SISTEMI PER L'INDUSTRIA E PLC

SEZIONE 3

il PLC

LOGICHE BINARIE A RELAIS: SOLO CONTATTI E BOBINE

- **Nelle logiche binarie a relais si disponeva solo di contatto NA, concatto NC, e bobina**
  - **Struttura del segmento: logica ingresso -> assegnazione di uscita**
  - **"Se INGRESSO =1 allora USCITA = 1 altrimenti USCITA =0"**
  - **Contatto NA = interrogazione vera ("se INGRESSO =1")**
  - **Contatto NC = interrogazione negata ("se INGRESSO =0")**
  - **OR = parallelo, AND = serie**
  - **Le variabili si realizzavano con relè di appoggio**
  - **Le memorie in logica di Set Reset si realizzavano con schemi ad autoritenuta**
  - **I ritardi dei relais venivano sfruttati per generare segnali impulsivi (es. rilevatori di fronte)**

- **Nei PLC non c'è esecuzione concorrente e vi sono delle "facilitazioni"**
  - **Operatori di Set e Reset secondo la logica "se conditione allora operazione, altrimenti non fare nulla (memoria)"**
  - **Istruzioni di rilevazione di fronte**
  - **Istruzione NOT che nega il risultato logico precedente**
  - **… altro**

LOGICHE A RELAIS: E I SENSORI/ATTUATORI NON BINARI?

- **Nelle logiche a relais come si realizzavano semplici logiche di gestione sensori?**
  - **Esempio: fai partire il motore M se viene premuto START e se la temperatura T è inferiore 50°C**

- **Esistevano relais speciali che, invece della bobina, avevano altri tipi di ingressi (le uscite erano invece sempre contatti NA e/o NC).**
  - **Comparatori (due ingressi analogici A e B e il contatto NA si chiude se A>B)**
  - **Elementi di ritardo -timer- (un ingresso logico e uno numerico -soglia-; quando l'ingresso logico si abilita un orologio si avvia e il contatto NA si chiude dopo che l'orologio ha superato la soglia)**
  - **Elementi di conteggio -counter- (un ingresso logico e uno numerico -soglia-; quando l'ingresso logico si abilita e si disabilita un contatore interno si incrementa e il contatto NA si chiude dopo che il contatore ha superato la soglia)**
  - **….**

- **Con i PLC tutte queste limitazioni e artifici non sono più necessari**
  - **Un PLC è un calcolatore in grado di eseguire operazioni numeriche di ogni tipo**

PLC: NON SOLO BIT

- **Il PLC viene utilizzato per gestire sensori e attuatori "on/off" (finecorsa, lampade, abilitazioni, consensi,...)**
  - **Ingressi/uscite digitali e relative immagini di processo (IPI e IPU)**
  - **Logiche a bit, variabili a bit (merker)**

- **Se il PLC deve regolare una temperatura o dare un riferimento di velocità ad un azionamento, deve poter gestire numeri**
  - **Ingressi/uscite analogici e relative immagini di processo**
  - **Numeri (interi, reali), variabili e operatori associati**

- **Se il PLC deve organizzare una sequenza temporizzata, deve poter gestire numeri**
  - **Dopo 1,2 s dall'evento A attua per 0,3 s quindi attendi evento B al massimo per 3 s**

- **Ci sono poi operazioni banali che richiedono tipi di dati speciali**
  - **Esempio: ogni Lunedì che non sia una festività apri la porta alle 8:00 (tenendo conto dell'ora legale)**

LOGICHE TEMPORIZZATE

- **Come faccio lampeggiare una sirena?**
  …e a frequenza variabile?
- **Un disturbo sullo Start non deve far partire il motore**
  … ne il motore si deve fermare per un disturbo
  … ma dopo un po' si deve fermare (guasto allo Stop)

quindi, ad esempio

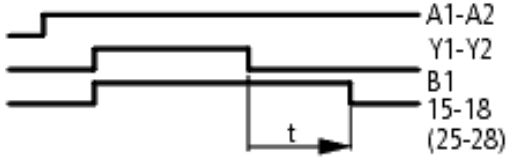
- **Gli ingressi devono essere filtrati**
- **Le uscite devono essere "temporizzabili"**
- **…**

Come si faceva con le logiche a relè?

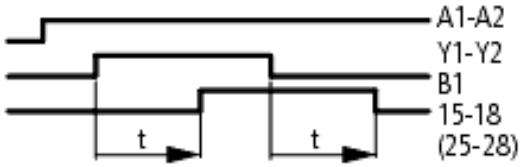
Il temporizzatore (timing relè)

- **I tempi selezionabili vanno da 50ms a 100h**
- **Varie tipologie e funzioni:**
  - **Ritardo all'inserzione (Funzione 11)**

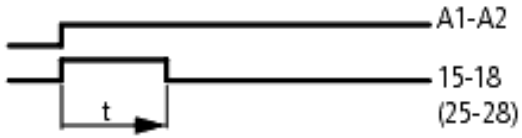
  - **Ritardo alla disinserzione (Funzione 12)**



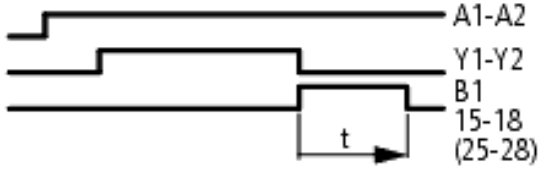
  - **Ritardo (Funzione 16)**



  - **Passante all'inserzione (Funzione 21)**

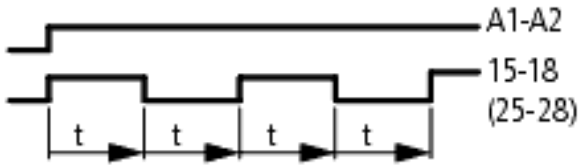


  - **Passante alla disinserzione (Funzione 22)**

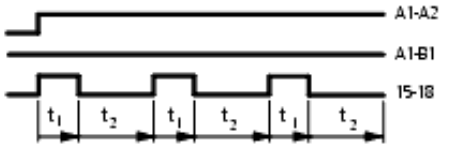


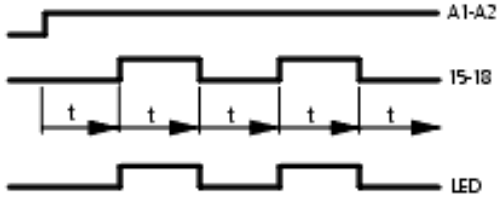
Il temporizzatore (timing relè)

- **Varie tipologie e funzioni:**
  - **Lampeggiante, inizio (Funzione 42)**

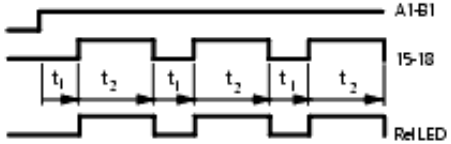


  - **Lampeggiante, pausa (Funzione 43)**

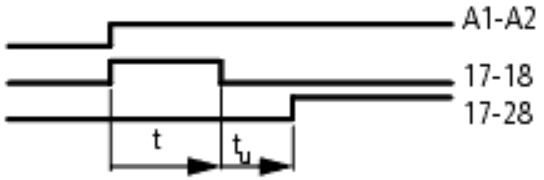




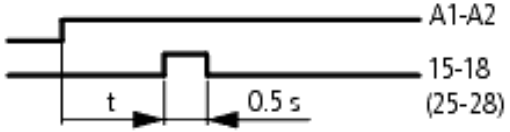
  - **Lampeggiante, 2 tempi (Funzione 44)**



  - **Stella-triangolo (Tu=50ms) (Funzione 51)**



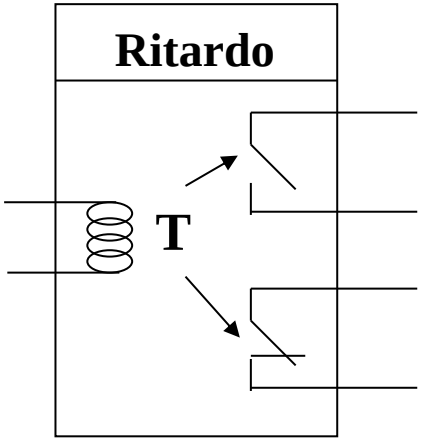
  - **Generazione impulsi fissi (Funzione 81)**

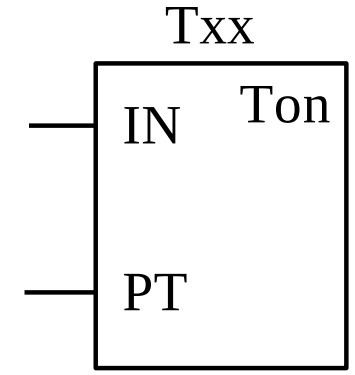


www.moeller.it/manuale/index.html

USO DEI TIMER

- **Tutte queste funzioni si possono realizzare mediante funzioni logiche e un modulo "ritardo"**
- **Il PLC implementa un solo modulo SW "Timer" che agisce come ritardo**
- **Come esprimo il ritardo PT?**
- **Esiste uno standard IEC61131 -> T#0s**
- **L'ingresso IN corrisponde alla bobina del temporizzatore. Nel rele' (vecchi PLC)**
  - **se IN=falso ->Txx=0**
  - **se IN=vero ->Txx si incrementa (con saturazione)**
  - **se Txx>PT allora Txx(contatto)=vero**





- **Nel PLC std IEC e' il fronte di salita di IN che avvia il timer**

IL TIPO DI DATO TIME

- **Il contenuto di un operando del tipo di dati TIME viene interpretato in ms. La rappresentazione comprende le indicazioni di giorni (d), ore (h), minuti (m), secondi (s) e millisecondi (ms).**

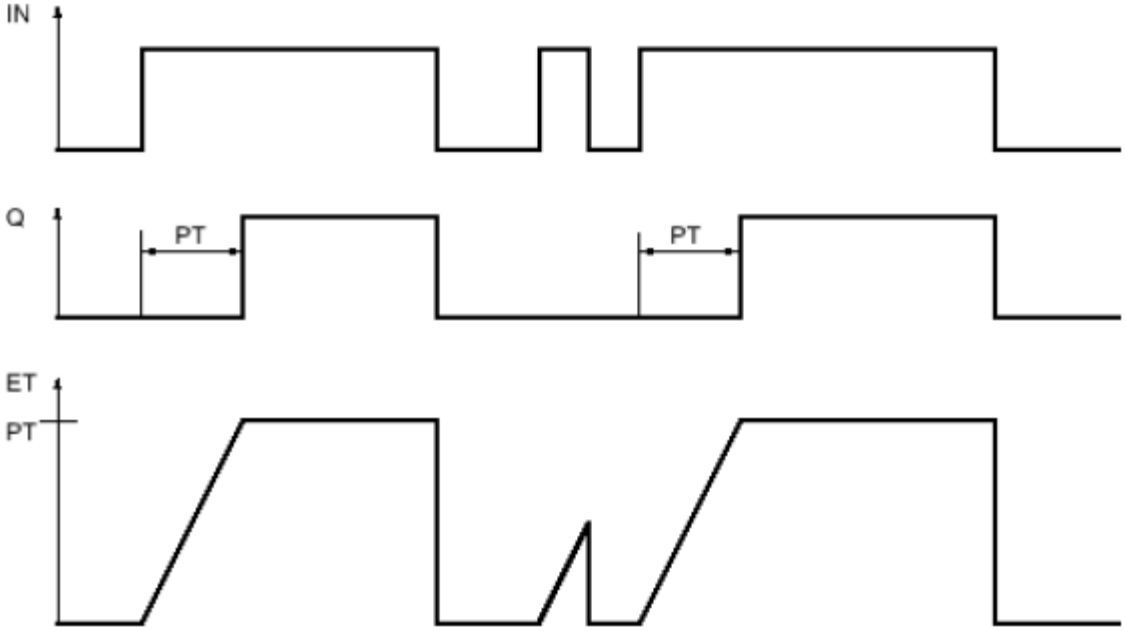
- **TIME viene espresso con 32 bit (integer, unità ms) nel formato con segno da T#-24d20h31m23s648ms a T#+24d20h31m23s647ms**

- **Esempi di rappresentazione:**
  - **T#10d20h30m20s630ms,**
  - **TIME#10d20h30m20s630ms**
  - **10d20h30m20s630ms**
  - **T#5h10s (Non è necessario indicare tutte le unità di tempo)**
  - **Non si devono eccedere i limiti (23 h, 59 m, 59 s o 999 ms).**

- **Un timer IEC è un oggetto con ingresso IN e uscita OUT booleane, il cui valore corrente ET e la cui costante di tempo PT sono espresso come TIME**

TIMER RITARDO ALL'INSERZIONE (es. filtro di abilitazioni)

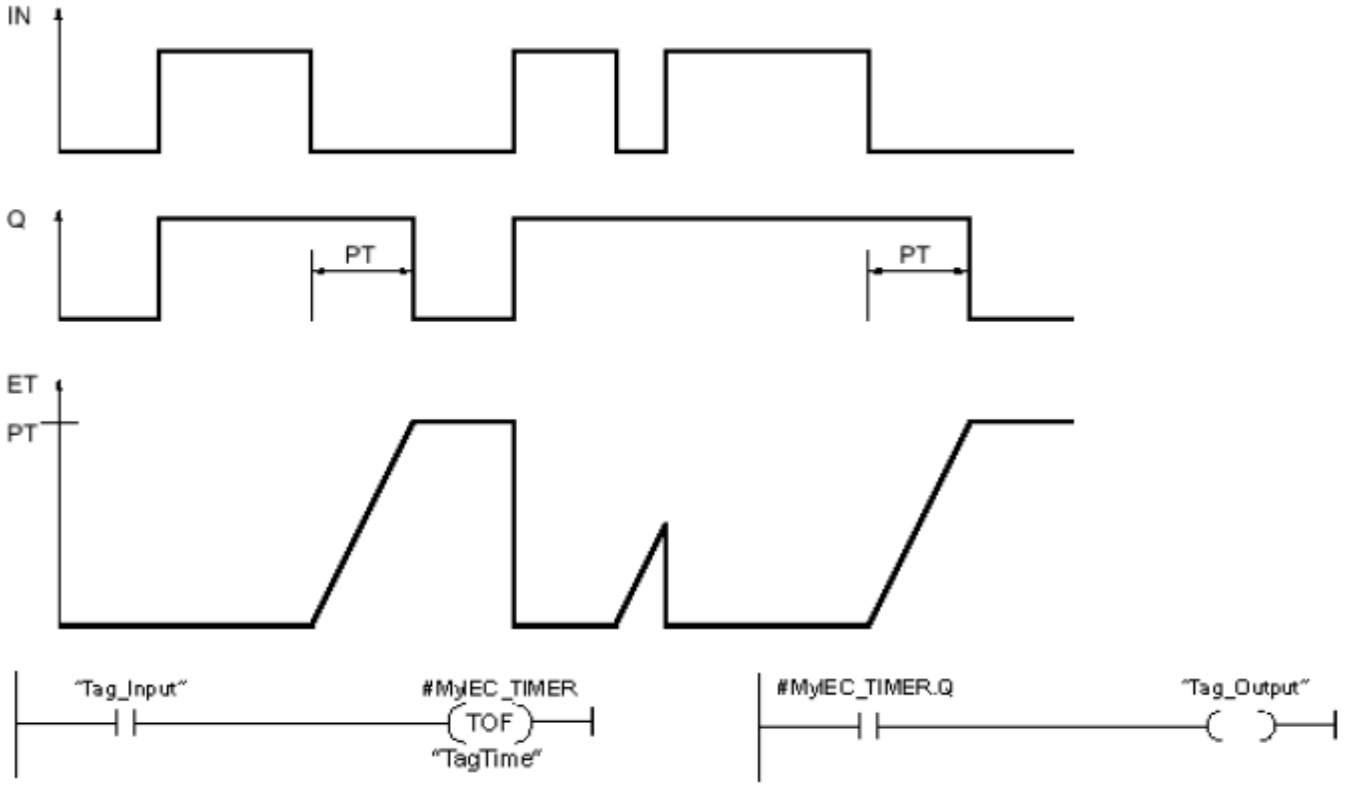
- **Se l'informazione presente su IN ha un fronte di salita, si attende PT quindi l'uscita va a 1 fino a quando IN non ha una commutazione a zero. La commutazione a zero di IN durante PT azzera il Timer**
  - **Istruzione "Avvia ritardo all'inserzione" –(TON)-**



Immagini prese da manuale Siemens

TIMER RITARDO DISINSERZIONE (es. “buchi di rete”)

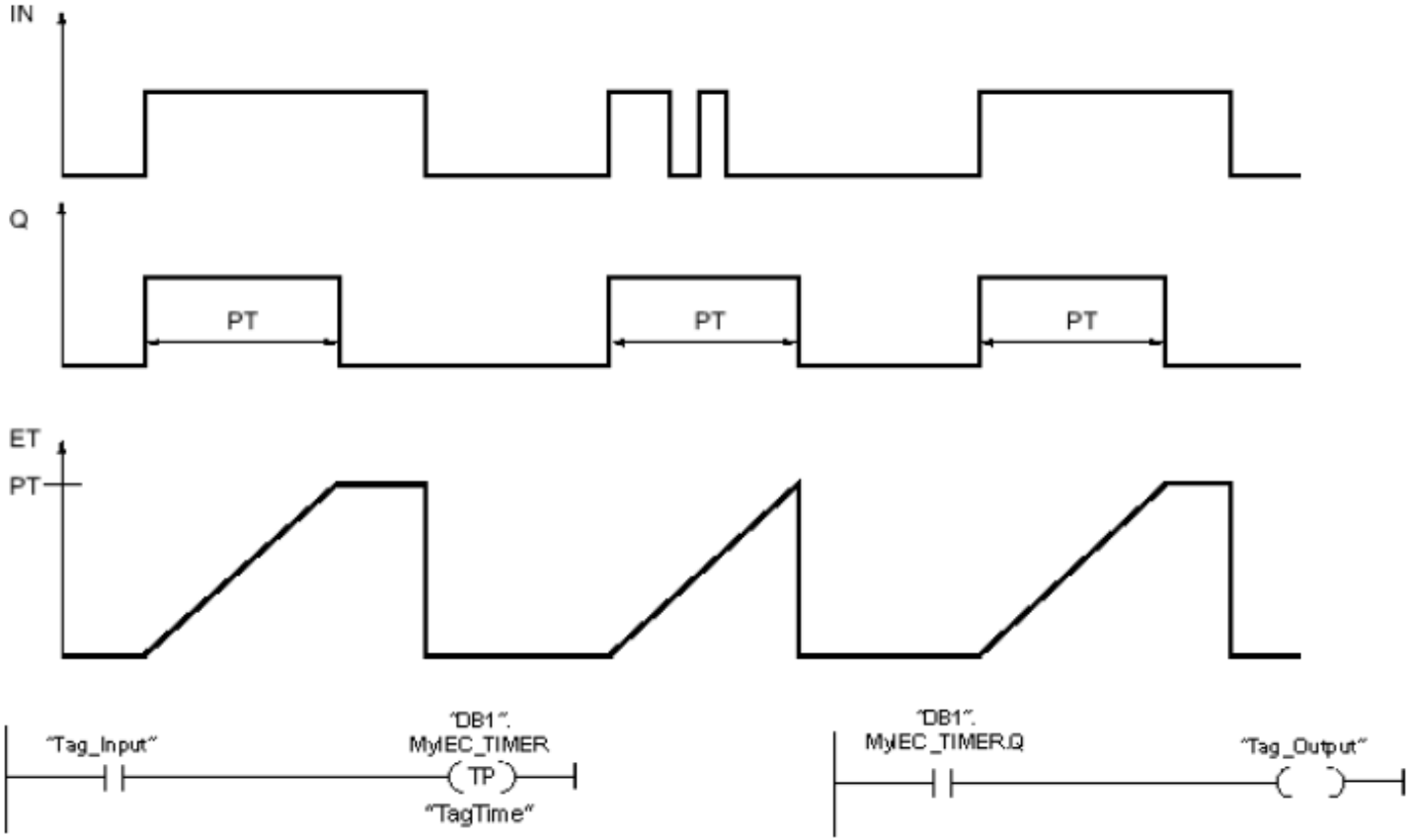
- **Se l’informazione presente su IN ha un fronte di salita l’uscita viene posta a 1; al fronte di discesa di IN, l’uscita rimane a 1 e vi rimane per un tempo PT quindi viene posta a 0. Un nuovo fronte di salita di IN fa ripartire il meccanismo**
  - **Istruzione “Avvia ritardo alla disinserzione” –(TOF)-**



Immagini prese da manuale Siemens

TIMER GENERAZIONE DI IMPULSI (es. timeout motore nastro)

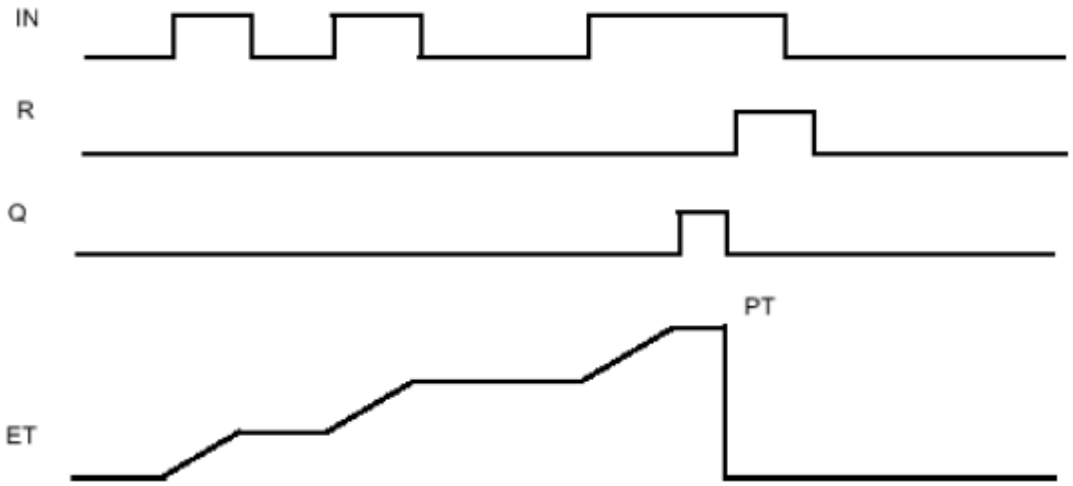
- **Se l'informazione presente su IN ha un fronte di salita, l'uscita viene impostata per una durata programmata PT indipendentemente da come evolve IN**
  - **Istruzione "Avvia temporizzazione quale impulso" –(TP)-**



Immagini prese da manuale Siemens

TIMER ACCUMULATORE TEMPORALE (es. durata utensile)

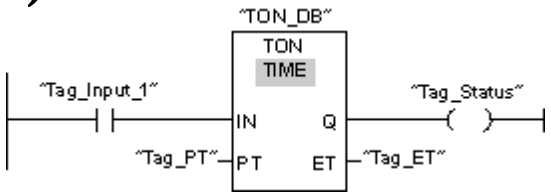
- **Se l'informazione presente su IN ha un fronte di salita, ET accumula i valori temporali in corrispondenza di IN=1 e l'uscita viene posta a 1 allo scadere di PT. Il segnale di reset R (booleano) resetta ET e Q**
  - **Istruzione "Avvia accumulatore temporale" –(TONR)-**

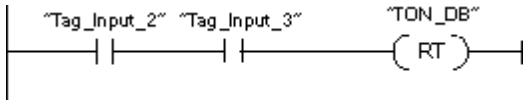


Immagini prese da manuale Siemens

OSSERVAZIONI SUI TIMER

- **I timer (e in particolare i timer con memoria) possono essere resettati mediante la bobina –(RT)-**





Immagini prese da manuale Siemens

- **La costante di tempo puo' essere sovrascritta –(PT)-**

- **I timer TON e TONR sono i piu' utilizzati**

- **Esercizio: il ritardo all'inserzione dei rele' corrisponde esattamente al timer TON?**
  - **Si. La rilevazione del fronte di salita di In di TON equivale alla rilevazione di In in quanto il fronte di salita attiva e il fronte di discesa disattiva (definizione dell'assegnazione)**

- **Esercizio: il ritardo alla disinserzione dei rele' corrisponde esattamente al timer TOF?**
  - **Si. Il fronte di discesa fa partire il timer e il fronte di salita lo resetta**

TIMER: RIASSUNTO

- **I timer dei PLC sono oggetti software (ce ne sono tanti, inutile risparmiare)**
- **Tutti i timer si attivano sul fronte di salita di In e si disattivano con il reset o con condizioni specifiche (*). L'uscita Out è a 0 se il timer non è attivo**

Tipo Timer	Il conteggio inizia	Il conteggio si ferma	Il conteggio si azzera (*)	Out è a 1 se timer attivo e	Il timer si disattiva
TON	Fronte salita In		Fronte discesa In	Se ET >= PT	Fronte discesa In
TOF	Fronte discesa In		Fronte salita In	Se ET < PT	Solo al reset
TP	Fronte salita In		Fronte discesa In & ET>=PT	Se ET < PT	Fronte discesa In & ET>=PT
TONR	Fronte salita In	Fronte discesa In	Solo al reset	Se ET >= PT	Solo al reset

ESERCIZI SUI TIMER

- **Esercizio: il ritardo ha un suo timer corrispondente?**
  - **No, ma è possibile ottenerlo per combinazione tra TON e TOF**
  - **Ritardo = TON + (TOF&!In)**

- **Esercizio: il passante all'inserzione è corrispondente al generatore di impulsi –(TP)-?**
  - **No. Infatti il passante all'inserzione va a zero se IN va a zero, mentre il generatore di impulsi si attiva sul fronte di salita di In anche se In va subito dopo a zero**
  - **La funzione passante all'inserzione si ricava da TON (In&!TON)**

- **Esercizio: come si ottiene il passante alla disinserzione?**
  - **A partire da TOF (TOF&!In)**

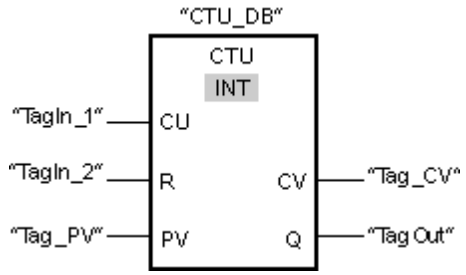
- **Esercizio: come si ottiene il lampeggiante?**
  - **A partire da TON o da TP…ma vedremo in laboratorio**

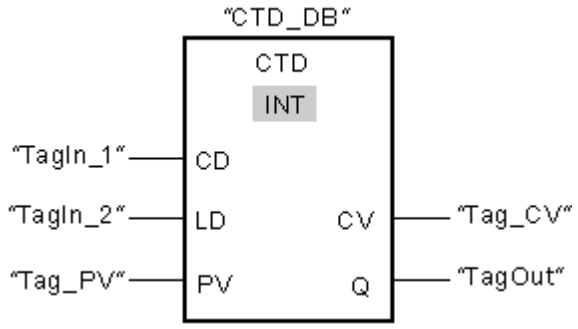
IL CONTATORE

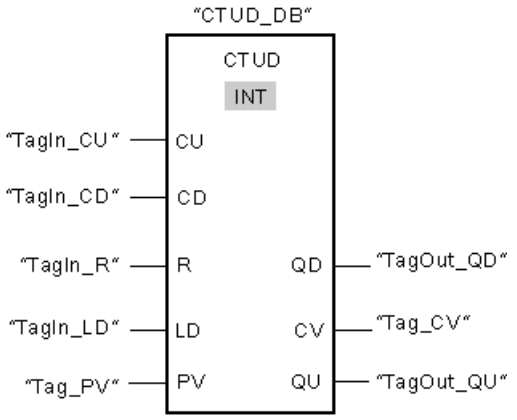
- **Il temporizzatore conta il tempo, il contatore conta eventi**

Immagini prese da manuale Siemens

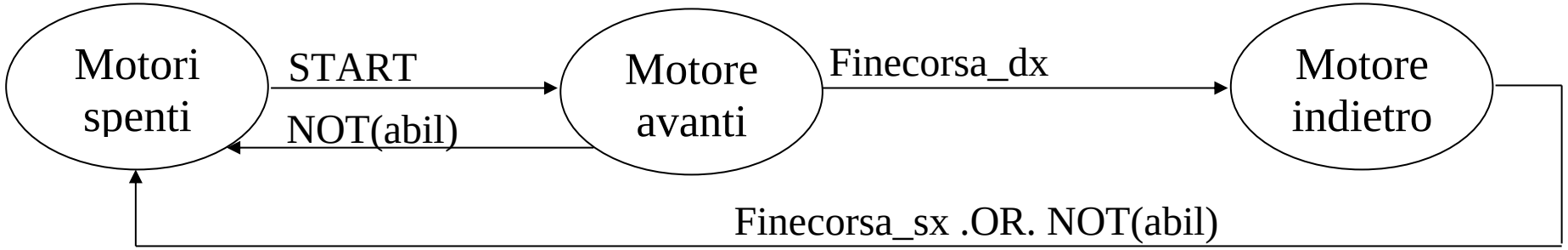
- **Esistono tre tipi di contatori: abanti, indietro, up/down (Es. CTU –avanti-)**
  - **Evento = fronte di salita su CU; PV = soglia del contatore (max INT = 32767)**
  - **Il valore corrente CV del contatore si incrementa fino a maxINT e si blocca**
  - **Applicando 1 all'ingresso R di reset il valore CV si azzera inibendo CU**
  - **L'uscita binaria Q indica se PV>=CV**
  - **Contatore down (CTD): CV si decrementa ad ogni evento fino a raggiungere la soglia (=0) (minINT=-32768). Applicando 1 su LD si carica CV al valore PV**
  - **Contatore up/down (CTUD): +1-1=0**







FUNZIONI SEQUENZE AUTOMATICHE



- **Metodo dei flag (implementabili con i merker):**
  - **flag1 (stato di attesa dello START)**
  - **flag2 (stato di motore_avanti e attesa del finecorsa_dx)**
  - **flag3 (stato di motore_indietro e attesa del finecorsa_sx)**
- **Si definiscono delle reti combinatorie per il passaggio tra gli stati**
  - **Se flag1 .AND. START .AND. abil -> res(flag1),set(flag2),res(flag3)**
  - **Se flag2 .AND. finecorsa_dx -> res(flag2), set(flag3), res(flag1)**
  - **Se flag3 .AND. finecorsa_sx -> res(flag3), set(flag1), res(flag2)**
  - **Se .NOT. abil -> res(flag2), res(flag3), set(flag1)**
  - **Se flag1 -> spegni motori**
  - **Se flag2 -> motore avanti**
  - **Se flag3 -> motore indietro**

PASSI

TRANSIZIONI

- **Nota: Potrebbe esserci più di una transizione alla volta, creando transizioni che non ci sono**

SEQUENZE AUTOMATICHE (2)

- **START:motore_avanti fino al finecorsa_dx, poi motore indietro fino al finecorsa_sx**
  - **Metodo dei flag (implementabili con i merker):**
    - **flag1 (stato di attesa dello START)**
    - **flag2 (stato di motore_avanti e attesa del finecorsa_dx)**
    - **flag3 (stato di motore_indietro e attesa del finecorsa_sx)**
  - **Metodo del semaforo (flag0):**

    Si definisce un flag (flag0) che indica un passaggio di stato
    - **Set flag0 // inizialmente il semaforo è verde**

    Transizioni
    - **Se flag0.AND.flag1.AND.START.AND.abil -> res(flag0,flag1,flag3), set(flag2)**
    - **Se flag0.AND.flag2.AND.finecorsa_dx -> res(flag0,flag1,flag2), set(flag3)**
    - **Se flag0.AND.flag3.AND.finecorsa_sx -> res(flag0,flag2,flag3), set(flag1)**
    - **Se .NOT. abil -> res(flag0,1,2,3)**

    Passi
    - **Se flag1 -> spegni motori**
    - **Se flag2 -> motore avanti**
    - **Se flag3 -> motore indietro**

SEQUENZE AUTOMATICHE (3)

- **START:motore_avanti fino al finecorsa_dx, poi motore indietro fino al finecorsa_sx**
  - **Metodo delle variabili (implementabili con i merker), simile al sistema IPI e IPU:**
    - **Stato (variabile che può assumere valori 0, 1, 2). Non varia per tutte le transizioni**
    - **Stato_futuro (variabile "immagine di stato" che aggiorno solo alla fine)**
    - **In questo modo eseguo solo una transizione per ogni ciclo**

  - **Metodo della variabile di appoggio (Stato_futuro):**

    Transizioni
    - **Se (Stato=0).AND.START.AND.abil -> Stato_futuro = 1**
    - **Se (Stato=1).AND.finecorsa_dx.AND.abil -> Stato_futuro = 2**
    - **Se (Stato=2).AND.finecorsa_sx.AND.abil -> Stato_futuro = 0**
    - **Se .NOT. abil -> Stato_futuro = 0 (transizione più prioritaria)**
    - **Stato = Stato_futuro**

    Passi
    - **Se (Stato=0) -> spegni motori**
    - **Se (Stato=1) -> motore avanti**
    - **Se (Stato=2) -> motore indietro**

PROGETTO DELLE SEQUENZE AUTOMATICHE

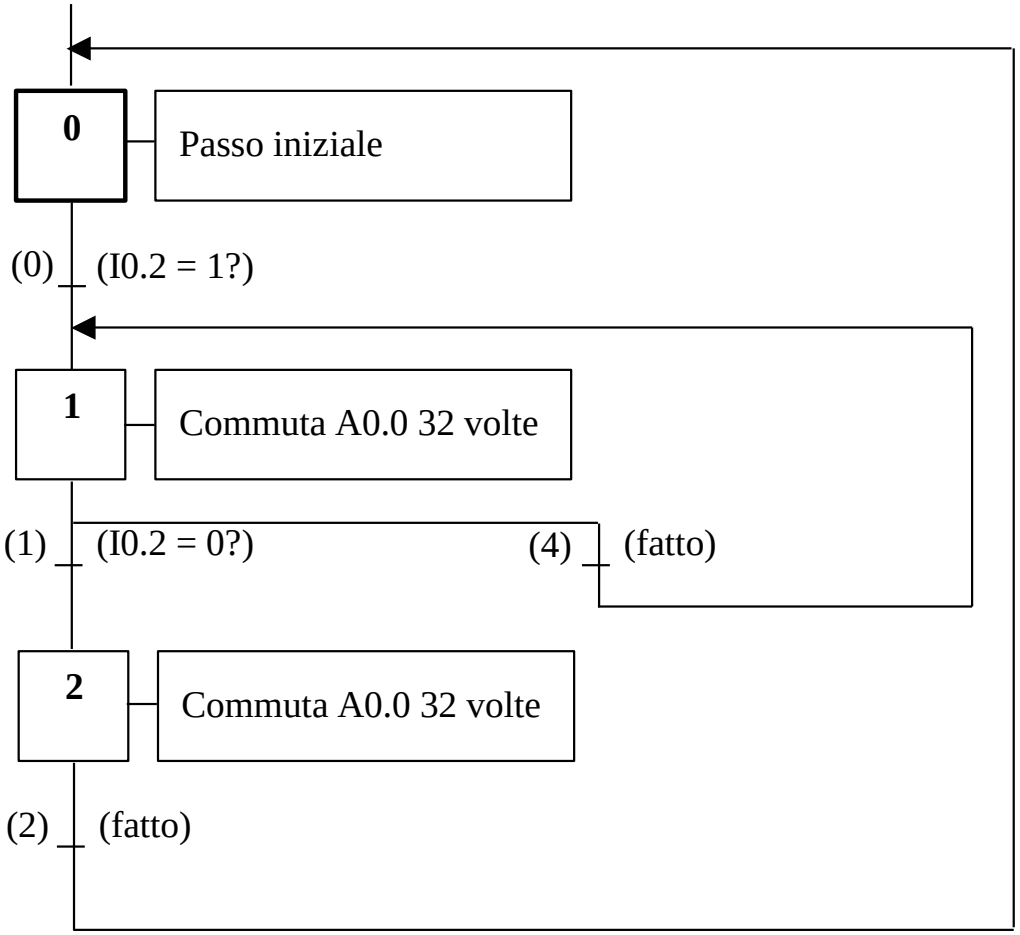
- **La vera fase di progetto è il disegno della sequenza e l'identificazione degli STATI (passi e relative azioni) e delle TRANSIZIONI**
  - **Due tipi di transizioni:**
    - **Condizionate: hanno uno stato di partenza e uno stato di arrivo**
    - **Incondizionate: hanno solo lo stato di arrivo e normalmente sono prioritarie rispetto a quelle condizionate (sono gli ultimi segmenti della parte "transizioni")**

- **Fasi realizzative a valle del progetto della sequenza in termini di passi e transizioni**

1. **Dichiarare le variabili che servono**
2. **Predisporre il programma con una parte di TRANSIZIONI, il passaggio di stato (allineamento tra Stato e Stato_futuro), e una parte di PASSI**
3. **Numerare gli N STATI e realizzare la relativa parte di programma instanziando gli N segmenti necessari e popolandoli con le azioni da fare per ogni stato**
4. **Numerare le M TRANSIZIONI e realizzare la relativa parte di programma instanziando gli N segmenti necessari e popolandoli con le interrogazioni necessarie per lo svolgimento della transizione**

LINGUAGGI ORIENTATI ALLE SEQUENZE

- **Sequenze ordinate di "passi" e "transizioni" (Es. Grafcet)**
  - **Passi (quadrati): le azioni da eseguire**
  - **Transizioni (croci): le condizioni di test per proseguire**



LINGUAGGI TRADIZIONALI: PROBLEMATICHE

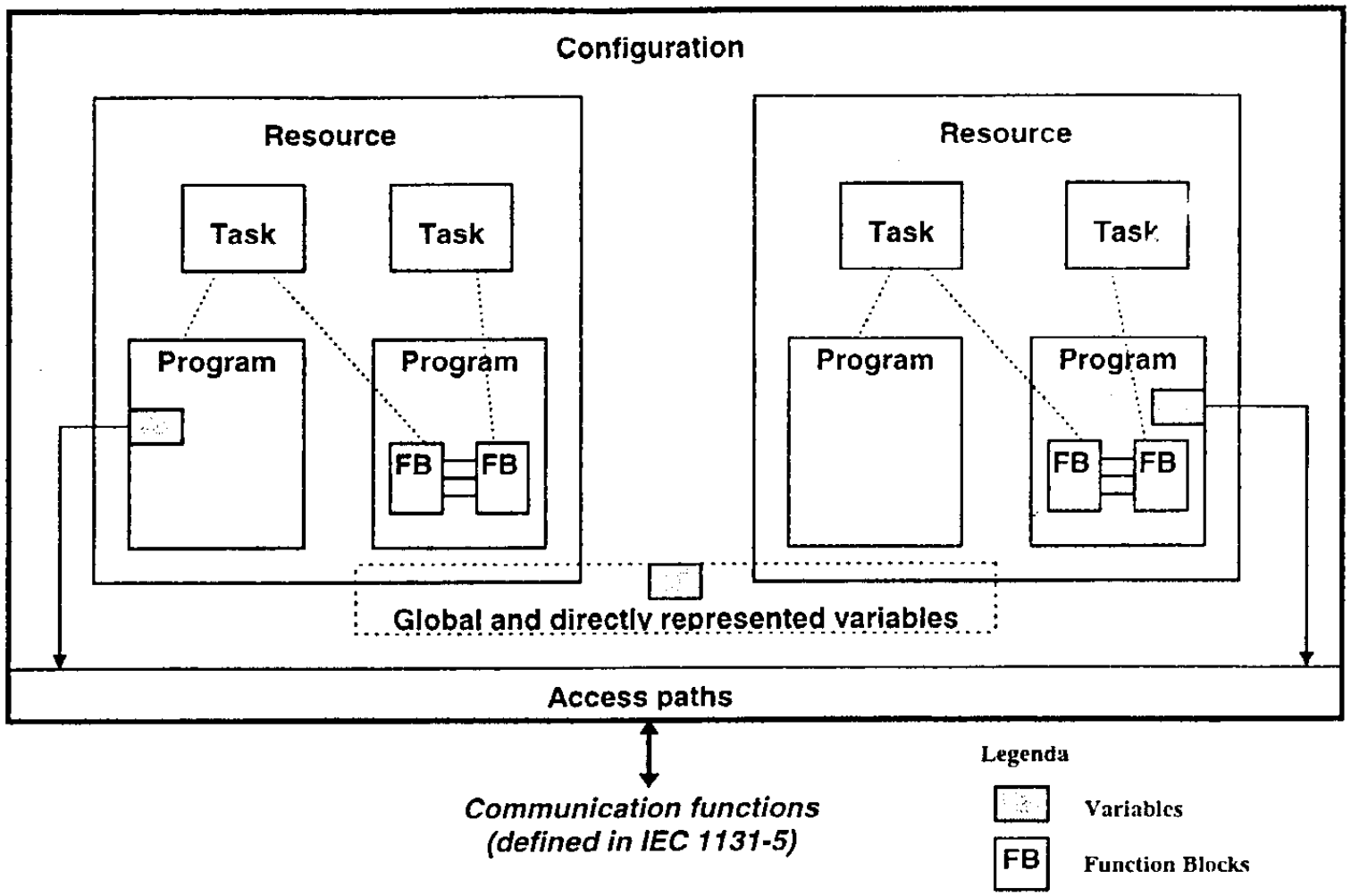
- **Legati alla formazione culturale degli "addetti ai lavori"**
- **Limitati all'implementazione di applicazioni semplici**
- **Mancanza di strutturazione (un progettista per ogni programma)**
- **Assenza di metodologia di progetto del SW**
  - **Lunghi tempi di sviluppo**
  - **Povertà di documentazione e quindi scarsa affidabilità**
- **Notevole diversità tra i linguaggi proposti dai diversi costruttori**
  - **Assenza di strumenti di sviluppo adeguati ad applicazioni complesse (simulatori)**
  - **Difficoltà negli "upgrade" di impianto (il SW risulta troppo legato all'HW)**

⇒ **Necessità di uno standard**

CARATTERISTICHE DELLO STANDARD IEC/CEI61131

- **Sviluppo strutturato del software**
  - **Approccio bottom-up o top-down**
  - **Suddivisione del programma in elementi funzionali (POU) clonabili e riusabili**
- **Strong data typing**
  - **Verifica della coerenza tra tipo di dato e assegnazione e/o operazione**
- **Pieno controllo dell'esecuzione**
  - **Suddivisione del programma in task con differenti tempi di scansione**
- **Supporto alla descrizione delle sequenze**
  - **Linguaggio SFC (Sequenzial Flow Chart)**
- **Supporto alla gestione di strutture dati (es. record) -> integrazione (database)**
- **Coesistenza di più linguaggi (5) per la descrizione di un'applicazione**
- **Portabilità del software tra PLC di costruttori differenti**

IEC 61131-3: STANDARD DI PROGRAMMAZIONE PER PLC



IEC 61131-3: STRUTTURAZIONE DEL CODICE

- **Configuration**
  Puo' essere inteso come il software relativo ad un PLC (In IEC 1131-5 sono definite le modalita' di comunicazione tra piu' configuration –access paths-)

- **Resource**
  Puo' essere inteso come il software relativo ad una CPU

- **Task**
  Motore di attivazione di piu' parti (es. Function blocks) di piu' POU di una resource
  - **esecuzione su evento (SINGLE:0->1) o periodica (TIME) con priorità (zero task ⟶ i program vengono eseguiti secondo una struttura sequenziale)**

- **Program (POU= Program Organisation Unit)**
  E' un modulo e cioe' un insieme organizzato di:
  - **Function (sistemi a piu' ingressi e una uscita privi di stati interni)**
  - **Function block (procedure con parametri di ingresso e uscita e variabili locali –Es. PID,… -)**

IEC 61131-3: Program Organization Unit (POU)

- **Program (OB in Tia Portal)**
  Contenitori di costrutti eseguibili scritti nei linguaggi previsti dallo standard.

- **Standard Function (FC in Tia Portal)**
  - **Conversione di tipo (real_to_int, trunc, BCD_to_int)**
  - **Numeriche (ADD, SUB, MUL, MOD, SQRT, MOVE,….)**
  - **Booleane (AND, OR, NOT, XOR)**
  - **Gestione di stringhe di bit (ROR, ROL, SHL, SHR)**
  - **Gestione di stringhe di caratteri (Insert, Lenght,…)**
  - **Gestione di dati….**

- **Standard Function blocks (FB in Tia Portal)**
  - **Elementi bistabili (Sr, sR, semafori)**
  - **Rilevatori di fronte positivo e negativo**
  - **Contatori (up, down e up-down)**
  - **Timer (Temporizzazine di impulso, con attivazione/disattivazione ritardata)**
  - **Comunicazione (IEC1131-5)**

IEC 61131-3: Blocchi dati

- **I blocchi dati e symbol table**
  - **La symbol table è prevalentemente utilizzata per la mappatura di ingressi e uscite. Prevede anche la mappatura dell'area M (merker) che, analogamente a quanto accade per le immagini di processo di ingressi e uscite, è organizzata a byte e ad indirizzamento geografico (ad esempio non supporta Array).**
  - **I blocchi dati sono insiemi di variabili "riallocabili"**

- **Blocchi dati di istanza**
  - **Sono blocchi dati associati agli FB che quindi vengono "replicati" ogni qual volta si richiama l'FB**
  - **Si pensi alla struttura dati di un timer, si tratta di un blocco dati di istanza (es. DB12)**
    - **ET, variabile di uscita di tipo Time (valore corrente del timer)**
    - **Q, varibile Bool, uscita del Timer che indica se ET>=PT**
    - **PT, variabile d'ingresso di tipo Time (soglia di tempo)**
    - **IN, variabile di ingresso di tipo Bool (abilita il conteggio del tempo)**
    - **R, variabile di ingresso di tipo Bool (azzera il valore corrente ET del timer)**

IEC 61131-3: Blocchi dati, FC e FB

- **Variabili e blocchi dati a singola istanza**
  - **E' possibile usare una variabile dichiarata nella symbol table all'interno di un FC o di un FB, tuttavia questa pratica limita la riusabilità dell'FC o dell'FB**
  - **E' possibile dichiarare dati di ingresso, di uscita, di InOut in FC o FB: negli FC ingressi ed uscite saranno prive di memoria, mentre negli FB si tratterà di dati di istanza, ossia parte di una struttura dati replicata ogni volta che si richiama l'FB**
  - **I blocchi dati a singola istanza hanno un'identificazione assoluta. Ad esempio, ogni volta che si usa un timer si crea un DBxy diverso per ogni timer.**
  - **E' possibile usare un timer a singola istanza all'interno di un FB, tuttavia quando si richiama l'FB più di una volta il risultato è uguale a quello di dichiarare due volte lo stesso timer, ossia il timer non funziona.**

- **Blocchi dati multi-istanza**
  - **All'interno degli FB è possibile dichiarare dei dati "static" che arricchiscono i dati di istanza. Tra i dati static è possibile dichiarare dei timer (IEC_TIMER), ma si deve trattare di timer multi-istanza, in modo che la struttura dati associata al timer non sia un blocco dati di tipo DBxy, ma faccia parte del blocco dati di istanza dell'FB.**

IEC 61131-3: ALTRE CARATTERISTICHE

- **Ricco insieme di tipi di dati elementari, tra i quali strutture complesse (matrici)**
  - **Bit, interi con e senza segno (8, 16, 32, 64 bit), reali (32, 64 bit)**
  - **Altri tipi di dato elementari definiti dall'implementazione (data, ora, tempo,..)**
  - **Tipi di dati generici (ANY)**
- **Vari linguaggi supportati per la strutturazione interna di un POU**
  - **Ladder Diagram (Es. KOP)**
  - **Instruction List (Es. AWL)**
  - **Function Block Diagram (Es. FUP)**
  - **Structured Text (Linguaggio ad alto livello tipo PASCAL o BASIC)**
  - **Sequential Function Charts**
    (diagrammi a passi tipo flow charts caratterizzati da
    - **Steps (stati delle sequenza di controllo)**
    - **Actions (più azioni contemporanee possono essere assegnate ad un passo)**
    - **Transitions (passaggi tra passi)**

    Nota: la temporizzazione di ciascuna azione è fissata mediante qualificatori
- **Alcune caratteristiche (Es. mappa I/O) non sono definite (vincoli hardware)**

MEMORIE DELLA CPU

- **Memoria ROM/EPROM:**
  - **sistema operativo**

- **Memoria EEPROM/Flash:**
  - **programma applicativo**
  - **parametri di configurazione della CPU**
  - **area variabili di "backup"**

- **Memoria RAM:**
  - **(programma applicativo)**
  - **variabili del sistema operativo**
  - **immagini di processo**
  - **immagini dell'I/O analogico**
  - **variabili di programma (merker, variabili,...)**
    (accessibili a bit, byte, word –indirizzi pari- double word –indirizzi mod. 4-)
  - **temporizzatori (timer), contatori (counter)**

MEMORIA RAM: immagini di I/O

- **Immagini di processo o dell'I/O logico:**
  - **area RAM organizzata a vettore di byte**
  - **accesso al bit 5 del byte 2 della IPI I2.5 (I=ingresso Q=uscita)**
  - **l'area riservata all'I/O logico è in genere ad allocazione esclusiva**

- **Immagini dell'I/O analogico:**
  - **area RAM organizzata a vettore di byte**
  - **oggi convertitori a 12 o 16 bit (accesso alla word)**
  - **l'area riservata all'I/O analogico può essere utilizzata anche per moduli funzionali**

- **La dimensione dell'area RAM delle immagini limita il numero degli I/O fisici (indipendentemente dalle possibilità HW)**

MEMORIA RAM: variabili e tipi di dati a disposizione dell'utente

- **Bit, stringhe di bit (bit array), byte (8 bit), word (16 bit), Dword (32 bit)**
  - **operazioni prevalentemente logiche**
- **Interi short (S, 8 bit), normali (16 bit) e Double (D, 32 bit) concepiti con segno e Interi senza segno (U))**
  - **operazioni aritmetiche**
- **Numeri Reali $(-1)^s 1.m \cdot 2^e$ (32bit, s,8e,23m) e in precisione estesa (L, 64 bit, s,11e,52m)**
- **… altri tipi di dati (es. TIME, 32bit oppure CHAR, 8bit, ecc.)**
- **Aree di I/O (P=periferia, I=Ingressi, Q=uscite)**
  - **normalmente non soggetta a ritenzione**
- **Area M merker (variabili globali, es. "flag" o numeri), soggetta a ritenzione (backup)**
  - **BD (Blocco dati), DI (Blocco dati di Istanza) (IEC61131 blocchi OB, FB, FC, DB)**
- **Area L (dati locali)**
  - **area V, dati locali precedenti, non soggetta a ritenzione**
- **Possibili aree SM (es Special Merker, indicatori di ciclo, bit che oscillano, tools,..)**

MEMORIA RAM: contatori e temporizzatori

- **Temporizzatori (timer)**
  - **elementi che conteggiano il tempo**
  - **partono da zero e, se abilitati, ogni dt incrementano il proprio valore fino al valore di soglia**
  - **possono essere azzerati, fermati, letti,...**
  - **Spazio RAM di un timer:**
    - **Valore corrente ET (32-bit TIME che indica il valore di conteggio del tempo)**
    - **Costante di tempo PT (32-bit TIME che indica il valore di tempo da contare)**
    - **Bit di ingresso IN (bit di abilitazione al conteggio)**
    - **Bit di temporizzazione Q: indica se il timer ha raggiunto l'obiettivo; viene utilizzato in operazioni logiche ("dopo che è passato questo tempo fai...)**

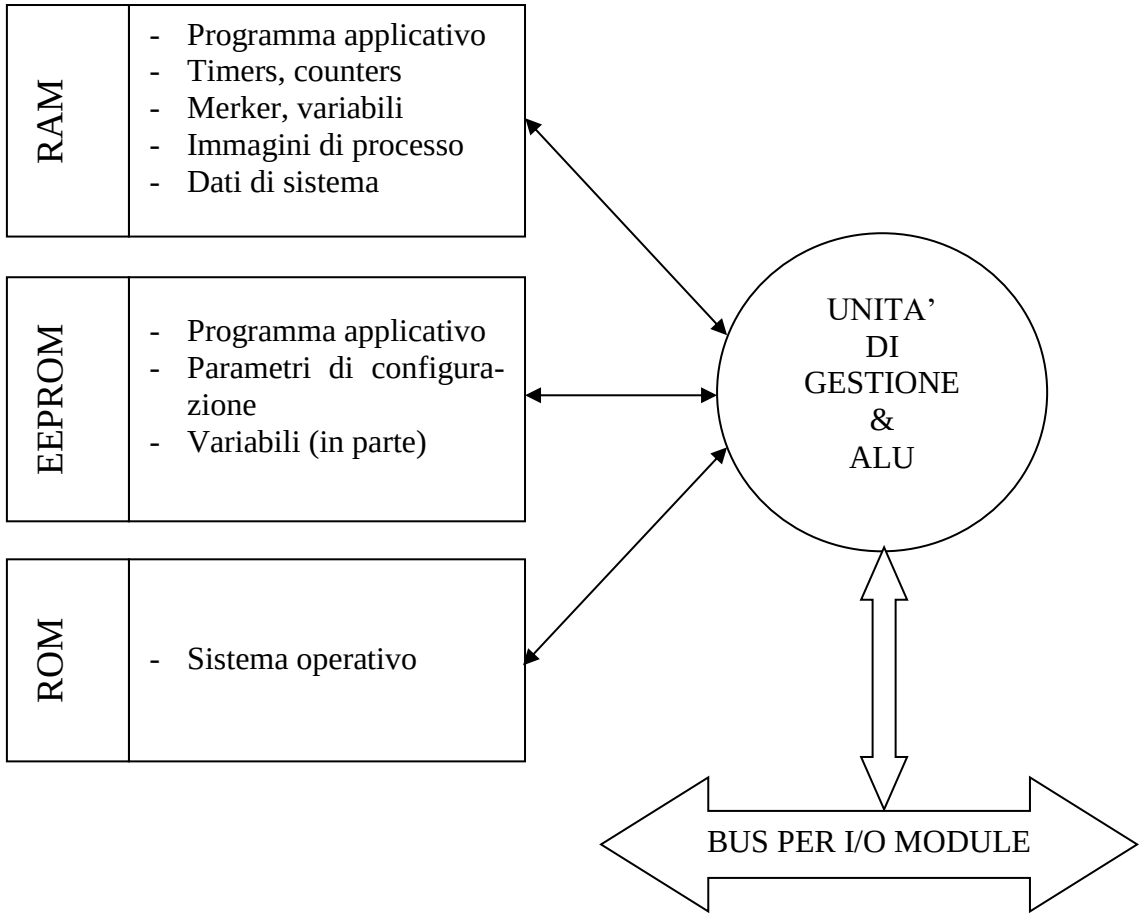
MEMORIA RAM: contatori e temporizzatori

- **Temporizzatori (timer) = elementi che conteggiano il tempo**

- **Contatori (counter)**
  - **elementi che conteggiano eventi (es. gli impulsi di un segnale esterno)**
  - **si precaricano ad un valore (costante di tempo) e, se abilitati, ad ogni evento decrementano il proprio valore fino a zero (oppure partono da zero e...)**
  - **possono essere azzerati, fermati,...**
  - **possono essere di tipo "avanti", "indietro" o "bidirezionali"**
  - **Spazio RAM di un counter:**
    - **Valore corrente CV (16-bit integer che indica il valore di conteggio)**
    - **Costante di tempo PV (16-bit integer che indica il valore da contare)**
    - **Bit di ingresso CU (bit di abilitazione al conteggio in salita)**
    - **Bit di ingresso CD (bit di abilitazione al conteggio in discesa)**
    - **Bit di temporizzazione QU (e QD): indica se il counter ha raggiunto l'obiettivo; viene utilizzato in operazioni logiche ("dopo quel numero di eventi fai...)**
    - **Bit di reset R (azzera), bit di Load LD (carica un valore)**

IL MODULO CPU

- **ALU (Arithmetic Logic Unit)**
- **I/O bus generator**
- **Memorie**
- **Alimentazione**
- **I/O locale**
- **Interfacce di comunicazione**
- **Interfacce diagnostiche**



UNITA' DI GESTIONE: unità aritmetico-logica

- **Più unità aritmetiche e logiche (interi, reali, booleani,..)**

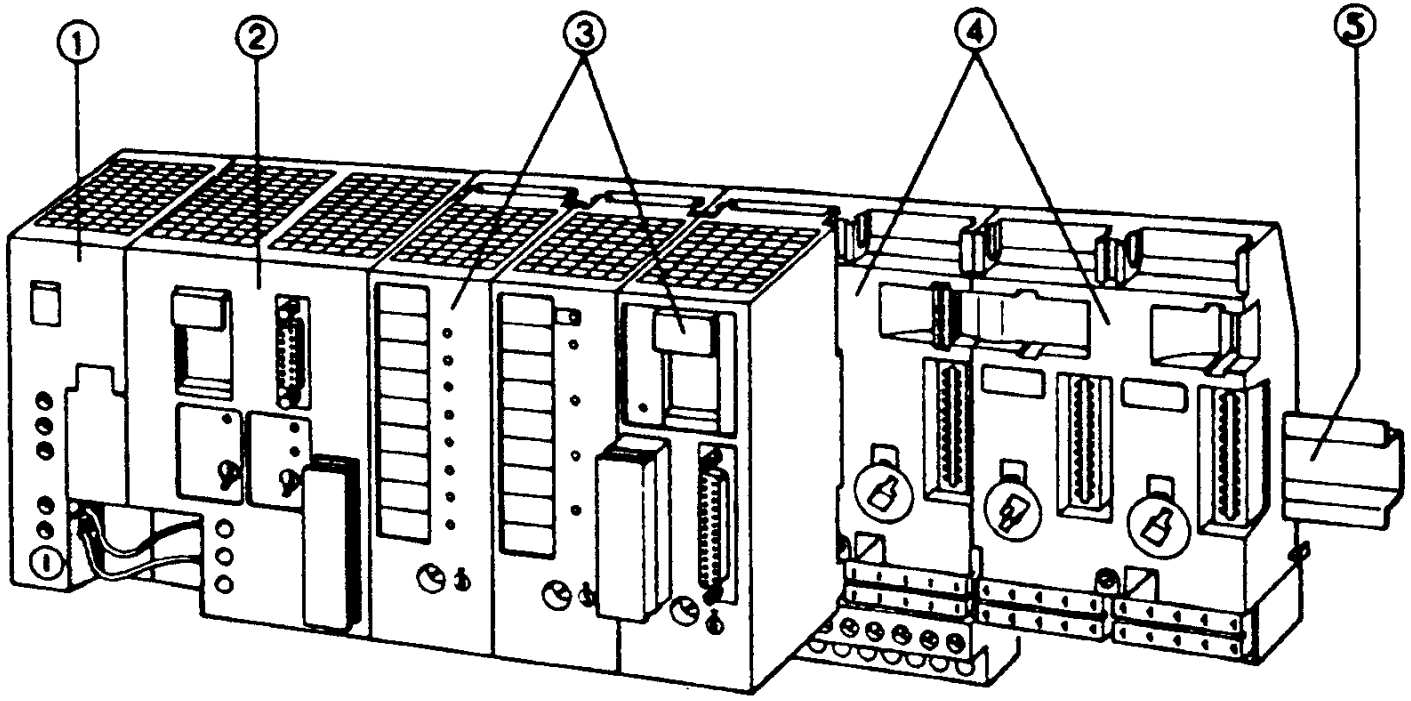
- **RLC (Risultato Logico Combinatorio) = accumulatore dell'unità logica**
  - **Istruzioni del tipo RLC := RLC .operatore. operando**

- **OUT1 := (A.AND.B.AND.C).OR.(D.AND.(E.OR.F))**

```
LD  A    Carica A nell'RLC, ossia RLC:=A
AND  B   Eseguo l'AND con B, ossia RLC:=RLC&B
AND  C   Eseguo l'AND con C, ossia RLC:=RLC&C
OR (D    OR=>stack operatori, RLC=>stack dati(=A&B&C), RLC:=D
AND  (E  AND=>stack operatori, RLC=>stack dati(=D), RLC:=E
OR F     Eseguo l'OR con F, ossia RLC:=RLC+F
)        RLC:=RLC.(stack operatori).(stack dati), ossia RLC:=(E+F).AND.D
)        RLC:=RLC.(stack operatori).(stack dati), ossia RLC:=RLC.OR.(A&B&C)
```

Nota: questo approccio coincide con il programma in linguaggio IL

PLC: ELEMENTI INDISPENSABILI

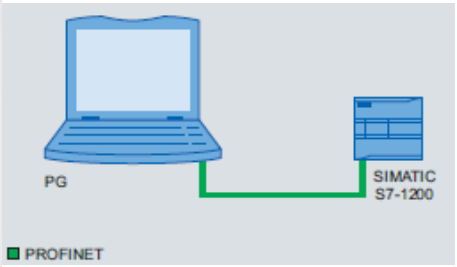


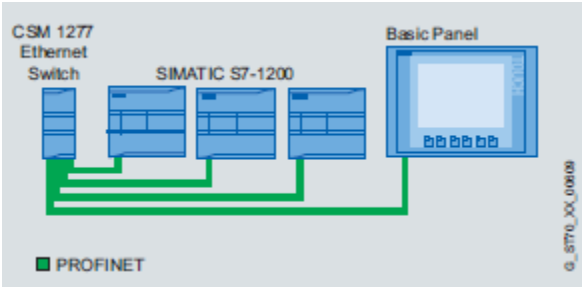
1) Alimentatore
2) CPU
3) Unita' periferiche (I/O logico, I/O analogico, controllo assi,…)
4) Moduli di bus con punto di attacco
5) Guide profilate normalizzate DIN

PLC COMPATTI

- **Monoblocchi di diverse taglie (90/90/110mmx100mmx75mm), ~10W**
- **Compatibilità IEC61131 (Ladder, FBD, STL)**
- **Rapida esecuzione di istruzioni (logica 0.1us, INT 2us, REAL 3us per istruzione)**
- **Siemens Simatic S7-1200 (Esempio CPU 1215C)**

Immagini prese da manuale Siemens





- **2 Interfacce Profinet (Ethernet + Stack) (programmazione, diagnostica, TCP/IP)**
- **14 ingressi e 10 uscite digitali**
- **1024 IPI + 1024 IPU**
- **2 ingressi analogici 0-10V (10bit, 625us)**
- **2 uscite analogiche 0-20mA (10bit)**
- **4 Pulse Time Output (fino a 100kHz)**
- **PWM fino a 100kHz**
- **6 contatori veloci (3 fino a 100kHz)**
- **125kB di memoria utente (10k a ritenzione)**

OPZIONI DEI PLC COMPATTI (es. S7-1200)

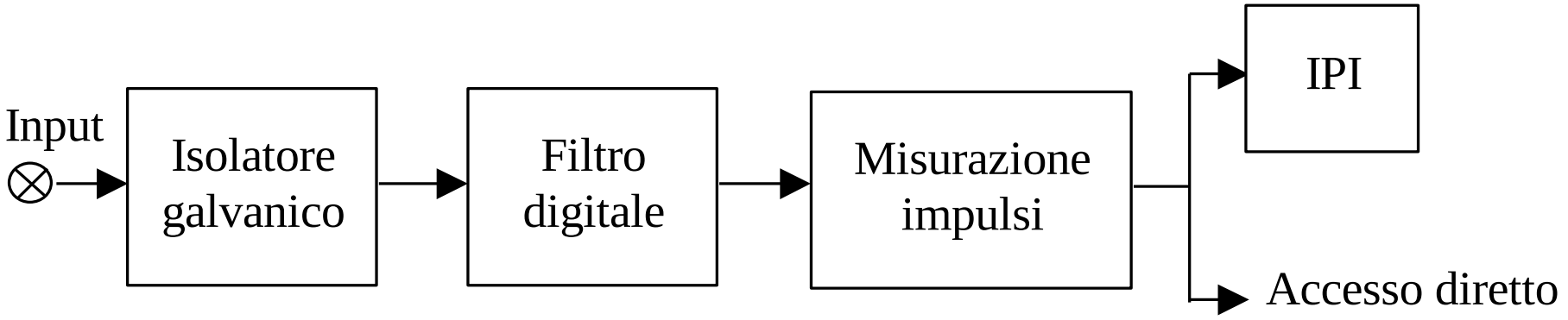
- **I modelli dei PLC compatti differiscono tra loro per numero di I/O e relative funzioni (es. counter o I/O analogico) e numero e tipologia di opzioni (es. S7-1215C differisce dalla S7-1214C solo avere due interfacce Profinet)**
  - **Supporto Signal Module (0/2/8) che sono ad esempio**
    - **8/16 ingressi digitali in gruppi di 2/4**
    - **8/16 uscite digitali (con o senza rele) in gruppi di 1/2**
    - **ingressi e uscite digitali (8+8, 8+8R, 16+16, 16+16R)**
    - **4 ingressi analogici (corrente, tensione, differenziale)**
    - **2 uscite analogiche (corrente, tensione)**
    - **Ingressi e uscite analogiche (4+2)**

  - **Supporto Signal Board (max. 1) che sono plug-in delle CPU per ad esempio:**
    - **estendere I/O digitale (2I+2O)**
    - **generare un'uscita analogica**

  - **Supporto Communication Modules (max 3)**
    - **seriale RS485 per ASCI, Modbus-RTU,…**
    - **Interfaccia Ethernet aggiuntiva**

I/O LOGICO DEI PLC COMPATTI

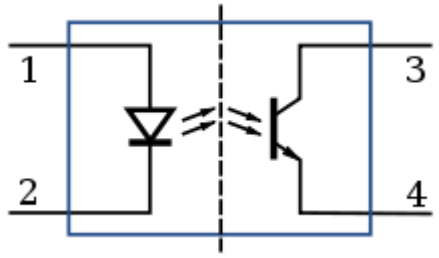
- **Ingressi logici**
  - **Isolati potenza-controllo e isolati canale-canale a gruppi, typ. 24V**
  - **filtrati (~0.1ms < τ < ~10ms) a gruppi**
    - NOTA: es. filtro digitale che campiona a 0.1ms e verifica che siano uguali 2, 4, 8, 16, 32, 64, 128 campioni successivi prima di abilitare un cambio di segnale.
  - **funzioni di appoggio**
    - **misurazione durata impulso**
    - **conteggio veloce**
    - **generazione interrupt**
    - **….**
  - **Accessibili da IPI o da accesso diretto**



I/O LOGICO DEI PLC COMPATTI

- **Uscite logiche**
  - **Isolati potenza-controllo e isolati canale-canale a gruppi**
    - **Transistor (isolatori galvanici): 500V AC potenza-controllo e canale –canale**
    - **Rele': 1500V AC per 1 minuto potenza-controllo e 750V AC canale-canale**
  - **Capacità di supportare correnti da 4A (transistor, correnti di perdita di 10µA) a 10A (rele')**
  - **Capacità di commutare correnti da 0.5A (transistor) a 2-3A (rele')**
  - **Supporto di tensioni in uscita**
    - **Transistor: 0-24VDC, max 5W**
    - **Rele': 0-30VDC, 5-250VAC, max 30WDC e 200WAC (rele')**
  - **Ritardi di commutazione di 100µs (transistor) o 10ms (rele')**
  - **Accessibili da IPU**

NOTA: gli isolatori galvanici, rispetto ai relè, sono direzionali



I/O ANALOGICO DEI PLC COMPATTI

- **Ingressi analogici**
  - **Non isolati, max. 35V, max. 40mA**
  - **Bassa modularità (es. 4)**
  - **Configurabilità dell'ingresso:**
    - **Tensione differenziale ±10V, ±5V, ±2,5V (resistenza d'ingresso ~10MΩ)**
    - **Corrente 0..20mA (resistenza d'ingresso ~10MΩ)**
    - **Possibile compensazione in temperatura**
    - **….**
  - **Tempo di scansione ~ 100μs/canale**
  - **Possibile inserimento di semplici filtri (smoothing)**
  - **Possibile inserimento di filtri notch per 50/60Hz**
  - **Risoluzione ~ 14 bit (es. PLC S7-1214c AW64: 20mA/10V <-> 27648)**
    - **0.2% errore di temperatura (0°C..55°C)**
    - **0.1% accuratezza (linearità, ripetibilità, risoluzione,…)**
    - **…. Le prestazioni sono nell'ordine del 0.1% e per questo è possibile adattare il range d'ingresso)**

I/O ANALOGICO

- **Ingressi e uscite analogiche a 16 bit**
  - **Zero-padding (LSB) in caso di minor risoluzione**
  - **Possibilità di interrupt su overshoot (OB40, limiti definiti da utente)**

Units	Measured value in %	Data word																Range
		2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}	
32767	>118,515	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	Overflow
32511	117,589	0	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	Overshoot range
27649	>100,004	0	1	1	0	1	1	0	0	0	0	0	0	0	0	0	1	
27648	100,000	0	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	Rated range
1	0,003617	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	
0	0,000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
-1	-0,003617	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	
-27648	-100,000	1	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	
-27649	≤-100,004	1	0	0	1	0	0	1	1	1	1	1	1	1	1	1	1	Undershoot range
-32512	-117,593	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	
-32768	≤-117,596	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Underflow

Range ingresso bipolare

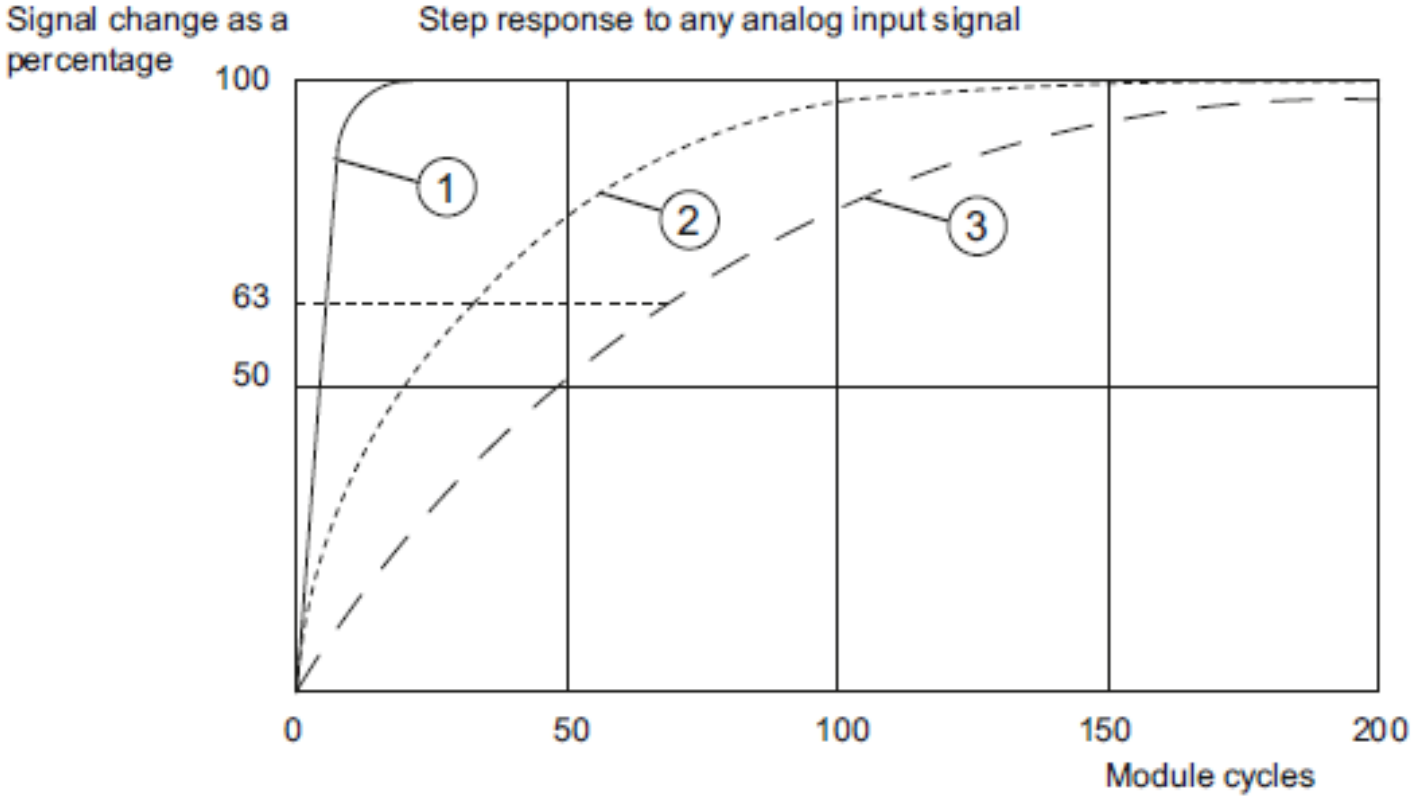
I/O ANALOGICO

❑ Ingressi analogici, range ingresso unipolare

Units	Measured value in %	Data word																Range
		2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}	
32767	≥118,515	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	Overflow
32511	117,589	0	1	1	1	1	1	1	0	1	1	1	1	1	1	1	1	Overshoot
27649	≥100,004	0	1	1	0	1	1	0	0	0	0	0	0	0	0	0	1	range
27648	100,000	0	1	1	0	1	1	0	0	0	0	0	0	0	0	0	0	range
1	0,003617	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	Rated range
0	0,000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
-1	-0,003617	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	Undershoot
-4864	-17,593	1	1	1	0	1	1	0	1	0	0	0	0	0	0	0	0	range range
-32768	≤-17,596	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	Underflow

I/O ANALOGICO

- **Ingressi e uscite analogiche a 16 bit**
  - **Moduli a 2/4/8 canali**
  - **I canali sono scansionati ciclicamente (cycle time/Ncanali ~ 10ms-100ms)**
  - **Possibilità di filtri digitali (smoothing: high, medium, low)**
  - **Esempio modulo a 8 canali a 14 bit**

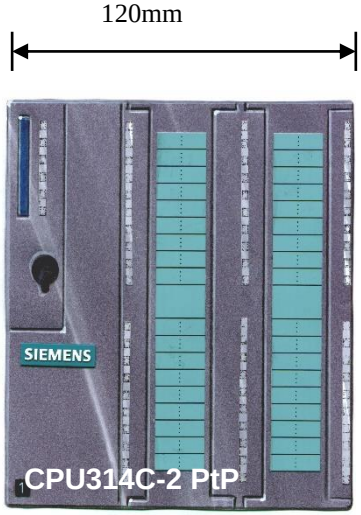


PLC ESPANDIBILI E MODULARI

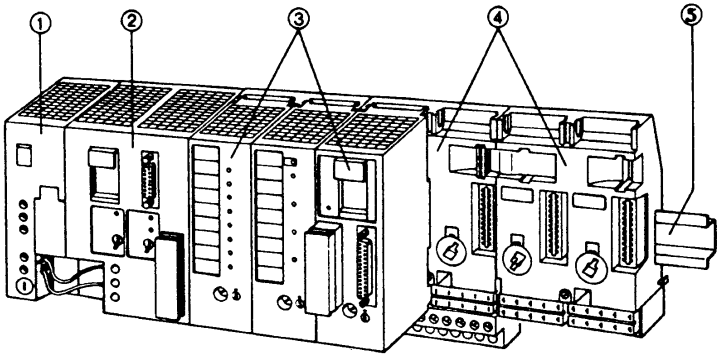
- **PLC vs PMC: aggiungo moduli (HW+SW) invece che schede (+ facile, + rigido)**

- **PLC espandibili e modulari vs PLC compatti:**
  - **HW modulare -> sistemi "ad hoc" (ampia varietà di moduli periferici)**
  - **Integrabilità grazie alla presenza di molti moduli "comunicazione"**
  - **Scalabilità:**
    - **maggiori prestazioni? -> campio CPU**
    - **maggiore interfacciabilità? -> aggiungo I/O**

- **Siemens Simatic S7-300**



Siemens Simatic S5



RACK E MODULI DI BUS

- **BUS = sistema di connessione per le schede che condivide alimentazione e segnali**
- **RACK = telaio a rastrelliera dove vengono inseriti il bus e le schede**
  - **adatto per montaggio in quadro**
  - **ospita più schede (tip. Rack da 19”)**
  - **si possono connettere più rack**
- **MODULI DI BUS = bus in esecuzione a blocchi montato a scatto su guide a Ω**

MODULO ALIMENTATORE

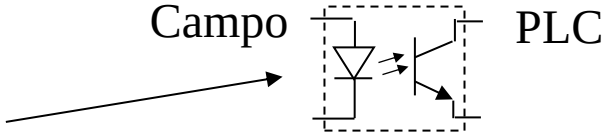
- **Genera le tensioni per tutti i moduli (+5V, ±15V) a partire da $+24V_{DC}$ o $220V_{AC}$**
- **Genera le tensioni per tutti i moduli (+5V, ±15V) a partire da $+24V_{DC}$ o $220V_{AC}$**
  - **intelligenza per la sorveglianza dell’alimentazione e della CPU (watch-dog)**
  - **circuiti di backup e protezione**

MODULI CPU

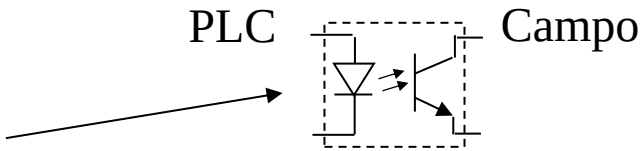
- **Possibilità di scelta tra diverse CPU sulla base di:**
  - **capacità di memoria (programma e/o dati)**
  - **tempo di elaborazione (tempo di istruzioni tipiche)**
  - **spazio di indirizzamento (numero di I/O)**
    - **I/O locali**
    - **I/O remoti**
  - **numero e tipo di interfacce disponibili**
    - **interfaccia diagnostica (punto a punto o multipunto)**
    - **interfacce di rete**
    - **adatto per montaggio in quadro**
  - **interazione con altre CPU (PLC modulari)**

UNITA' PERIFERICHE "NON INTELLIGENTI"

- **Moduli di ingresso logico**
  - **Elevato parallelismo (8/16/32/64) per modulo**
  - **Possibilità di relais o di separazione galvanica**
  - **Presenza di filtro (costante di tempo 0.1ms-10ms)**



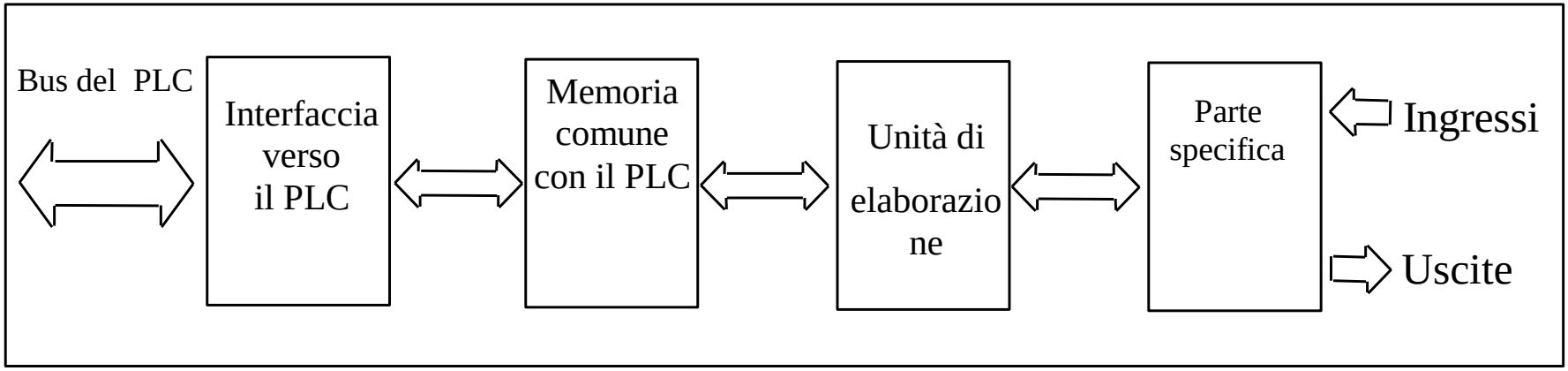
- **Moduli di uscita logica**
  - **Elevato parallelismo (8/16/32/64) per modulo**
  - **Possibilità di relais o di separazione galvanica**
  - **Lo stadio di uscita determina la f_{max} (relais -> 100Hz, transistor -> MHz)**
  - **Lo stadio di uscita determina la I_{max} (relais -> 10A, transistor -> 1A)**



- **Moduli di ingresso/uscita analogico**
  - **Basso parallelismo (2/4/8) per modulo**
  - **Diversi stadi di ingresso (0..+10V, -10V..+10V, 0..20mA, 4..20mA, R, C, ...)**
  - **Risoluzione (8-16bit) e tempo di conversione (100μs-10ms)**

UNITA' PERIFERICHE INTELLIGENTI

- **Unità periferiche intelligenti = moduli funzionali**
  - **Assolvono autonomamente funzioni tipicamente di tipo continuo (difficili da implementare secondo un'architettura sequenziale)**
  - **Utilizzati per controllo assi, interfaccia encoder, comunicazione,…**
  - **Costituite da una parte di gestione I/O e una parte di dialogo con la CPU**



UNITA' PERIFERICHE INTELLIGENTI: TIPOLOGIE

- **Moduli per ingressi veloci**
  - **Es. Moduli di interfaccia encoder (segnali 1Hz-1MHz, stima di p, v, a)**

- **Moduli verso motori**
  - **Es. Moduli per motori passo-passo (un impulso = un angolo α di rotaz. del motore)**

- **Moduli di posizionamento assi (APM –Axis Positioning Module-)**
  - **Moduli di controllo asse (tipo CNC)**

    Encoder + regolatore PID + uscita analogica verso l'azionamento

    - Es. muovi l'asse fino a P con velocità V (profilo a S) e accelerazione A
  - **Moduli di interpolazione assi (tipo CNC)**

    - Gestione integrata di più controlli asse (da 2 a 8)

    - Possibilità di traiettorie su due o tre dimensioni (moduli CNC)

- **Moduli controllore**
  - **Controllori PID (Out = Kp*Err + Ki$\int$Err + Kd*Err')**

    Tempo di ciclo da 100μs

    E' possibile implementare un PID software, ma con tempi di ciclo di $\approx$0.1s

MODULI COMUNICAZIONE

- **Interfaccia diagnostica verso il sistema (PC) di programmazione**
  - **Tipicamente seriale (RS232 o RS485) o USB o Ethernet**
  - **Integrata nella CPU (attiva nella modalità PROG/STOP e RUN)**
- **Interfaccia verso il campo (gestione distribuita di I/O)**
  - **Bus di campo veloce, semplice, robusto (Es AS-Interface)**
  - **Il PLC è master di tali bus**
- **Interfaccia trasversale (integrazione di controllo con altri PLC)**
  - **Bus di campo potente, veloce, robusto (Es Profibus)**
  - **Il PLC può essere master e/o slave di tali bus**
- **Interfaccia verso l'automazione (integrazione con supervisori e computer)**
  - **LAN potente, compatibile, sicura (Es Ethernet)**
  - **Il PLC è in genere "server"**
- **Oggi tutto tende verso Industrial Ethernet (Real-Time Ethernet+TCP/IP)**

IL VANTAGGIO DEI "MODULI INTELLIGENTI"

- **Interfaccia verso sensori (encoder, termocoppie,…)**
  - **Non devo conoscere il sensore, scrivo in celle di memoria le informazioni del sensore che collego e leggo in celle di memoria le informazioni di misura**

- **Interfaccia verso motori**
  - **Non devo conoscere particolari strategie di regolazione, o conoscere l'elettrotecnica del motore, scrivo in celle di memoria le caratteristiche degli azionamenti che collego e scrivo gli obiettivi di posizione e/o velocità e posso leggere lo stato e la diagnostica in opportune variabili**

- **Interfaccia di comunicazione**
  - **Non devo conoscere quel particolare protocollo di comunicazione (es. la struttura del frame, il baud rate,…), ho a disposizione un'area di memoria in scrittura e un'area di memoria in lettura**