ Klockner-Moeller
- ▶ Mitsubishi
- ▶ Matsushita
- ▶ Bosch
- ▶ Hitachi
- ▶ Modicon

## Normativa IEC 61131-3: linguaggi di programmazione per PLC

La standardizzazione dei linguaggi di programmazione:

- ▶ favorisce il progresso verso metodi moderni di sviluppo (spingendo gli sviluppatori ad applicare concetti di programmazione strutturata e modularità)
- ▶ aiuta la portabilità del software
- ▶ facilita la verifica e il riuso del codice
- ▶ riduce costi e tempi di sviluppo (e di adattamento degli sviluppatori a nuovi sistemi)

Linguaggi di programmazione

tipo	nome	acronimo	
linguaggi grafici	diagramma funzionale sequenziale	SFC	Sequential Functional Chart
	linguaggio a contatti	LD	Ladder Diagram
	diagramma a blocchi funzionali	FBD	Function Block Diagram
linguaggi testuali	lista di istruzioni	IL	Instruction List
	testo strutturato	ST	Structured Text

Molti ambienti di sviluppo consentono di usare più linguaggi, anche all'interno del medesimo progetto.

## Elementi comuni a tutti i linguaggi

Lo standard IEC 61131-3 definisce in maniera dettagliata le diverse tipologie di dati (boolean, integer, long integer, real, *time*, *date*, string, word, ecc.) e loro derivati (array, intervalli, enumerati, ecc.) che possono essere utilizzati, indicando anche il numero di bit della loro rappresentazione e il valore iniziale (assegnato di default dal sistema).

- esempio di possibile notazione di una costante temporale:

*t#1d12h50m51s34ms*

Gli identificatori di costanti, variabili, funzioni ecc. sono stringhe di caratteri (prive di spaziature).

I commenti sono delimitati da “(\*)” e “(\*)”.

## Diagramma Funzionale Sequenziale

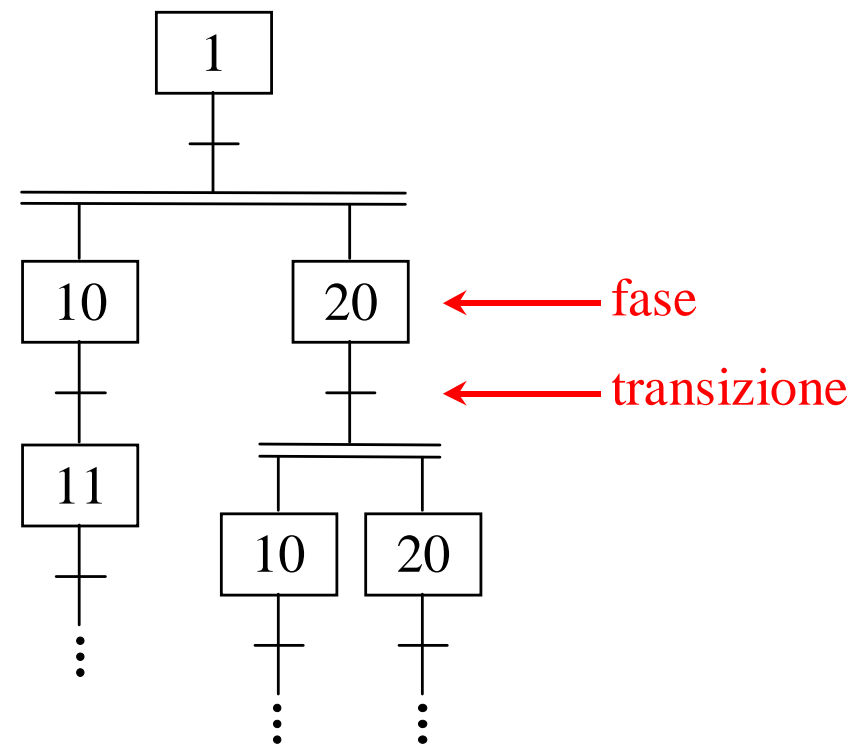
Il *Diagramma Funzionale Sequenziale* (*Sequential Functional Chart, SFC*) è un linguaggio derivato dalle reti di Petri, basato sui concetti di

- ▶ *fase*, che rappresenta le azioni da compiere
- ▶ *transizione*, che rappresenta le condizioni da soddisfare per passare da una fase all'altra

In senso stretto, più che un linguaggio di programmazione vero e proprio, l'SFC è un formalismo grafico per la descrizione di azioni logico/sequenziali.

Infatti, per definire un programma di controllo sequenziale completo, occorre definire azioni e condizioni utilizzando formalismi propri di altri linguaggi di programmazione (come ST o LD).

Esempio:



In questo senso l'SFC è gerarchicamente superiore agli altri quattro linguaggi della normativa.

Da un altro punto di vista, l'SFC rappresenta uno strumento di alto livello per la modellizzazione di problemi logico/sequenziali anche complessi e si presta bene per le fasi iniziali di prototipazione del codice, dove ne viene definita la struttura generale e vengono evidenziate le funzionalità da implementare.

Per questo, spesso l'SFC è usato anche come *linguaggio di specifica*.

## Linguaggio a Contatti

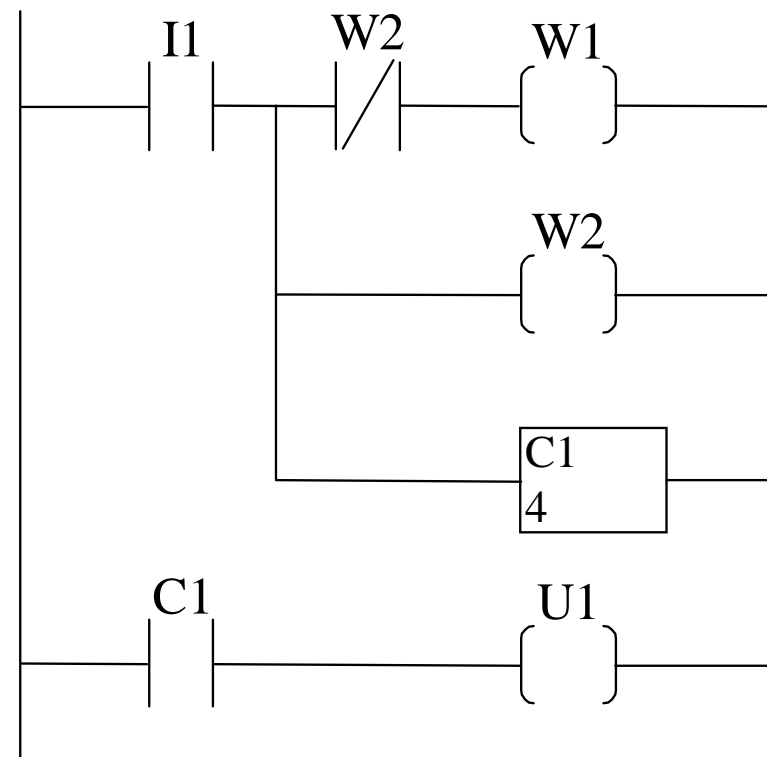
Il *Linguaggio a Contatti* o *Diagramma a Scala* (*Ladder Diagram, LD*) è derivato direttamente dagli schemi di controllo a relé elettromeccanici.

Concepito inizialmente per svolgere funzioni di logica binaria, è stato esteso per trattare numeri reali e controllo di processo.

E' un linguaggio a basso livello e come tale non particolarmente adatto alla programmazione diretta di sistemi complessi.

Tuttavia è importante perché è presente in tutti i PLC industriali ed è uno standard di fatto del mercato americano.

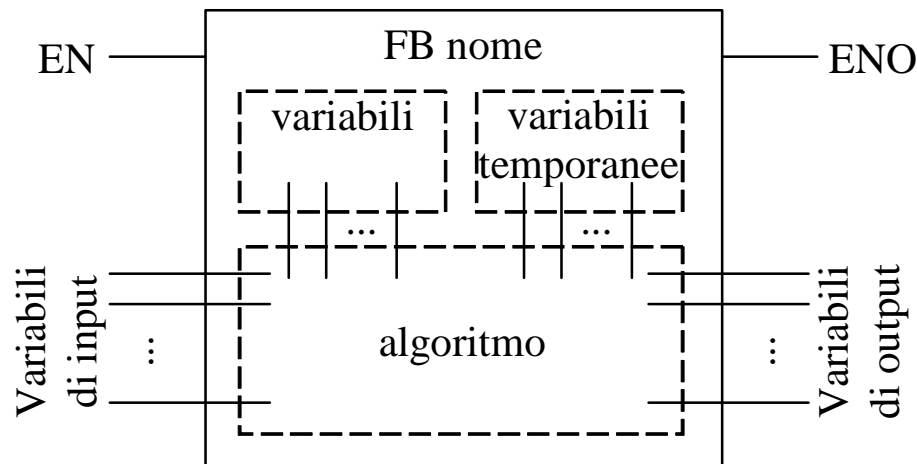
Esempio:



## Diagramma a Blocchi Funzionali

Il *Diagramma a Blocchi Funzionali* (*Function Block Diagram*, FBD) può essere utilizzato per esprimere il comportamento di funzioni, blocchi funzionali e programmi. E' analogo ai diagrammi di circuiti elettrici in cui le connessioni elettriche rappresentano i percorsi dei segnali tra i componenti.

Un blocco funzionale ha due caratteristiche principali, la definizione dei *dati* (ingressi e uscite) e un *algoritmo*, che viene lanciato ogni volta che viene eseguito il blocco. Esso processa i valori correnti dei parametri di ingresso e i valori delle variabili interne (locali o globali) e produce i nuovi valori dei parametri di uscita.



EN: segnale di enable  
(se ha valore logico falso il blocco non viene eseguito)

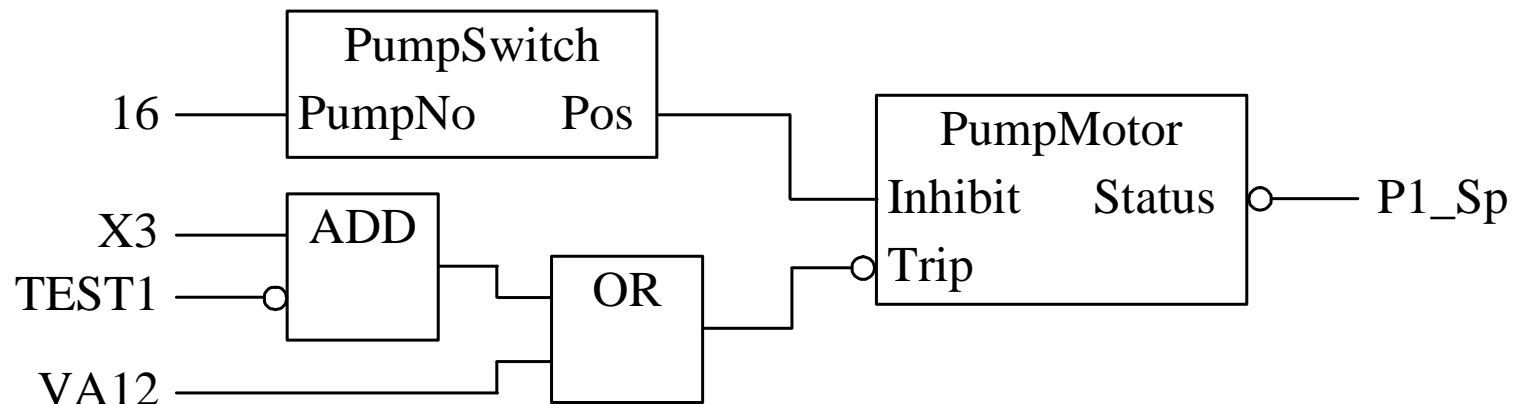
ENO: assume un valore vero quando il blocco ha terminato la propria esecuzione in modo corretto

Il flusso di segnale si considera per convenzione diretto da sinistra verso destra.

Regole di valutazione degli elementi di una rete FBD:

- ▶ nessun elemento della rete deve essere valutato prima che siano stati valutati i valori di tutti i propri ingressi
- ▶ la valutazione di un elemento della rete non è completa finché non sono stati valutati i valori di tutte le sue uscite
- ▶ la valutazione della rete termina quando tutte le uscite di tutti i suoi elementi sono state valutate

Esempio:



## Lista Istruzioni

La *Lista Istruzioni* (*Instruction List*, IL) è un linguaggio di basso livello simile all'ASSEMBLER. E' adatto per compiti molto specifici quali l'interfacciamento di hardware particolare. Anch'esso, come il LD, è disponibile per tutti i PLC.

Un'istruzione IL è composta da un'eventuale label (etichetta) seguita da un operatore, un'eventuale modificatore ed un operando.

Una riga di codice del tipo OP Operando va interpretata implicitamente come Accumulatore := Accumulatore OP Operando.

Esempio:

<i>Label</i>	<i>Operator</i>	<i>Operand</i>	<i>Comment</i>
	LD	Speed	(* Load Speed and *)
	GT	1000	(* Test if > 1000 *)
	JMPCN	VOLTS_OK	(* Jump if not *)
	LD	Volts	(* Load Volts and *)
	SUB	10	(* Reduce by 10 *)
	ST	Volts	(* Store Volts *)
VOLTS_OK:	LD	1	(* Load 1 and *)
	ST	%Q75	(* Store in output 75 *)

## Testo Strutturato

Il Testo Strutturato (Structured Text, ST) è un linguaggio testuale ad alto livello, simile al PASCAL.

Risulta particolarmente adatto per

- ▶ eseguire complesse elaborazioni matematiche
- ▶ eseguire test condizionali con molteplici alternative

operazioni che in LD o IL richiederebbero molte istruzioni (con salti), riducendo così la leggibilità del codice.

Esempio:

```
IF Speed > 1000 Then
    Volts := Volts - 10;
END_IF;
%Q75 = 1;
```