



I.T.I. "Modesto PANETTI" – B A R I



Via Re David, 186
70125 BARI

☎ 0805421332 - 0805425412
**Presidenza - 0805560840-Fax -
0805426432**

Classe 5 Elettronica e Telecomunicazioni sez B.
a.s. 2004/2005

Alunni: **BOZZI PASQUALE - VITALE GIUSEPPE**
Docente: Prof. **ETTORE PANELLA**

RELAZIONE DI SISTEMI SUL PLC SIEMENS 216-2

Che cosa sono:

I PLC (Programmable Logic Controller) sono quei dispositivi strutturalmente simili ai PC ma che diversificano negli organi di Ingresso e di uscita. Questi accettano come segnali d'ingresso valori digitali del tipo ON-OFF o segnali analogici di tensione-corrente variabili e restituiscono all'uscita segnali uguali a quelli d'ingresso. I segnali d'ingresso vengono opportunamente elaborati dal software nella CPU che a sua volta comanda le uscite secondo la logica del software. Il software di programmazione risiede nella memoria EEPROM dove viene prelevato dalla CPU ed elaborato.

I PLC sono provvisti di una porta di comunicazione che permette di trasferire sul PC, tutti gli stati delle variabili, degli ingressi e delle uscite, dei contatori, dei temporizzatori, ecc.

La programmazione dei PLC varia, anche se di poco, tra marca e marca, pertanto per ogni tipo di PLC è necessario avere il suo programma, la sua connessione per il trasferimento dei dati (chiave fisica), il suo manuale.

SVILUPPO STORICO:

I controllori logici programmabili furono introdotti, negli Stati Uniti, verso la fine degli anni '60 allo scopo di ridurre i sempre più elevati costi e complessità dei controlli. La realizzazione dei controlli dei nuovi apparati introdotti sul mercato, infatti, avrebbe avuto bisogno di circuiti con centinaia o migliaia di relè. Nella metà degli anni '70 la tecnologia dominante dei PLC era basata su dispositivi sequenziali. La capacità di far comunicare tra loro i PLC iniziò intorno al 1973 con la realizzazione del Modbus relativo al Modicon. Sfortunatamente la mancanza di standardizzazione produsse una serie di protocolli e reti incompatibili così da rendere la comunicazione tra PLC di costruttori diversi quasi impossibili. Negli anni '80 la General Motors tentò di standardizzare la comunicazione con un protocollo denominato MAP (Manufacturing Automation Protocol) ed allo stesso tempo rese programmabili i PLC attraverso personal computer, invece che con tastiere dedicate o programmazioni manuali. Negli anni '90 si è ottenuta una notevole riduzione degli ingombri dei PLC ed una graduale unificazione dei protocolli. Attualmente i PLC possono essere programmati in linguaggio C, schemi a scala, diagrammi a blocchi funzionali, liste di istruzione ed altro ancora.

INTRODUZIONE:

I controlli logici programmabili, indicati come PLC (Programmable Logic Controller) sono apparecchi elettronici basati sul microprocessore e furono inventati per sostituire i circuiti sequenziali realizzati a relè.

Si può affermare che il PLC lavora portando ON oppure OFF le sue uscite sulla base dello stato degli ingressi e di un programma, realizzato appositamente dall'utilizzatore, per ottenere i risultati desiderati.

Le norme CEI 65-23 danno la seguente definizione:

Il controllore a logica programmabile o controllore programmabile, è un sistema elettronico a funzionamento digitale, destinato all'uso in ambito industriale. Sia il CP sia le periferiche associate sono stati progettati in modo da poter essere facilmente integrati in un sistema di controllo industriale e utilizzati in tutte le funzioni previste.

Numerose sono le norme CEI che riguardano i controlli programmabili. Un elenco essenziale è il seguente:

- CEI 3-35. Riguarda la preparazione dei diagrammi funzionali per sistemi di comando e controllo.
- CEI 44-5. Si occupa della sicurezza del macchinario.
- CEI 65-23. Si occupa delle informazioni generali circa i PLC.
- CEI 65-39. Descrive le prove tecniche e meccaniche.
- CEI 65-40. Riguarda i diversi linguaggi di programmazione dei PLC.
- CEI 110-13. Si riferisce alla compatibilità elettromagnetica (norma generica sull'emissione).
- CEI 110-25. Riguarda la compatibilità elettromagnetica (norma tecnica sull'immunità).

I PLC sono impiegati in moltissime applicazioni industriali ove occorrono controlli elettronici dei tipi più disparati quali:

- macchine utensili ed operatrici automatiche;
- macchine da stampa;
- macchine per la lavorazione del legno;
- impianti di dosaggio e miscelazione;
- controlli di processo nell'industria chimica, siderurgica, metallurgica;
- impianti di trattamento delle acque;
- sistemi di immagazzinamento automatico.

Alcune delle qualità che li fanno preferire sono le seguenti:

- funzionamento a bassa tensione e bassi valori di correnti.
- risultano leggeri, poco ingombranti, e possono essere sistemati facilmente in qualsiasi posizione.
- elevata flessibilità.
- possibilità di interfacciamento con video

Per questi motivi spesso sono una valida alternativa, anche in termini di costi e ingombri.

Il PLC consiste principalmente una CPU. Può essere immaginata come una scatola contenente migliaia di relè, insieme a contatori, temporizzatori e memorie per dati. In realtà tutti questi oggetti non esistono fisicamente all'interno della scatola, dato che sono simulati dal microprocessore.

Circuiti o moduli d'ingresso: servono al PLC per ricevere segnali dal mondo esterno.

Circuiti o moduli d'uscita: vengono impiegati dal PLC per inviare i comandi in uscita.

CPU: è l'unità centrale contenente il microprocessore, le memorie RAM ed EPROM, i circuiti di clock e quelli necessari al loro funzionamento.

Dispositivo di programmazione ed interfaccia con l'operazione: sono gli elementi necessari per programmare il PLC e per controllarne lo stato .

Inoltre, si può immaginare che all'interno della CPU siano presenti i seguenti elementi.

Circuiti o relè interni: essi non esistono fisicamente ma si può immaginarli simulati dalla CPU.

Contatori: possono essere sia simulati dal software che realizzati dall'hardware.

Temporizzatori: sono simulati dal software e realizzano principalmente la funzione di flip-flop.

Memoria dei dati: è costituita da registri che sono impiegati per la temporanea memorizzazione di dati durante l'esercitazione di un programma.

Memoria di programma: è la memoria che contiene il programma realizzato dall'operatore.

Un PLC lavora scorrendo, continuamente e ciclicamente, un programma.

Lettura dello stato di ingresso: durante questa fase il PLC controlla in che stato, ON oppure OFF, si trova ciascuno dei suoi ingressi.

A questi ingressi sono collegati i vari sensori, finecorsa, interruttori, pulsanti, che servono per conoscere in quali condizioni si trova la macchina alla quale il PLC è collegato e per inviare comandi.

Il PLC registra questi dati nella sua memoria per usarli durante il passo successivamente.

Esecuzione del programma: il PLC esegue il programma un'istruzione alla volta, in base alle informazioni ricevute dai suoi ingressi, prende delle decisioni per far eseguire alla macchina controllata la sequenza opportuna.

Aggiornamento dello stato delle uscite: le uscite servono a comandare la macchina che il PLC deve controllare.

Lo stato ON oppure OFF di ciascuna uscita viene cambiato oppure no, nella terza fase in base alle letture degli ingressi.

Eseguito il terzo passo, il PLC ritorna al primo passo per ricominciare la sequenza e ripeterla continuamente.

I tecnici che lavorano con i PLC usano comunemente alcuni termini particolari .

Di seguito è data una breve spiegazione di quelli più impiegati.

Sensore: un sensore è un congegno che converte una grandezza fisica in un segnale elettrico.

I sensori sono connessi agli ingressi del PLC.

Esistono sensori di fine corsa, di posizione, di temperature ecc.

Un semplice esempio di sensore è costituito da un pulsante.

Attuatore: un attuatore svolge la funzione inversa del sensore: converte un segnale elettrico in una grandezza fisica.

Ingresso digitale: è anche chiamato ingresso discreto.

Accetta solo segnali di tipo ON/OFF.

Esempi di segnali discreti che possono essere connessi a questi ingressi sono, oltre a quelli provenienti dalle porte logiche, quelli prodotti da pulsanti, interruttori, finecorsa, contatti meccanici.

Uscita analogica: l'uscita analogica è quella che può emettere segnali di valore variabile con continuità da un minimo ad un massimo.

Un esempio è costituito da una tensione continua che può variare tra 0 e 10 V.

CPU: questo acronimo è formato con le iniziali delle parole inglesi *Central Processing Unit*.

È l'unità centrale del sistema, che contiene il microprocessore e la memoria.

Gli elementi che caratterizzano un PLC sono essenzialmente due.

Il primo è costituito dalla velocità di esecuzione delle operazioni e dal loro tipo e numero.

Il secondo è dato dalla possibilità di interfacciamento verso l'esterno con segnali di tipo digitali o analogico.

La vera e propria interfaccia fisica tra il PLC e la macchina, avviene attraverso schede o moduli di ingresso/uscita che acquisiscono o inviano segnali digitali o analogici. Tutti i costruttori dei PLC ne offrono una grande varietà per adattarsi ai diversi tipi di sensori ed ai diversi comandi da erogare.

Si hanno PLC di:

- gamma bassa, quando controllano fino a 64 I/O
- gamma media, quando controllano tra 64 e 512 I/O
- gamma alta, quando controllano più di 512 I/O.

Si dicono compatti o monoblocco i PLC che si presentano in una configurazione rigida che non può essere quasi mai modificata.

Si dicono invece modulari, quelli che sono configurabili a piacere dall'utente in base alle sue esigenze.

L'interfaccia con l'operatore si utilizza in fase di programmazione e per il controllo dei dati nel corso del processo. I linguaggi di programmazione dei PLC sono normalizzati secondo uno standard mondiale il quale definisce come strutturare il programma e quattro diversi linguaggi che sono:

- a) lista di istruzioni (Instruction list = IL);
- b) diagrammi a scala o a contatti (Ladder Diagram = LD);
- c) diagramma a blocchi funzionali (Function Block Diagram = FBD);
- d) linguaggi strutturati, tipo il C (Structured Text = ST)

Lista di istruzioni: è un linguaggio assembly ispirato a quello dei microprocessori

Diagrammi ladder: sono nati per facilitare gli operatori che utilizzavano abitualmente gli schemi elettromeccanici a relè.

IL LINGUAGGIO KOP:

Il linguaggio a contatti (KOP) è un linguaggio grafico che presenta delle analogie con gli schemi elettrici. Quando si scrive un programma in KOP, si utilizzano componenti grafici e li si organizza in modo da creare segmenti logici. Il programma viene creato utilizzando i seguenti tipi di elementi:

- **Contatti.** Si tratta di interruttori che vengono attraversati dal flusso di corrente. Generalmente la corrente passa attraverso un contatto aperto solo quando è chiuso (valore logico pari a uno) e passa attraverso un contatto chiuso o negato (NOT) solo quando è aperto (valore logico pari a zero).
- **Bobine.** Sono relé che vengono eccitati dal flusso della corrente.
- **Box.** Si tratta di una funzione (ad esempio, un temporizzatore, un contatore o un'operazione matematica) che viene eseguita quando il flusso di corrente raggiunge il box.

Un segmento è composto da questi elementi e costituisce un circuito completo. La corrente scorre partendo dalla barra di alimentazione sinistra (rappresentata nell'editor KOP da una linea verticale sul lato sinistro della finestra) attraverso i contatti chiusi per andare ad eccitare le bobine o i box.

COME COLLEGARE IL PLC AL COMPUTER:

Il tipo di comunicazione tra il PC in cui si sta eseguendo STEP 7-Micro/WIN 32 e la CPU dipende dall'hardware installato. Se si decide di collegare il PC alla CPU semplicemente con un cavo PC/PPI, non si deve far altro che collegare il cavo e confermare i parametri di default assegnati al PC e alla CPU durante l'installazione di STEP 7-Micro/WIN 32.

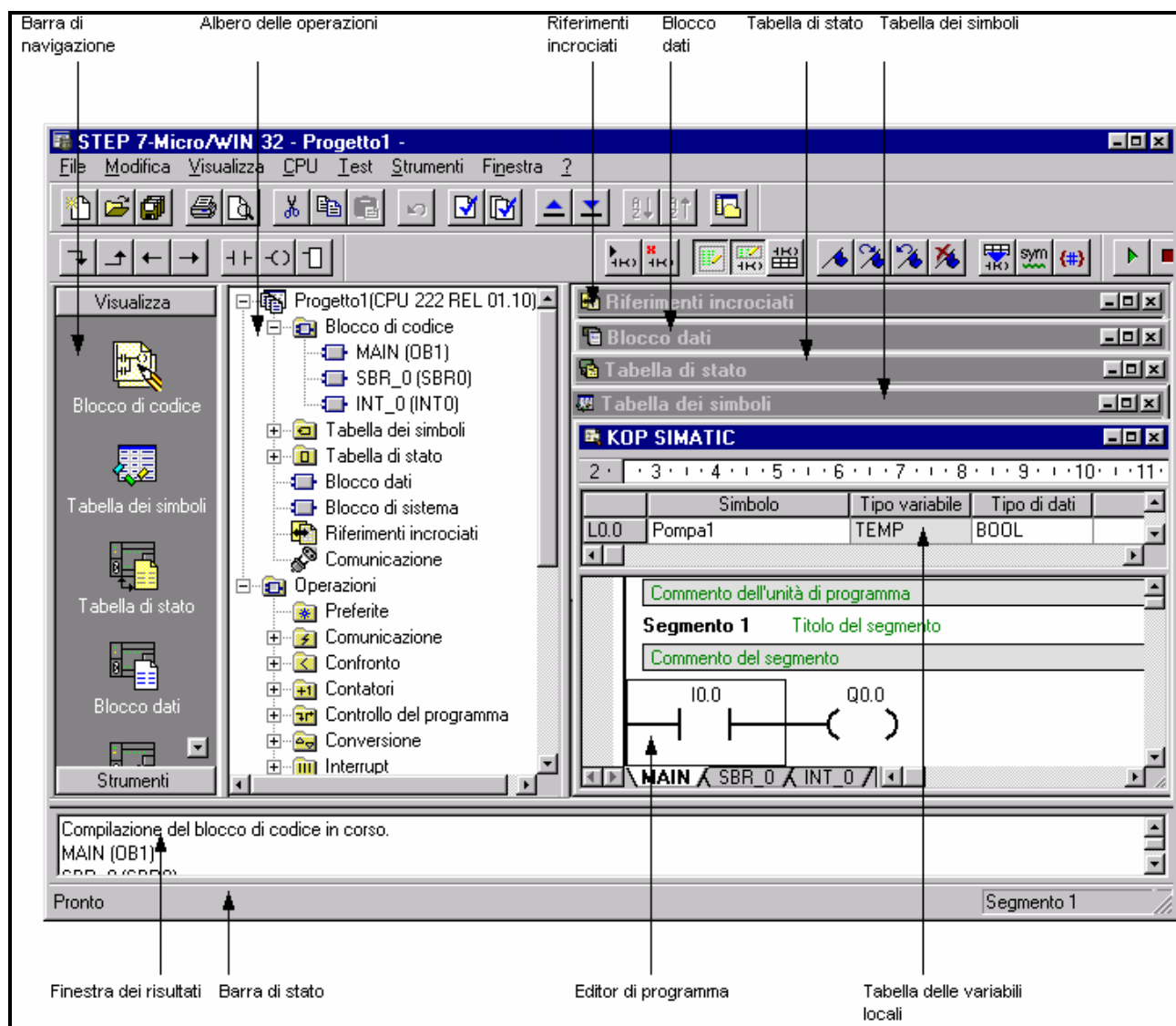
È possibile stabilire la comunicazione e modificare le relative impostazioni in qualsiasi momento.

Di seguito sono elencate le operazioni necessarie per stabilire la comunicazione:

- Connettere un cavo tra la CPU e il PC sul quale è installato STEP 7-Micro/WIN 32. Per connessioni PC/PPI semplici, impostare i microinterruttori su 9600 baud, DCE, 11 bit. Se si utilizza un modem o una scheda di comunicazione, consultare le istruzioni di installazione fornite con l'hardware. Accertarsi che il tipo di CPU selezionato in STEP 7-Micro/WIN 32 corrisponda a quello che si sta utilizzando.

- Se si utilizza un collegamento PC/PPI semplice, si può confermare il protocollo di comunicazione visualizzato per default nella finestra di dialogo "Impostazione interfaccia PG/PC" durante l'installazione di STEP 7-Micro/WIN 32. In alternativa selezionare un protocollo di comunicazione diverso e controllare i parametri (indirizzo della stazione, velocità di trasmissione ecc.) del PC nella finestra di dialogo "Impostazione interfaccia PG/PC". Controllare la configurazione (indirizzo della stazione, velocità di trasmissione ecc.) della CPU nella scheda Porte di Blocco di sistema. Se necessario, apportare delle modifiche e caricare il blocco di sistema modificato.

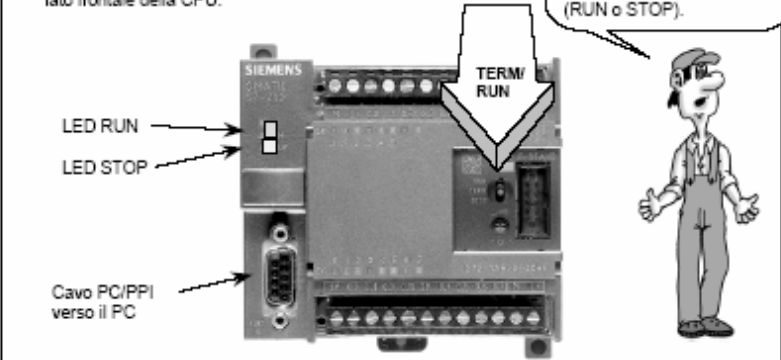
Nella figura seguente viene riportata la schermata principale del programma STEP 7-Micro/WIN 32.



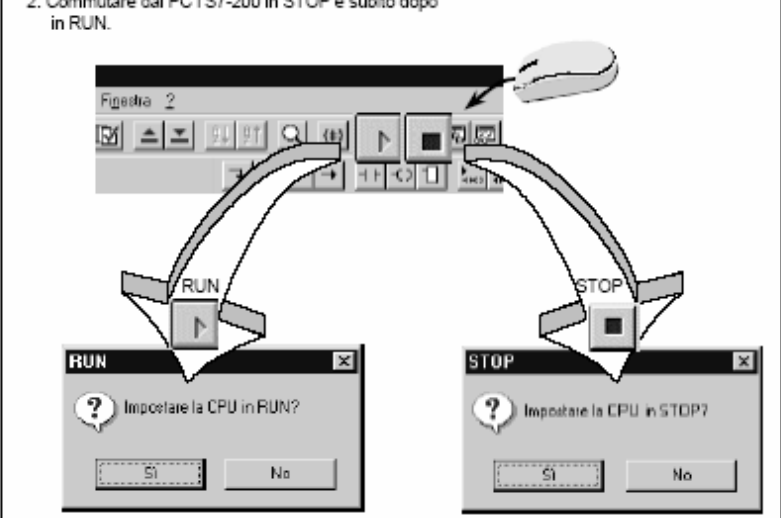
Primo test

1. Posizionare il selettore dei modi operativi del PLC su Term o RUN. Il selettore è nascosto da uno sportellino sul lato frontale della CPU.

Solo in posizione TERM o RUN si può effettuare dal PC/PG l'impostazione remota dello stato operativo (RUN o STOP).



2. Commutare dal PC l'S7-200 in STOP e subito dopo in RUN.



Sull'S7-200 nello stato operativo RUN si illumina il LED "RUN", mentre nello stato operativo STOP si illumina il LED "STOP".

Se la CPU non commuta il suo stato è indispensabile verificare se i cavi sono stati collegati correttamente, se è corretta la velocità di trasmissione ed infine, nel menu *Visualizza > Comunicazioni...* se è stata scelta l'esatta interfaccia COM.

INDIRIZZAMENTO DIRETTO DELLE AREE DI MEMORIA DELLA CPU

La CPU 216-2 memorizza informazioni in diverse posizioni di memoria che hanno indirizzi unici. La memoria dati di S7-200 consiste di cinque aree:

- I Ingresso
- Q Uscita
- M Merker interno
- SM Merker speciale
- S Memoria S
- V Memoria variabile

Per utilizzare un indirizzo di memoria si compone l'indirizzo utilizzando il tipo e il numero di memoria. Si può accedere alle aree di memoria in formato bit, byte, parola o doppia parola.

Esempio di programmazione ladder

Contatti standard (operazioni logiche combinatorie a bit)

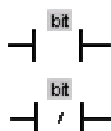
Ingressi/Uscite	Operandi	Tipi di dati
Bit	I, Q, M, SM, T, C, V, S, L	BOOL

Queste operazioni ricavano il valore indirizzato dalla memoria o dal registro dell'immagine di processo quando il tipo di dati è I o Q.

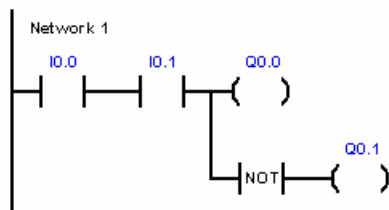
Il **Contatto normalmente aperto** (LD, A, O) è chiuso (on) quando il bit vale 1.

Il **Contatto normalmente chiuso** (LDN, AN, ON) è chiuso (on) quando il bit vale 0.

In KOP le operazioni normalmente aperte e normalmente chiuse sono rappresentate mediante contatti rappresentati in figura



Esempio di programma



NETWORK 1: I contatti n. a. IO.0 AND IO.1 devono essere on (chiusi) per attivare Q0.0. Il contatto Not agisce come un invertitore. In modo RUN, Q0.0 e Q0.1 hanno stati logici opposti.

PILOTAGGIO DI MOTORI TRAMITE IL PLC

Attraverso l'utilizzo del PLC abbiamo pilotato tre diversi tipi di motori: **motore in corrente alternata**, **motore passo-passo** e **motore in corrente continua**.

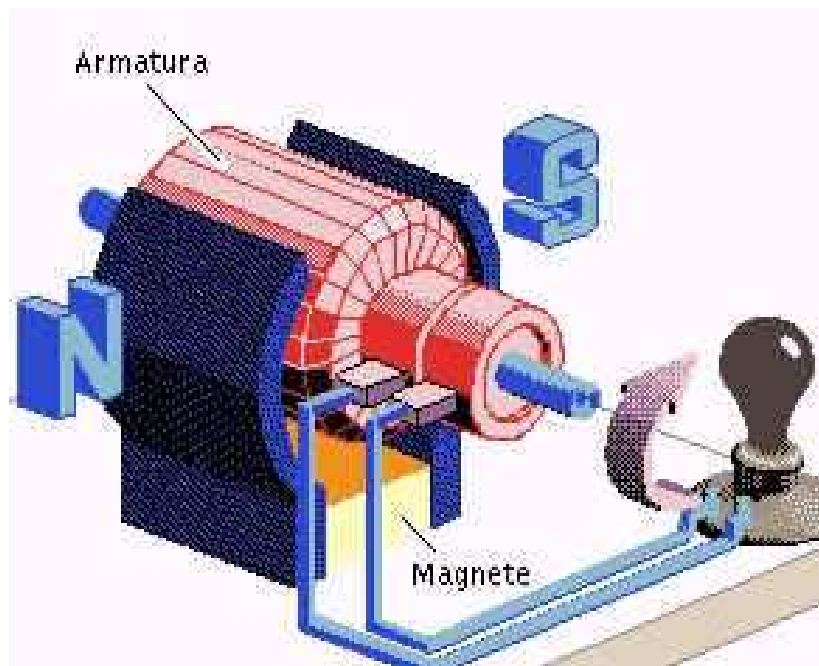
Di seguito si riportano le informazioni sui tre motori, con le rispettive schermate del programma STEP 7 MICRO/WIN 32, con cui si è operato.

Motore in corrente alternata

Sono macchine elettriche rotanti, usate per convertire energia elettrica in energia meccanica o viceversa. In particolare, le macchine che trasformano energia elettrica in energia meccanica sono dette motori elettrici, quelle che eseguono l'operazione inversa sono invece chiamate generatori elettrici.

Il funzionamento delle macchine elettriche rotanti si basa su due fenomeni fisici correlati. Il primo è il fenomeno dell'induzione elettromagnetica, scoperto nel 1831 dal fisico britannico Michael Faraday: se un conduttore si sposta in un campo magnetico, o più precisamente, se varia il flusso concatenato con il conduttore, viene indotta una corrente elettrica.

Il secondo fenomeno fu invece osservato per la prima volta nel 1820 dal fisico francese André-Marie Ampère: un conduttore percorso da corrente immerso in un campo magnetico è sottoposto a una forza che dipende dalla geometria del sistema. Si mostra una semplice schematizzazione di una macchina elettrica rotante.



I motori in corrente alternata maggiormente utilizzati nel campo dell'automazione industriale di bassa potenza sono:

- 1) Motore monofase. È il classico motore in c.a. utilizzato in molti elettrodomestici come lavatrici, lucidatrici, etc.
- 2) Motore universale. È un motore di piccola potenza impiegati in alcuni elettrodomestici come frullatori e ventilatori.
- 3) Motore bifase. È un motore utilizzato nei servomeccanismi di posizione e velocità ad elevata precisione con comando in corrente alternata. È normalmente impiegato per potenze fino ad alcune centinaia di watt.

Nei sistemi di elevata potenza si impiegano i motori trifase in grado di operare su carichi anche di diverse migliaia di KW.

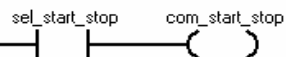
Il motore bifase consente la selezione del verso di rotazione. Tale motore dispone di tre contatti denominati 1-2-3. Se la tensione di rete a 220 V alimenta i contatti 1-2 il motore si muove in un verso se l'alimentazione giunge sui contatti 1-3 il motore ruota nel verso opposto. In tal modo è molto semplice controllare il verso di rotazione del motore. Questo tipo di motore è impiegato nel movimento dei cancelli, delle tapparelle per finestre, ecc.

Si riporta il listato del programma per il comando del motore in corrente alternata bifase con controllo del verso di rotazione.

COMANDO MOTORE AC

Segmento 1

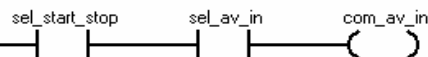
Start motore



Simbolo	Indirizzo	Commento
com_start_stop	Q1.0	Comando motore start/stop (1=start)
sel_start_stop	M0.0	Selettore start/stop (1=start)

Segmento 2

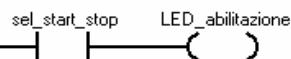
Verso di rotazione del motore



Simbolo	Indirizzo	Commento
com_av_in	Q1.1	Comando motore avanti/indietro (1=avanti)
sel_av_in	M0.1	Selettore avanti indietro (1= avanti)
sel_start_stop	M0.0	Selettore start/stop (1=start)

Segmento 3

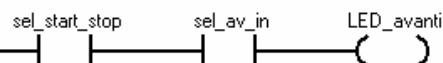
accensione LED abilitazione



Simbolo	Indirizzo	Commento
LED_abilitazione	Q1.2	Accensione led abilitazione
sel_start_stop	M0.0	Selettore start/stop (1=start)

Segmento 4

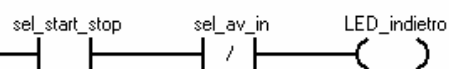
accensione LED avanti



Simbolo	Indirizzo	Commento
LED_avanti	Q1.3	Accensione led avanti
sel_av_in	M0.1	Selettore avanti indietro (1= avanti)
sel_start_stop	M0.0	Selettore start/stop (1=start)

Segmento 5

accensione LED indietro



Simbolo	Indirizzo	Commento
LED_indietro	Q1.4	Accensione led indietro
sel_av_in	M0.1	Selettore avanti indietro (1= avanti)
sel_start_stop	M0.0	Selettore start/stop (1=start)

Segmento 6

MEND

Tabella dei simboli

			Simbolo	Indirizzo	Commento
1			sel_start_stop	M0.0	Selettore start/stop (1=start)
2			sel_av_in	M0.1	Selettore avanti indietro (1= avanti)
3			com_start_stop	Q1.0	Comando motore start/stop (1=start)
4			com_av_in	Q1.1	Comando motore avanti/indietro (1=avanti)
5			LED_abilitazione	Q1.2	Accensione led abilitazione
6			LED_avanti	Q1.3	Accensione led avanti
7			LED_indietro	Q1.4	Accensione led indietro

Motore passo-passo

I motori passo-passo sono motori che, a differenza di tutti gli altri, hanno come scopo quello di mantenere fermo l'albero in una posizione di equilibrio: se alimentati si limitano infatti a bloccarsi in una ben precisa posizione angolare.

Solo indirettamente è possibile ottenerne la rotazione: occorre inviare al motore una serie di impulsi di corrente, secondo un'opportuna sequenza in modo tale da far spostare, per scatti successivi, la posizione di equilibrio.

E' così possibile far ruotare l'albero nella posizione e alla velocità voluta semplicemente contando gli impulsi ed impostando la loro frequenza, visto che le posizioni di equilibrio dell'albero sono determinate meccanicamente con estrema precisione.



I **vantaggi** dei motori passo-passo:

- E' possibile realizzare azionamenti di precisione controllati da computer in catena aperta, cioè senza utilizzare sensori di posizione o di velocità. Sono quindi utilizzabili con relativa semplicità e senza richiedere particolare potenza di calcolo.
- Hanno un'elevata robustezza meccanica ed elettrica: infatti non esistono contatti elettrici striscianti e, se necessario, possono essere realizzati anche in esecuzione completamente stagna.
- E' facile far compiere all'albero piccole rotazioni angolari arbitrarie in ambedue i versi e bloccarlo in una determinata posizione.
- La velocità di rotazione può essere molto bassa anche senza l'uso di riduttori meccanici.



Ovviamente hanno anche **difetti**:

- Richiedono sempre circuiti elettronici per il pilotaggio, in genere di tipo digitale.
- Hanno un funzionamento a scatti e con forti vibrazioni, soprattutto ai bassi regimi e se si adottano le tecniche di pilotaggio più semplici.
- Il loro rendimento energetico è basso e, in genere, la potenza meccanica è piccola.
- Hanno un costo elevato, relativamente ad altri tipi di motore con analoghe prestazioni.
- Difficilmente raggiungono velocità di rotazione elevate.

Si riporta la nota tabella per la sequenza di pilotaggio delle fasi di un motore passo-passo unipolare nella modalità full-step:

CK	A	B	C	D
1	0	1	1	0
2	1	0	1	0
3	1	0	0	1
4	0	1	0	1
1	0	1	1	0

Tabella dei simboli

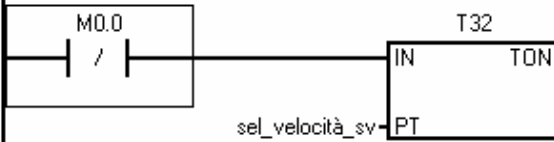
			Simbolo	Indirizzo	Commento
1			fase_D	Q1.3	1 FASE MOTORE PP
2			fase_C	Q1.2	2 FASE MOTORE PP
3			fase_B	Q1.1	3 FASE MOTORE PP
4			fase_A	Q1.0	4 FASE MOTORE PP
5			RES_CONT	M0.1	RESET CTU MPP
6			Sel_av_in	M0.2	Selettore avanti indietro (1=avanti) da supervisione
7			Sel_start_stop	M0.3	Selettore start stop (1=start) da supervisione
8			sel_velocità_sv	VW3	Selettore velocità motore (da 5 a 500 ms) da supervisione
9			num_step_mot	VW0	Numero step motore
10			cont_step_mot	C0	Contatore step motore
11					

Programma

Si riporta il listato del programma per il comando del motore passo-passo

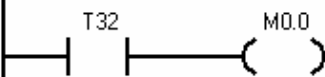
Programma per il controllo della velocità e del verso di rotazione di un motore passo passo

Segmento 1 Timer per velocità motore passo passo (risoluzione 1ms)

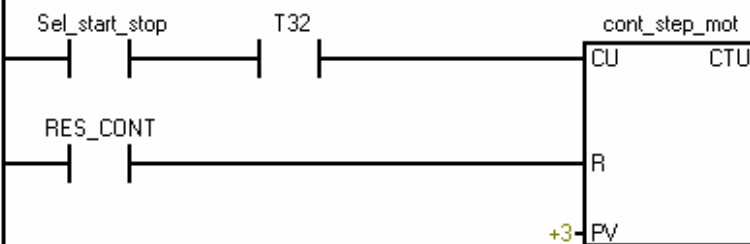


Simbolo	Indirizzo	Commento
sel_velocità_sv	VW3	Selettore velocità motore (da 5 a 500 ms) da supervisione

Segmento 2

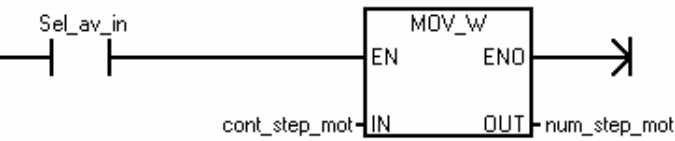


Segmento 3 Conteggio per gli step del motore passo passo



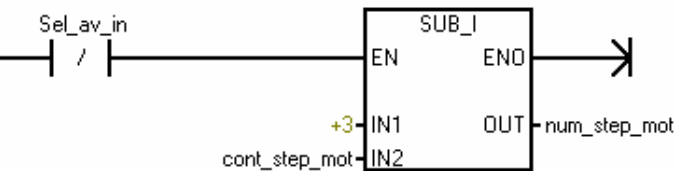
Simbolo	Indirizzo	Commento
cont_step_mot	C0	Contatore step motore
RES_CONT	M0.1	RESET CTU MPP
Sel_start_stop	M0.3	Selettore start stop (1=start) da supervisione

Segmento 4 Move del valore del contatore nella word VW0 che determina gli step del motore nel caso di rotazione in avanti



Simbolo	Indirizzo	Commento
cont_step_mot	C0	Contatore step motore
num_step_mot	VW0	Numero step motore
Sel_av_in	M0.2	Selettore avanti indietro (1=avanti) da supervisione

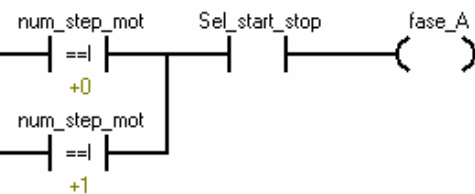
Segmento 5 Calcolo della word VW0 che determina gli step del motore nel caso di rotazione indietro



Simbolo	Indirizzo	Commento
cont_step_mot	C0	Contatore step motore
num_step_mot	VW0	Numero step motore
Sel_av_in	M0.2	Selettore avanti indietro (1=avanti) da supervisione

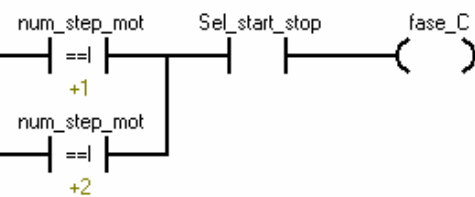
Segmento 6 Set dell'uscita Q0.0 FASE A DEL MOTORE PASSO PASSO

L'uscita viene settata in corrispondenza degli step 0 o 1

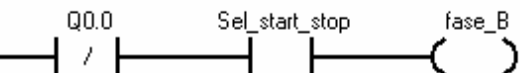


Simbolo	Indirizzo	Commento
fase_A	Q1.0	4 FASE MOTORE PP
num_step_mot	VW0	Numero step motore
Sel_start_stop	M0.3	Selettore start stop (1=start) da supervisione

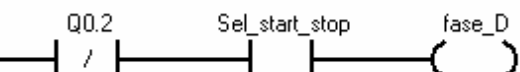
Segmento 7 Set dell'uscita Q0.2 FASE C DEL MOTORE PASSO PASSO



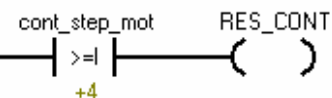
Simbolo	Indirizzo	Commento
fase_C	Q1.2	2 FASE MOTORE PP
num_step_mot	VW0	Numero step motore
Sel_start_stop	M0.3	Selettore start stop (1=start) da supervisione

Segmento 8 Fase B del motore passo passo (=Negato della fase A)

Simbolo	Indirizzo	Commento
fase_B	Q1.1	3 FASE MOTORE PP
Sel_start_stop	M0.3	Selettore start stop (1=start) da supervisione

Segmento 9 Fase D del motore passo passo (=Negato della fase C)

Simbolo	Indirizzo	Commento
fase_D	Q1.3	1 FASE MOTORE PP
Sel_start_stop	M0.3	Selettore start stop (1=start) da supervisione

Segmento 10 Reset del contatore

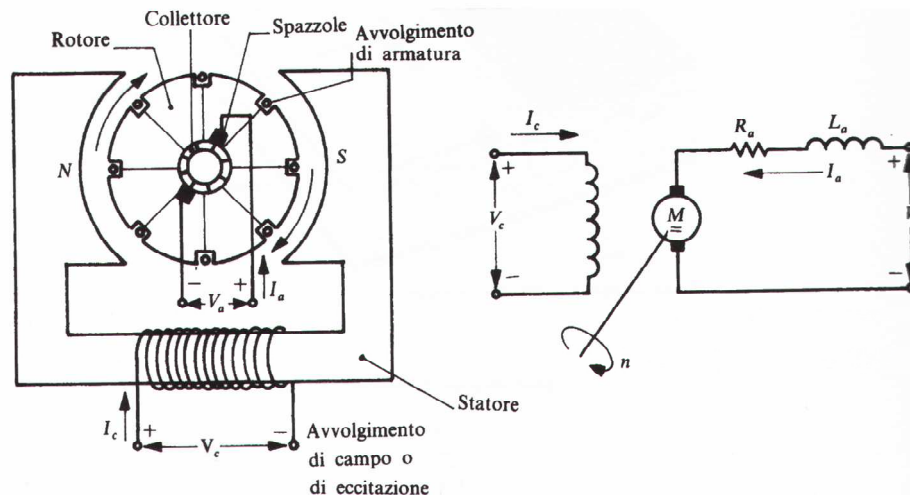
Simbolo	Indirizzo	Commento
cont_step_mot	C0	Contatore step motore
RES_CONT	M0.1	RESET CTU MPP

Segmento 11 Fine programma

MEND

Motore in corrente continua con comando PWM

Nei sistemi di controllo industriale i motori in corrente continua occupano certamente un posto in primo piano. Sono dispositivi in grado di trasformare una tensione continua d'entrata in una coppia motrice in uscita, quindi energia elettrica in meccanica. Nella seguente figura è mostrata la schematizzazione costruttiva di un motore in corrente continua ad eccitazione indipendente.



Un motore in c.c. è costituito da uno *statore* e da un *rotore*. Lo statore genera un flusso magnetico il cui valore dipende dall'intensità di corrente I_c del circuito di eccitazione. Il rotore è posto entro le espansioni polari dello statore. Esso è costituito di materiale ferroso o plastico, e presenta sulla periferia delle cave entro le quali è posto un avvolgimento da cui vengono derivati dei conduttori che sono connessi, in modo opportuno, alle lamelle di un dispositivo denominato *collettore*.

Sul collettore poggiano dei contatti striscianti (*spazzole*) generalmente di carbone attraverso cui si alimenta il motore. Il rotore, le spazzole e il collettore costituiscono il *circuito d'armatura* del motore. Con V_a e I_a si indicano rispettivamente la tensione e la corrente di armatura o alimentazione del motore.

Lo statore insieme all'avvolgimento di eccitazione, costituisce il circuito di campo cui applicare la tensione V_c e la corrente I_c che definiscono il campo magnetico induttore.

Sono possibili altri due modi di eccitazione.

1. **eccitazione serie**
2. **eccitazione parallela**

Quando si alimenta il motore, la corrente I_a di armatura che circola negli avvolgimenti rotorici interagisce con il campo magnetico di statore generando una forza magnetomotrice che pone in rotazione il rotore.

Tale movimento produce per induzione elettromagnetica una forza contro elettromotrice E che si oppone alla tensione V_a di armatura. Affinché possa circolare la corrente I_a che tiene in movimento il motore, la tensione V_a a regime, deve equilibrare sia la f.c.e.m. E che la caduta di tensione $R_a I_a$ nella resistenza del circuito di armatura.

$$V_a = E + R_a I_a$$

Si osservi che, essendo il motore alimentato in continua, l'effetto dell'induttanza di armatura L_a è nullo a regime. La f.c.e.m. E è legata alla velocità di rotazione dalla seguente relazione:

$$E = Kn\phi$$

dove K è una costante costruttiva del motore, n il numero di giri al minuto e ϕ l'intensità del flusso magnetico induttore. Combinando la formula precedente con la successiva, si ricava:

$$n = \frac{V_a - R_a I_a}{K\phi} \cong \frac{V_a}{K\phi}$$

essendo $R_a I_a$ trascurabile rispetto a V_a . La precedente indica che la velocità di rotazione di un motore c.c. è direttamente proporzionale alla tensione di alimentazione V_a . Per il principio di conservazione dell'energia e supponendo trascurabili le perdite, la potenza elettrica fornita al motore $P_a = V_a I_a$ deve uguagliare quella meccanica di rotazione $P_m = C_m \omega$. In particolare si è indicato che con C_m la coppia motrice e con $\omega = 2\pi n$, la velocità angolare. Si ha quindi:

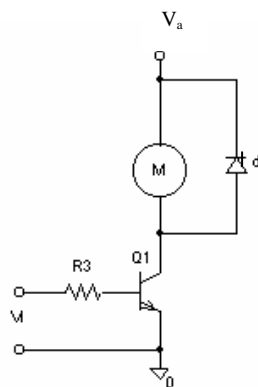
$$C_m \cdot \omega = V_a I_a$$

Ponendo $K_m = 60K/2\pi$, si ricava:

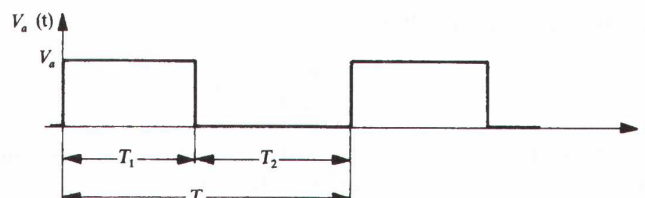
$$C_m = K_m \phi I_a$$

La formula precedente mostra che la coppia motrice C_m è proporzionale al flusso induttore e alla corrente di armatura.

CIRCUITO DI COMANDO DEL MOTORE



Il segnale V_i è costituito da un treno di onde rettangolari a frequenza 10 KHz. Supponendo che il BJT commuti tra la saturazione e l'interdizione, la tensione di alimentazione del motore, trascurando $V_{cesat} = 0.3 \text{ V}$ assume il seguente andamento.





Il motore è equivalente ad un carico induttivo e si comporta come un filtro passa – basso con frequenza di taglio di alcune decine di Hz, imposta dalla costante di tempo meccanica τ_m .

La tensione impulsiva di comando viene, quindi, mediata dall'azione filtrante del motore che, a tutti gli effetti, è come se fosse alimentato da una tensione efficace V pari a:

$$V = \sqrt{\frac{1}{T} \int_0^T V_a^2 \cdot dt} = \sqrt{V_a^2 \cdot \frac{T_1}{T}} = V_a \cdot \sqrt{D}$$

La precedente relazione mostra che è possibile regolare la velocità di rotazione del motore agendo sul duty-cycle D del segnale rettangolare di comando.

Tabella dei simboli

			Simbolo	Indirizzo	Commento
1			Sel_start_stop	M0.0	Start regolazione PWM da supervisione
2			Sel_micro_milli	M0.1	Selettore da supervisore base dei tempi 0= 1micro/ciclo: 1=1ms/ciclo
3			Aggiorna_ciclo_PWM	SM67.0	Aggiornamento del valore del tempo di ciclo di PTO/PWM 1= scrivi il nuovo tempo
4			Aggiorna_dur_imp_PWM	SM67.1	Aggiornamento del valore della durata degli impulsi. 1=scrivi il nuovo tempo
5			Base_tempi_PLS	SM67.3	Base dei tempi di PTO/PWM 1=1 microsec/ciclo 0=1ms/ciclo
6			Selez_modo_PLS	SM67.6	Selezione modo PTO/PWM 0=PTO, 1=PWM
7			Abilita_PLS	SM67.7	Bit di abilitazione PTO/PWM 1=abilita
8			Ciclo_PLS	SMw68	Tempo di ciclo di PTO/PWM (2-65535)
9			PWM_PW	SMw70	Durata degli impulsi dell'uscita PWM (0-65535)
10			Tempo_Ciclo_SV	Vw0	Tempo di ciclo da supervisore (2-65535)
11			Tempo_impulsi_SV	Vw2	Durata impulsi PWM (0-65535)
12					

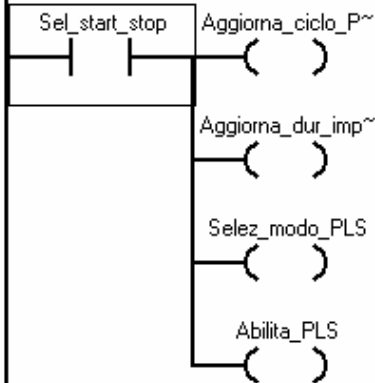
Programma

Si riporta il listato del programma per il comando del motore in corrente continua

COMANDO MOTORE CC REGOLAZIONE PwM (USCITA Q0.0)

Segmento 1

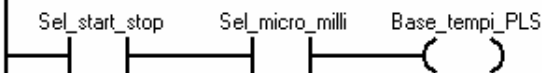
SETTAGGIO PwM



Simbolo	Indirizzo	Commento
Abilita_PLS	SM67.7	Bit di abilitazione PTO/PwM 1=abilita
Aggiorna_ciclo_PwM	SM67.0	Aggiornamento del valore del tempo di ciclo di PTO/PwM 1= scrivi il nuov...
Aggiorna_dur_imp_P...	SM67.1	Aggiornamento del valore della durata degli impulsi. 1=scrivi il nuovo tempo
Sel_start_stop	M0.0	Start regolazione PwM da supervisione
Selez_modulo_PLS	SM67.6	Seleziona modo PTO/PwM 0=PTO, 1=PwM

Segmento 2

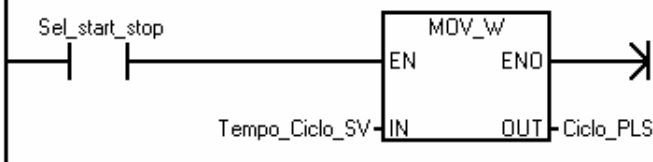
Selezione base dei tempi PwM



Simbolo	Indirizzo	Commento
Base_tempi_PLS	SM67.3	Base dei tempi di PTO/PwM 1=1 microsec/ciclo 0=1ms/ciclo
Sel_micro_milli	M0.1	Selettore da supervisore base dei tempi 0= 1micro/ciclo: 1=1ms/ciclo
Sel_start_stop	M0.0	Start regolazione PwM da supervisione

Segmento 3

Impostazione del tempo di ciclo da supervisore



Confronto delle caratteristiche degli editor: KOP, FUP, AWL (GS 2.5)

Le CPU S7-200 SIMATIC mettono a disposizione svariati tipi di operazioni che consentono di svolgere un'ampia gamma di compiti di automazione. Nella CPU S7-200 sono disponibili due set di operazioni di base:

- SIMATIC
- IEC 1131-3

STEP 7-Micro/WIN 32 consente di scegliere tra diversi editor che permettono di creare programmi di controllo con queste operazioni. L'utente può, ad esempio, scegliere di creare il programma in un ambiente grafico oppure preferire un tipo di editor basato sul linguaggio testuale. Per fare una scelta, occorre considerare i due seguenti aspetti fondamentali della creazione dei programmi.

- Il tipo di set di operazioni migliore per l'applicazione specifica: SIMATIC oppure IEC 1131-3
- Il tipo di editor più adatto alle esigenze di programmazione specifiche: lista istruzioni, schema a contatti o schema logico.

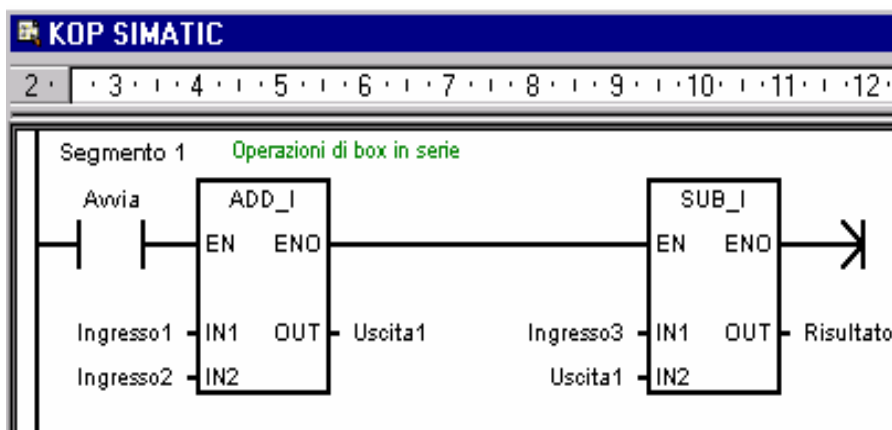
Sono possibili le seguenti combinazioni di set di operazioni ed editor:

- Set di operazioni SIMATIC con l'editor KOP, FUP e AWL
- Set di operazioni IEC 1131-3 con l'editor LD (KOP) o FBD (FUP)

EDITOR KOP

L'editor KOP di STEP 7-Micro/WIN 32 consente di creare programmi simili a circuiti elettrici. La programmazione in KOP è il metodo scelto da molti programmatori di PLC e addetti alla manutenzione ed è un ottimo linguaggio per i programmatori meno esperti. I programmi KOP consentono alla CPU di simulare il flusso della corrente elettrica che proviene da una sorgente e attraversa una serie di condizioni logiche di ingresso, che a loro volta abilitano condizioni logiche di uscita. La logica è suddivisa in "network" o "segmenti". Il programma viene eseguito un segmento per volta, da sinistra a destra e dall'alto verso il basso in base alle indicazioni del programma stesso. Una volta che la CPU ha raggiunto la fine del programma, ricomincia dall'inizio.

La seguente figura illustra un esempio di programma KOP.



Le varie operazioni sono rappresentate mediante simboli grafici e sono di tre tipi fondamentali:

- **I contatti:** rappresentano le condizioni logiche di ingresso in modo analogo a interruttori, pulsanti, condizioni interne, ecc.
- **Le bobine:** solitamente rappresentano i risultati logici di uscita in modo analogo a lampade, avviatori per motori, relè di interposizione, condizioni interne di uscita, ecc.
- **I box:** appresentano le altre operazioni in modo analogo a temporizzatori, contatori o operazioni matematiche.

I segmenti KOP possono essere semplici o molto complessi. È possibile creare segmenti con uscite intermedie e collegare in serie più box. Le operazioni dei box che possono essere collegate in serie vengono etichettate con la linea ENO (enable output). Se un box ha un flusso di corrente in un ingresso EN e viene eseguito senza errori, l'uscita ENO passa il flusso di corrente all'elemento successivo. ENO può essere utilizzata come bit di attivazione che indica la corretta esecuzione di un'operazione. Il bit ENO viene utilizzato con l'inizio dello stack per influire sul flusso di corrente ed eseguire sequenze di operazioni successive.

Avvertenza:

È possibile utilizzare la funzione ENO con il software di programmazione STEP 7-Micro/WIN 32 versione 3.0 (o successiva) solo se si sta usando una CPU 221, CPU 222, CPU 224, CPU 226 o CPU 226XM.

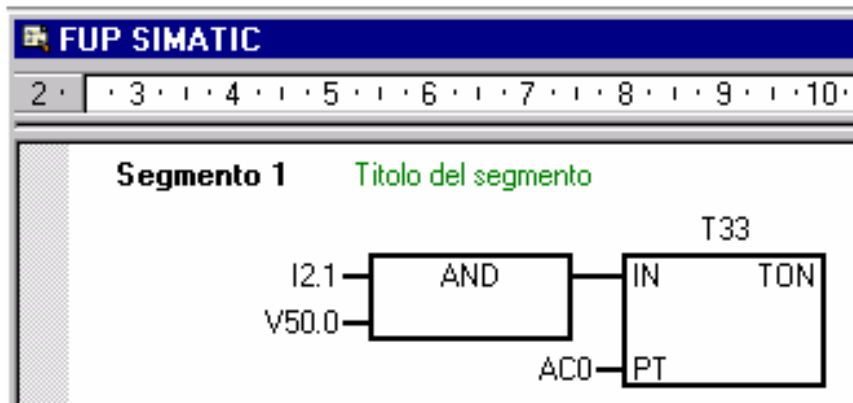
Le caratteristiche principali da considerare nella scelta dell'editor KOP sono le seguenti:

- Lo schema a contatti è facilmente utilizzabile dai programmatori poco esperti.
- La rappresentazione grafica è spesso semplice da interpretare ed è nota in tutto il mondo.
- L'editor KOP può essere utilizzato con i set di operazioni sia SIMATIC, che IEC 1131-3.
- È sempre possibile utilizzare l'editor AWL per visualizzare un programma creato con l'editor KOP.

Editor FUP

L'editor FUP di STEP 7-Micro/WIN 32 consente di visualizzare le operazioni in box logici simili ai comuni schemi a porte logiche. Non ci sono contatti e bobine come nell'editor KOP, ma operazioni equivalenti che vengono rappresentate come operazioni nei box. La logica del programma è derivata dalle connessioni tra le operazioni dei box. Ciò significa che l'uscita di un'operazione (ad es. di un box AND) può essere utilizzata per abilitare un'altra operazione (ad es. un temporizzatore) e creare la necessaria logica di controllo. Questo concetto di connessione consente di risolvere facilmente un'ampia gamma di problemi, proprio come con gli altri editor.

La seguente figura illustra un esempio di un programma creato con l'editor FUP.



Se un box ha un flusso di corrente in un ingresso EN e viene eseguito senza errori, l'uscita ENO passa il flusso di corrente all'elemento successivo. ENO può essere utilizzata come bit di attivazione che indica la corretta esecuzione di un'operazione. Il bit ENO viene utilizzato con l'inizio dello stack per influire sul flusso di corrente ed eseguire sequenze di operazioni successive.

Avvertenza:

È possibile utilizzare la funzione ENO con il software di programmazione STEP 7-Micro/WIN 32 versione 3.0 (o successiva) solo se si sta usando una CPU 221, CPU 222, CPU 224, CPU 226 o CPU 226XM.

Le caratteristiche principali da considerare nella scelta dell'editor FUP sono le seguenti:

- Lo stile di rappresentazione mediante porte logiche è ideale per seguire il flusso del programma.
- L'editor FUP può essere utilizzato sia con il set di operazioni SIMATIC che con il set IEC1131-3.
- È sempre possibile utilizzare l'editor AWL per visualizzare un programma creato con l'editor KOP.
- Grazie ai box AND/OR espandibili è facile disegnare combinazioni complesse di ingressi.

Editor AWL

L'editor AWL di STEP 7-Micro/WIN 32 consente di creare un programma di controllo specificando le abbreviazioni mnemoniche delle operazioni. In genere, l'editor AWL è adatto a programmatori esperti che hanno una buona conoscenza dei PLC e della programmazione. Questo editor consente inoltre di creare programmi che non potrebbero essere creati con gli editor KOP e FUP. Questo è possibile perché si programma nel linguaggio naturale della CPU, invece che con un editor grafico in cui occorre applicare alcune restrizioni per disegnare correttamente i diagrammi. La seguente figura illustra un esempio di un programma AWL.

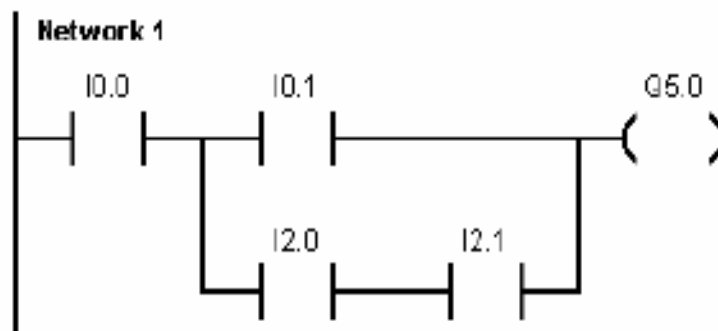
```
SEGMENTO 1
LD    I0.0
LD    I0.1
```

```

LD    I2.0
A     I2.1
OLD
ALD
=     Q5.0

```

Come si può vedere nella figura, questo tipo di linguaggio testuale è molto simile alla programmazione in assembly. La CPU esegue ogni operazione nell'ordine indicato dal programma, dall'inizio alla fine, e quindi riparte dall'inizio. Il linguaggio AWL e il linguaggio assembly presentano anche altre analogie. Le CPU S7-200 utilizzano uno stack logico per risolvere la logica di controllo. Gli editor KOP e FUP inseriscono automaticamente le operazioni necessarie per gestire il funzionamento dello stack. In AWL, l'utente deve inserire tali operazioni per gestire lo stack autonomamente. La seguente figura mostra un semplice programma in KOP e il programma corrispondente in AWL.



```

SEGMENTO 1
LD    I0.0
LD    I0.1
LD    I2.0
A     I2.1
OLD
ALD
=     Q5.0

```

Di seguito si mostra ciò che accade nello stack:

Operazioni

Stack	LD I0.0	LD I0.1	LD I2.0	A	OLD	ALD
S0	I0.0	I0.1	I2.0	I2.0 AND I2.1	(I2.0 AND I2.1) OR I0.1	I0.0 AND [(I2.0 AND I2.1) OR I0.1]
S1		I0.0	I0.1	I0.1	I0.0	
S2			I0.0	I0.0		
S3						
S4						
S5						
S6						
S7						
S8						

Le caratteristiche principali da considerare nella scelta dell'editor AWL sono le seguenti:

- AWL è più adatto ai programmatori esperti.
- AWL consente di risolvere problemi che non possono essere risolti facilmente con gli editor KOP e FUP.
- Con l'editor AWL si può utilizzare solamente il set di operazioni SIMATIC. Non esiste un set di operazioni IEC per AWL.
- Mentre è sempre possibile utilizzare l'editor AWL per visualizzare ed editare un programma creato con gli editor SIMATIC KOP o FUP, il contrario non è sempre vero. Non sempre è possibile utilizzare gli editor KOP o FUP SIMATIC per visualizzare un programma scritto con l'editor AWL.