SIEMENS

Documentazione didattica SCE

Siemens Automation Cooperates with Education | 05/2017

Modulo TIA Portal 032-200

Basi della programmazione di FB
con SIMATIC S7-1500

Cooperates with Education
Automation
SIEMENS

Trainer Package SCE adatti a questa documentazione didattica

Controllori SIMATIC

- **SIMATIC ET 200SP Open Controller CPU 1515SP PC F e HMI RT SW**
  N. di ordinazione: 6ES7677-2FA41-4AB1
- **SIMATIC ET 200SP Distributed Controller CPU 1512SP F-1 PN Safety**
  N. di ordinazione: 6ES7512-1SK00-4AB2
- **SIMATIC CPU 1516F PN/DP Safety**
  N. di ordinazione: 6ES7516-3FN00-4AB2
- **SIMATIC S7 CPU 1516-3 PN/DP**
  N. di ordinazione: 6ES7516-3AN00-4AB3
- **SIMATIC CPU 1512C PN con software e PM 1507**
  N. di ordinazione: 6ES7512-1CK00-4AB1
- **SIMATIC CPU 1512C PN con software, PM 1507 e CP 1542-5 (PROFIBUS)**
  N. di ordinazione: 6ES7512-1CK00-4AB2
- **SIMATIC CPU 1512C PN con software**
  N. di ordinazione: 6ES7512-1CK00-4AB6
- **SIMATIC CPU 1512C PN con software e CP 1542-5 (PROFIBUS)**
  N. di ordinazione: 6ES7512-1CK00-4AB7

SIMATIC STEP 7 Software for Training

- **SIMATIC STEP 7 Professional V14 SP1- licenza singola**
  Nr. di ordinazione: 6ES7822-1AA04-4YA5
- **SIMATIC STEP 7 Professional V14 SP1 - licenza per una classe da 6 postazioni**
  Nr. di ordinazione: 6ES7822-1BA04-4YA5
- **SIMATIC STEP 7 Professional V14 SP1 - licenza upgrade da 6 postazioni**
  Nr. di ordinazione: 6ES7822-1AA04-4YE5
- **SIMATIC STEP 7 Professional V14 SP1 - licenza per studenti da 20 postazioni**
  Nr. di ordinazione: 6ES7822-1AC04-4YA5

Tenere presente che questi Trainer Package potrebbero essere sostituiti da successivi pacchetti.
Potete consultare i pacchetti SCE attualmente disponibili su: siemens.com/sce/tp

Corsi di formazione

Per corsi di formazione regionali di Siemens SCE contattare il partner di contatto SCE regionale
siemens.com/sce/contact

Ulteriori informazioni su SCE

siemens.com/sce

Avvertenze per l'impiego

La documentazione didattica SCE per la soluzione di automazione omogenea Totally Integrated Automation (TIA) è stata creata per il programma "Siemens Automation Cooperates with Education (SCE)" specialmente per scopi di formazione per enti di formazione, di ricerca e di sviluppo pubblici. La Siemens AG declina qualunque responsabilità riguardo ai contenuti di questa documentazione.

Questa documentazione può essere utilizzata solo per la formazione base di prodotti e sistemi Siemens. Ciò significa che può essere copiata in parte, o completamente, e distribuita agli studenti nell'ambito della loro formazione professionale. La riproduzione, distribuzione e divulgazione di questa documentazione è consentita solo all'interno di istituzioni di formazione pubbliche e a scopo di formazione professionale.

Qualsiasi eccezione richiede un'autorizzazione scritta dal partner di riferimento di Siemens AG: Sig. Roland Scheuerer roland.scheuerer@siemens.com.

Le trasgressioni obbligano al risarcimento dei danni. Tutti i diritti sono riservati, incluso anche quelli relativi alla distribuzione e in particolare quelli relativi ai brevetti e ai marchi GM.

L'utilizzo per corsi rivolti a clienti del settore industria è esplicitamente proibito e non è inoltre permesso l’utilizzo commerciale della documentazione.

Ringraziamo la Technische Universität Dresden, e in particolare il Prof. Dr. Ing. Leon Urbas, la Michael Dziallas Engineering e tutte le persone coinvolte nella creazione di questa documentazione didattica.

Sommario

BASI DELLA PROGRAMMAZIONE DI FB

1 Obiettivo

Questo capitolo spiega gli elementi di base di un programma di comando: ***blocchi organizzativi (OB)***, ***funzioni (FC)***, **blocchi funzionali (FB)** e **blocchi dati (DB)**. Inoltre presenta la programmazione di funzioni e blocchi funzionali ***gestibili in biblioteche***. Il lettore imparerà a conoscere il linguaggio di programmazione ***schema logico (FUP)*** e a utilizzarlo per la programmazione di un blocco funzionale FB1 e di un blocco organizzativo OB1.

È possibile utilizzare tutti i controllori SIMATIC S7 riportati nel capitolo 3.

2 Presupposti

Questo capitolo si basa sulla configurazione hardware della CPU1516F-3 PN/DP SIMATIC S7 ma può essere realizzato anche con altre configurazioni hardware dotate di schede di ingresso e di uscita digitali. Per l'esecuzione di questo capitolo è possibile ad es. utilizzare il seguente progetto:

SCE_IT_012_101__Configurazione hardware_CPU1516F.zap13

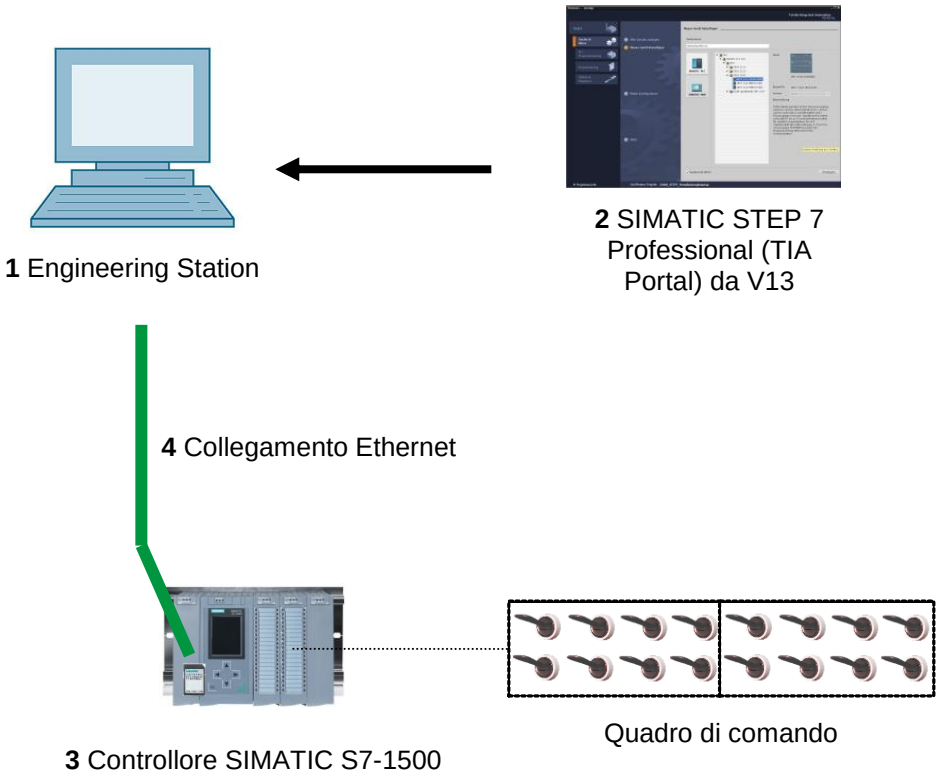
3 Requisiti hardware e software

1 Engineering Station: i requisiti sono hardware e sistema operativo
(per ulteriori informazioni vedere il file Readme/Leggimi sul DVD di installazione di TIA Portal)

2 Software SIMATIC STEP 7 Professional in TIA Portal – da V13

3 Controllore SIMATIC S7-1500/S7-1200/S7-300, ad es. CPU 1516F-3 PN/DP – dal firmware V1.6 con Memory Card e 16DI/16DQ e 2AI/1AQ
Nota: gli ingressi digitali devono essere condotti su un quadro di comando esterno.

4 Collegamento Ethernet tra Engineering Station e controllore



4 Base teorica

4.1 Sistema operativo e programma utente

Il ***sistema operativo*** è presente in ogni controllore (CPU) e organizza tutte le funzioni e i processi della CPU che non sono collegati con un compito di comando specifico. Tra i compiti del sistema operativo figurano ad es.:

- Gestione del nuovo avvio (a caldo)
- Aggiornamento dell'immagine di processo degli ingressi e delle uscite
- Richiamo ciclico del programma utente
- Rilevamento di allarmi e richiamo degli OB di allarme
- Identificazione e trattamento degli errori
- Gestione delle aree di memoria

Il sistema operativo è parte integrante della CPU ed è già in dotazione alla fornitura.

Il ***programma utente*** contiene tutte le funzioni necessarie per l'elaborazione di un compito di automazione specifico. Tra i compiti del programma utente figurano ad es.:

- Verifica dei presupposti necessari per un nuovo avvio (avviamento a caldo) con l’aiuto di OB di avvio
- Elaborazione dei dati di processo, ovvero comando dei segnali di uscita in funzione degli stati dei segnali di ingresso
- Reazione ad allarmi e ingressi di allarme
- Elaborazione di guasti durante la normale esecuzione del programma

4.2 Blocchi organizzativi

I blocchi organizzativi (OB) costituiscono l'interfaccia tra il sistema operativo del controllore (CPU) e il programma utente. Vengono richiamati dal sistema operativo e comandano le seguenti operazioni:

- Elaborazione ciclica del programma (ad es. OB1)
- Comportamento del controllore all'avvio
- Elaborazione del programma comandata da un allarme
- Trattamento degli errori

Un progetto deve contenere almeno ***un blocco organizzativo per l'elaborazione ciclica del programma***. Un OB viene richiamato da un ***evento di avvio***, come mostra la Figura 1. I singoli OB hanno priorità fisse, così, ad es., un OB82 per il trattamento di errori può interrompere l'OB1 ciclico.

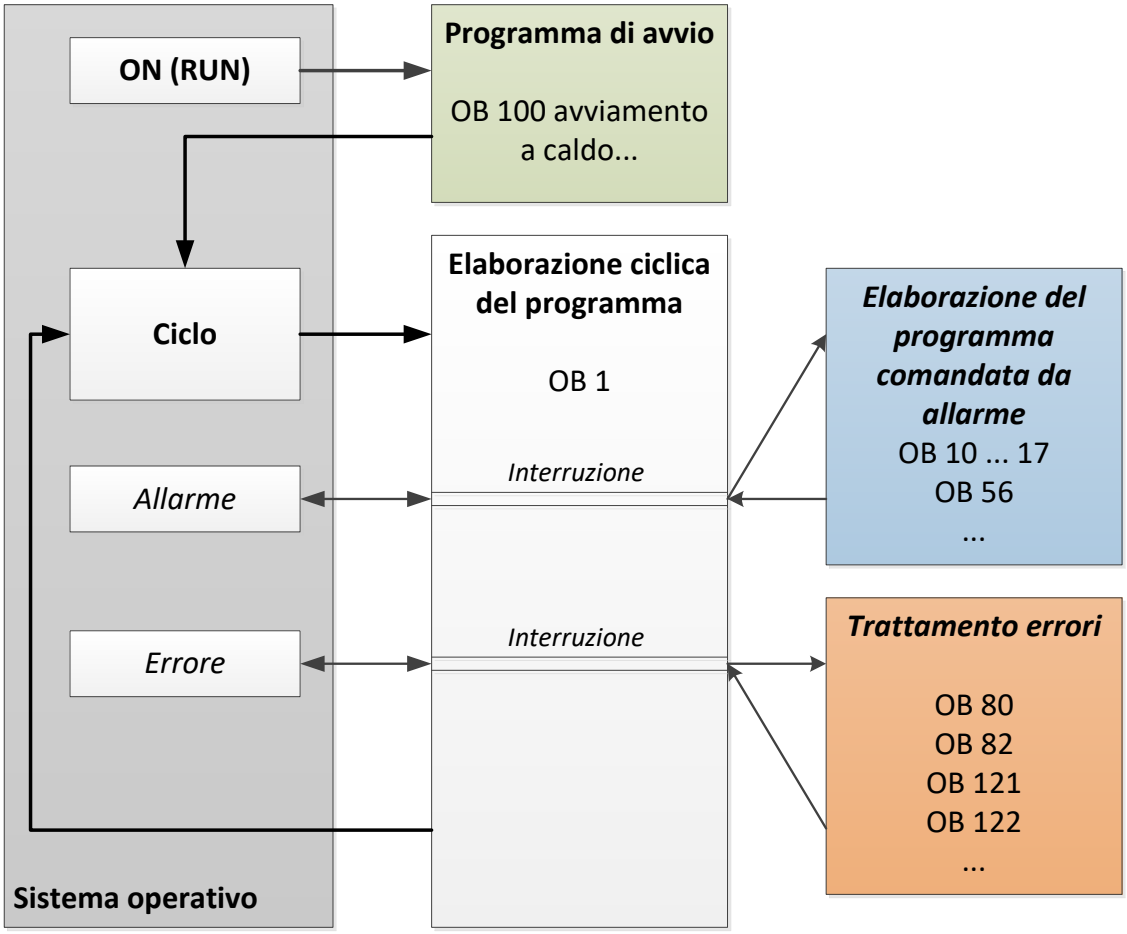


Figura 1: Eventi di avvio nel sistema operativo e richiami degli OB

Dopo che si è verificato un evento di avvio sono possibili le seguenti reazioni:

- Se all'evento è assegnato un OB, l'evento avvia l'esecuzione dell'OB che gli è assegnato. Se la priorità dell'OB assegnato è maggiore di quella dell'OB già in esecuzione, esso viene eseguito immediatamente (Interrupt). In caso contrario si attende prima l'esecuzione dell'OB con la priorità maggiore.
- Se all'evento non è assegnato un OB, si ha la reazione del sistema preimpostata.

La Tabella 1 mostra alcuni esempi di eventi di avvio per un SIMATIC S7-1500, i possibili numeri di OB e la reazione di sistema preimpostata qualora il blocco organizzativo non dovesse essere presente nel controllore.

Evento di avvio	**Possibili numeri di OB**	**Reazione di sistema preimpostata**
Avviamento	100, ≥ 123	Ignora
Programma ciclico	1, ≥ 123	Ignora
Allarme dall'orologio	10 ... 17, ≥ 123	-
Allarme di aggiornamento	56	Ignora
Tempo di controllo del ciclo superato una volta	80	STOP
Allarme di diagnostica	82	Ignora
Errore di programmazione	121	STOP
Errore di accesso alla periferia	122	Ignora

Tabella 1: numeri di OB per diversi eventi di avvio

4.3 Immagine di processo ed elaborazione ciclica del programma

Se nel programma utente ciclico vengono indirizzati gli ingressi (I) e le uscite (Q), normalmente gli stati del segnale non vengono interrogati direttamente dai moduli di ingresso/uscita ma si accede a un'area di memoria della CPU. Questa area di memoria contiene un'immagine degli stati di segnale ed è definita **immagine di processo**.

L'elaborazione ciclica del programma segue l'ordine seguente.

1. All'inizio del programma ciclico viene verificato se i singoli ingressi conducono tensione o meno. Questo stato degli ingressi viene salvato nell'**immagine di processo degli ingressi (PII)**. Per gli ingressi che portano la tensione viene salvata l'informazione 1 o "High", per quelli che non portano la tensione viene salvata l'informazione 0 o "Low".
2. Il processore elabora ora il programma salvato nel blocco organizzativo. Per ottenere l'informazione di ingresso necessaria si accede all'**immagine di processo degli ingressi (PII)** letta in precedenza e i risultati logici combinatori vengono scritti in una cosiddetta **immagine di processo delle uscite (PIQ)**.
3. Alla fine del ciclo l'**immagine di processo delle uscite** (**PIQ**) viene trasferita come stato di segnale ai moduli di uscita e questi ultimi vengono attivati/disattivati. In seguito l'elaborazione riprende dal punto 1.

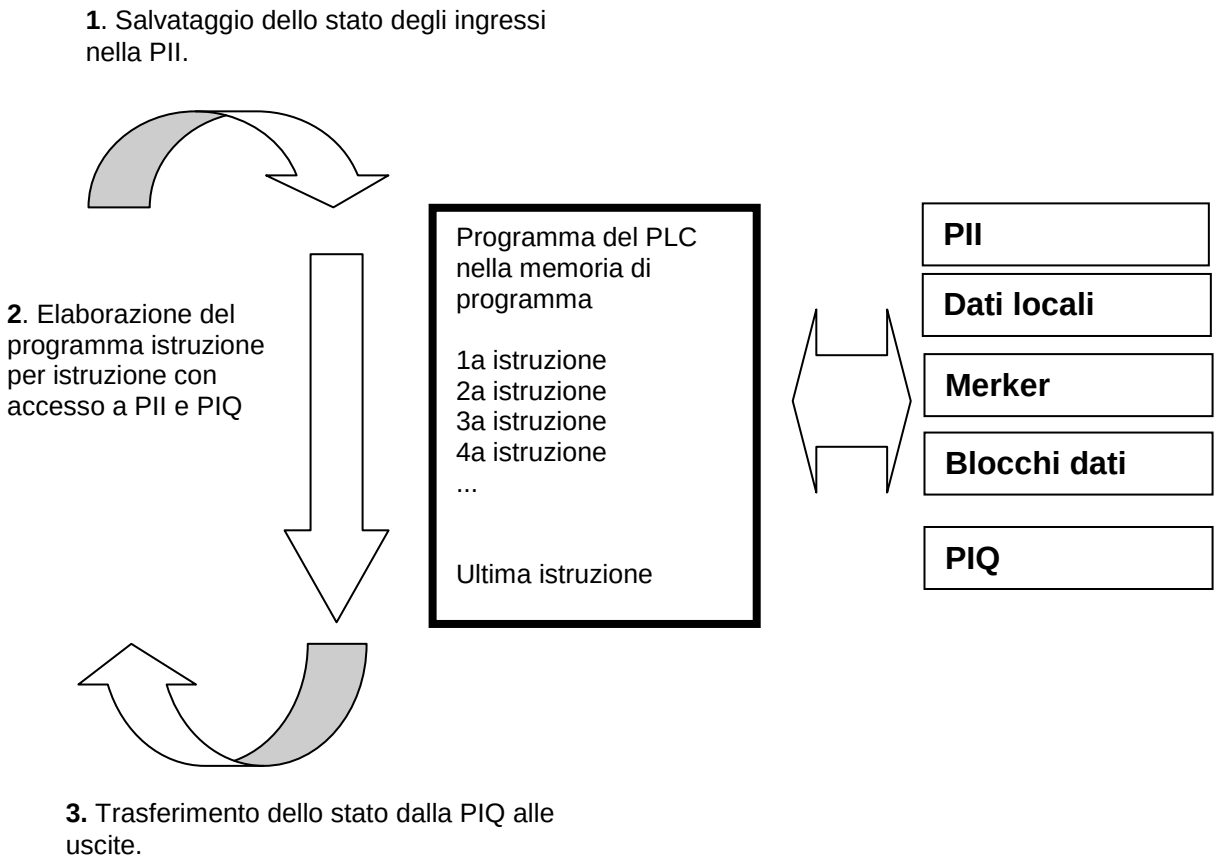


Figura 2: elaborazione ciclica del programma

Nota: *il tempo impiegato dal processore per eseguire questa sequenza è definito tempo di ciclo. Il tempo di ciclo a sua volta varia sia in funzione del numero e del tipo di istruzioni sia della potenza del processore del controllore.*

4.4 Funzioni

Le funzioni (FC) sono blocchi di codice senza memoria. Le funzioni ***non sono provviste di memoria dati*** in cui salvare i valori dei parametri dei blocchi. Per questo motivo quando una funzione viene richiamata tutti i parametri di interfaccia devono essere collegati. Per il salvataggio permanente dei dati è necessario creare prima dei blocchi dati globali.

Una funzione contiene un programma che viene sempre eseguito quando la funzione viene richiamata da un altro blocco di codice.

Le funzioni possono essere impiegate ad es. per i seguenti scopi:

- Funzioni matematiche che restituiscono un risultato in funzione di valori di ingresso.
- Funzioni tecnologiche come i controlli singoli con operazioni binarie.

Una funzione può essere richiamata anche più volte in diversi punti all'interno di un programma.

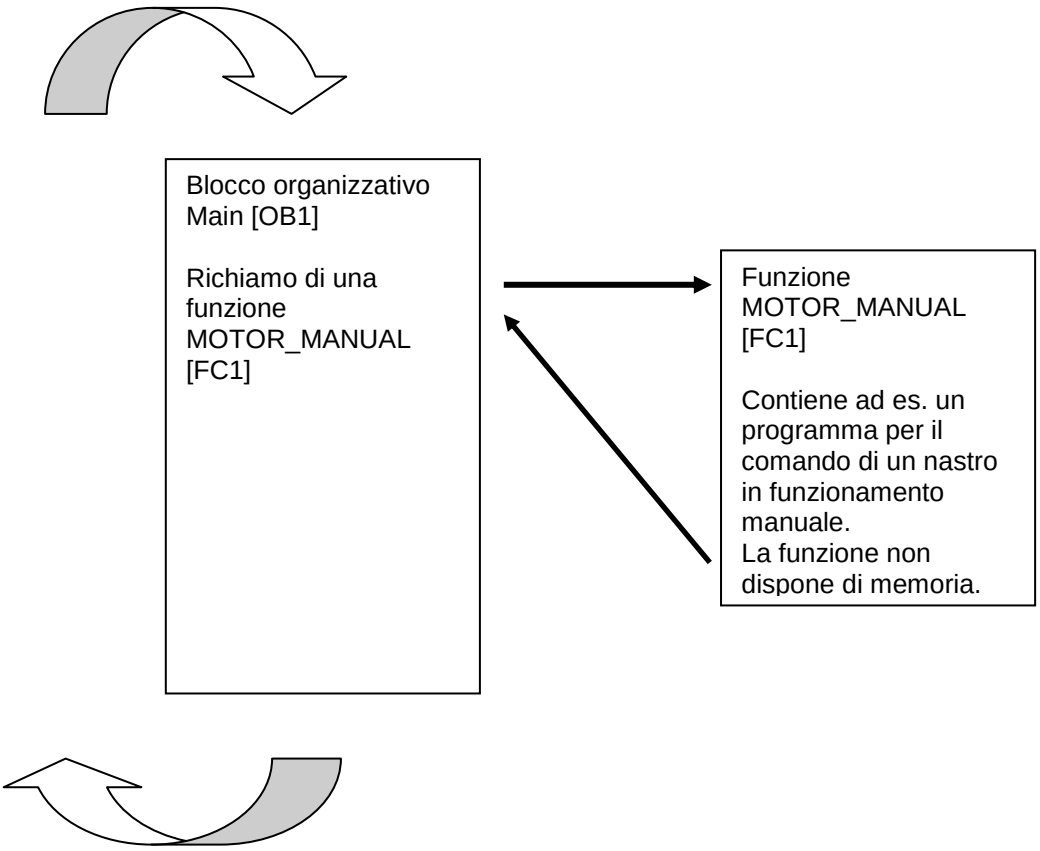


Figura 3: funzione con richiamo del blocco organizzativo Main[OB1]

4.5 Blocchi funzionali e blocchi dati di istanza

I blocchi funzionali sono blocchi di codice che memorizzano in modo permanente le proprie variabili di ingresso, di uscita, di transito e anche le variabili statiche in blocchi dati di istanza così da ***poterne disporre anche dopo l'elaborazione del blocco***. Per questo motivo vengono definiti anche blocchi con "memoria".

I blocchi funzionali possono utilizzare anche variabili temporanee. Le variabili temporanee, tuttavia, non vengono salvate nel DB di istanza ma restano a disposizione solo per un ciclo.

I blocchi funzionali vengono utilizzati per quei compiti che non si possono realizzare con le funzioni:

- Ogni volta che nei blocchi sono necessari temporizzatori e contatori.
- Ogni volta che un'informazione deve essere salvata nel programma. Un esempio è la preselezione del modo di funzionamento con un tasto.

I blocchi funzionali vengono eseguiti ogni volta che un blocco funzionale viene richiamato da un altro blocco di codice. Un blocco funzionale può anche essere richiamato più volte in punti diversi all'interno di un programma. La programmazione di funzioni complesse che ricorrono di frequente viene notevolmente semplificata.

Il richiamo di un blocco funzionale viene definito istanza. A ogni istanza di un blocco funzionale viene assegnata un'area di memoria che contiene i dati utilizzati dal blocco funzionale. Questa memoria viene messa a disposizione da blocchi dati creati automaticamente dal software.

È anche possibile rendere disponibile la memoria per diverse istanze in un blocco dati come **multiistanza**. Le dimensioni max. dei blocchi dati di istanza variano in funzione della CPU. Le variabili dichiarate nel blocco funzionale determinano la struttura del blocco dati di istanza.

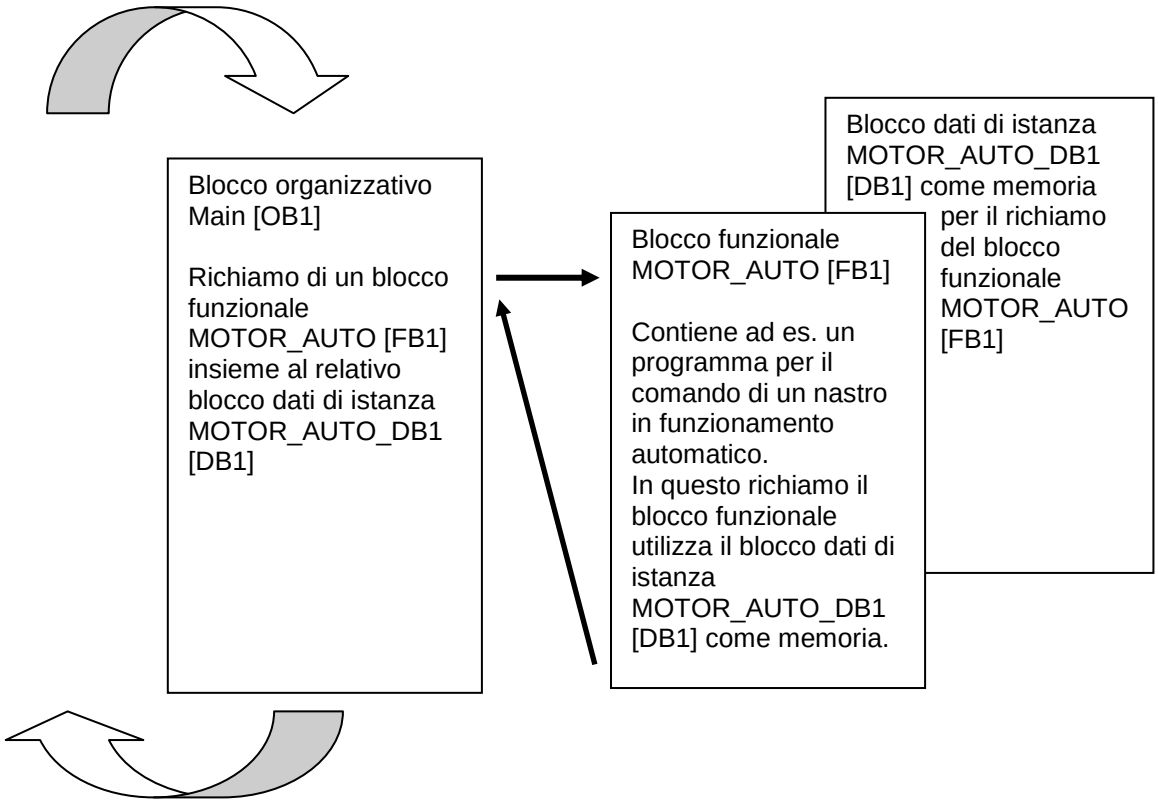


Figura 4: blocco funzionale e istanza con richiamo dal blocco organizzativo Main[OB1]

4.6 Blocchi dati globali

Diversamente dai blocchi di codice i blocchi dati non contengono istruzioni ma fungono da memoria per i dati utente.

I blocchi dati contengono quindi dati variabili che vengono utilizzati dal programma utente. La struttura dei blocchi dati globali si può definire liberamente.

I blocchi dati globali contengono dati che ***possono essere utilizzati da tutti gli altri blocchi*** (vedere figura 5). Ai blocchi dati di istanza deve accedere solo il rispettivo blocco funzionale. Le dimensioni max. dei blocchi dati variano in funzione della CPU.

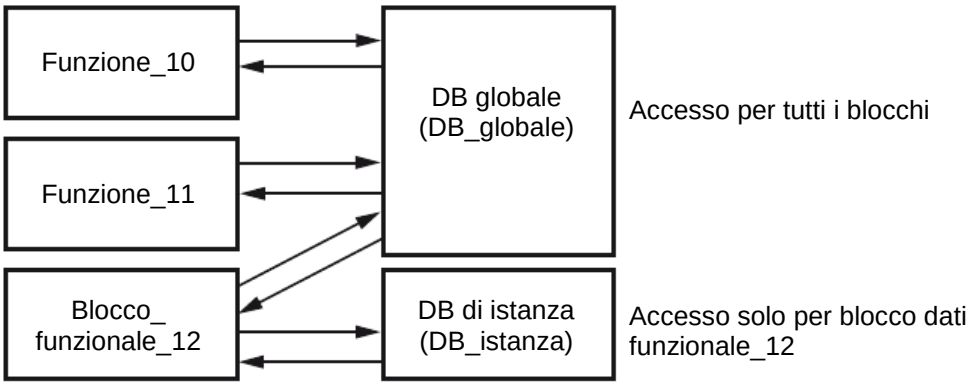


Figura 5: differenza tra DB globale e DB di istanza.

Esempi di applicazione dei **blocchi dati globali**:

- Salvataggio di informazioni relative a un sistema di gestione magazzino. “Dove si trova il tale prodotto?”
- Salvataggio di ricette per determinati prodotti.

4.7 Blocchi di codice gestibili in biblioteche

La creazione di un programma utente può essere lineare o strutturata. La ***programmazione lineare*** scrive l'intero programma utente nell'OB del ciclo ma è indicata solo per programmi utente molto semplici per i quali ormai vengono utilizzati sistemi di controllo più convenienti, come ad es. LOGO!.

Per i programmi più complessi è sempre raccomandata una ***programmazione strutturata***. Qui è possibile suddividere il compito di automazione complessivo in piccoli compiti parziali da risolvere con funzioni e blocchi funzionali.

In questo caso è preferibile creare blocchi di codice gestibili in biblioteche. In altri termini i parametri di ingresso e di uscita di una funzione o di un blocco funzionale vengono definiti in maniera generale e dotati delle attuali variabili globali (ingressi/uscite) solo al momento di utilizzare il blocco.

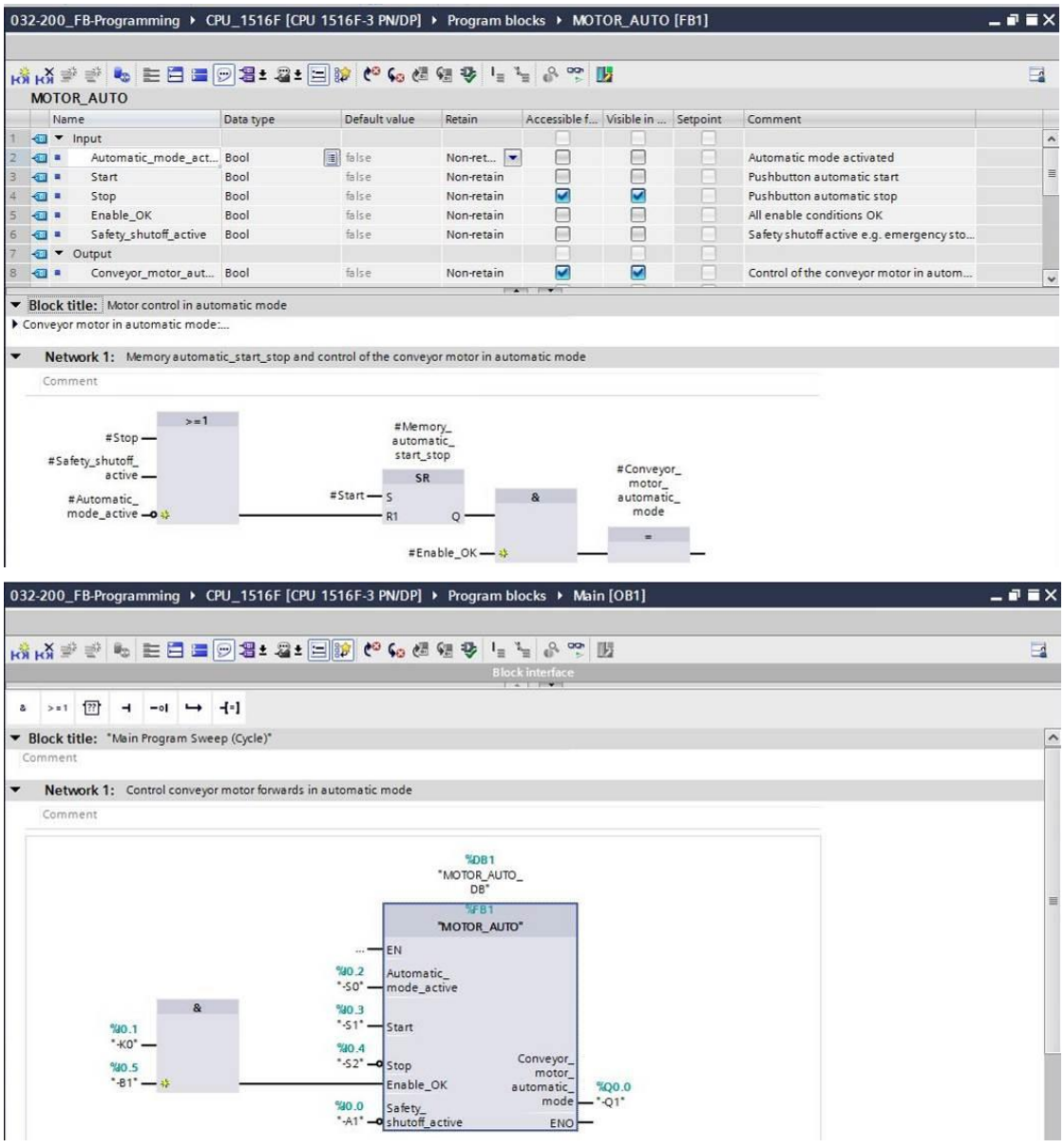


Figura 6: blocco funzionale gestibile in biblioteche con richiamo nell'OB1

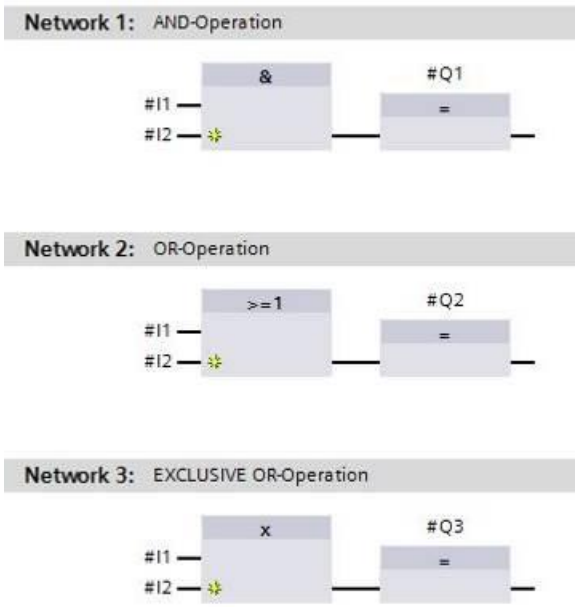
4.8 Linguaggi di programmazione

Per la programmazione delle funzioni sono disponibili i linguaggi di programmazione schema logico (FUP), schema a contatti (KOP), lista istruzioni (AWL) e Structured Control Language (SCL). Per i blocchi funzionali è inoltre disponibile il linguaggio di programmazione GRAPH per la programmazione di sequenze di passi grafiche.

Nel seguito viene descritto il linguaggio di programmazione ***schema logico (FUP)***.

FUP è un linguaggio di programmazione grafico. La rappresentazione è basata su sistemi circuitali elettronici. Il programma viene rappresentato in segmenti. Un segmento contiene uno o più percorsi logici. I segnali binari e analogici vengono collegati tra loro mediante box. Per la rappresentazione della logica binaria vengono utilizzati i simboli logici grafici dell'algebra booleana.

Le funzioni binarie permettono di interrogare gli operandi binari e di collegarne gli stati di segnale. Esempi di funzioni binarie sono le istruzioni "Combinazione logica AND", "Combinazione logica OR" e "Combinazione logica OR esclusivo", come mostra la Figura 7.



#e1	#e2	#a1
0	0	0
0	1	0
1	0	0
1	1	1

#e1	#e2	#a2
0	0	0
0	1	1
1	0	1
1	1	1

#e1	#e2	#a3
0	0	0
0	1	1
1	0	1
1	1	0

Figura 7: funzioni binarie in FUP e rispettiva tabella logica

Con istruzioni semplici si possono ad es. comandare uscite binarie, valutare fronti o eseguire funzioni di salto nel programma.

Le istruzioni complesse mettono a disposizione elementi di programma come ad es. temporizzatori IEC e contatori IEC.

Un box vuoto funge da segnaposto nel quale è possibile selezionare l'istruzione desiderata.

Meccanismo ingresso di abilitazione EN (enable) / uscita di abilitazione ENO (enable output):

- Un'istruzione senza meccanismo EN/ENO viene eseguita negli ingressi del box indipendentemente dallo stato del segnale.
- Le istruzioni con meccanismo EN/ENO vengono eseguite solo se l'ingresso di abilitazione "EN" ha lo stato di segnale "1". Se l'elaborazione del box è regolare l'ingresso di abilitazione "ENO" ha lo stato di segnale "1". Non appena si verifica un errore nel corso dell'elaborazione, l'uscita di abilitazione "ENO" viene resettata. Se l'ingresso di abilitazione EN non è interconnesso, il box viene sempre eseguito.

5 Definizione del compito

Lo scopo di questo capitolo è di pianificare, programmare e testare le seguenti funzioni della descrizione del processo “impianto di smistamento”:

- Funzionamento automatico – motore nastro

6 Pianificazione

Per questioni di visibilità di insieme e di riusabilità si consiglia di non programmare tutte le funzioni nell’OB1. Il codice di programma perciò viene dislocato in funzioni (FC) e blocchi funzionali (FB) per la maggior parte. Con la seguente pianificazione decideremo quali funzioni dislocare nell’FB e quali invece eseguire nell’OB1.

6.1 ARRESTO D’EMERGENZA

L’arresto d’emergenza non richiede una funzione propria. Al pari del modo di funzionamento anche lo stato attuale del relè di arresto d’emergenza può essere utilizzato direttamente nei blocchi.

6.2 Funzionamento automatico – motore nastro

Il funzionamento automatico del motore del nastro deve essere “incapsulato” in un blocco funzionale (FB) “MOTOR_AUTO”. Da un lato ciò garantisce la visibilità d’insieme nell’OB1 e dall’altro è garantita la riusabilità in caso di ampliamento dell’impianto con un ulteriore nastro trasportatore. Nella tabella 2 sono riportati i parametri pianificati.

Input	Tipo di dati	Commento
Funzionamento_automatico_attivo	BOOL	Modo di funzionamento automatico attivato
Start	BOOL	Comando di avvio del funzionamento automatico
Stop	BOOL	Comando di arresto del funzionamento automatico
Abilitazione_OK	BOOL	Tutte le condizioni di abilitazione sono soddisfatte
Disattivazione_protezione_attiva	BOOL	Disinserzione di protezione attiva, ad es. arresto d’emergenza azionato
Output		
Motore_nastro_automatico	BOOL	Comando del motore del nastro in funzionamento automatico
Static		
Memoria_automatica_Start/Stop	BOOL	Memoria per le funzioni di avvio e arresto in funzionamento automatico

Tabella 2: parametri per FB “MOTOR_AUTO”

Il parametro Memoria_automatica_Start/Stop viene attivato con memoria con il comando Start, ma solo se non sono presenti le condizioni di reset.

Il parametro Memoria_automatica_Start/Stop viene resettato se è presente il comando Stop, se è attiva la disinserzione di protezione o se non è attivo il funzionamento automatico (funzionamento manuale).

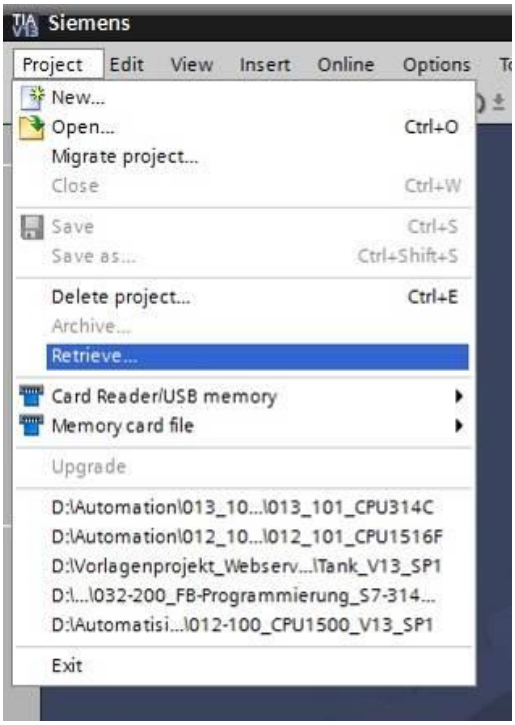
L’uscita Motore_nastro_automatico viene comandata se è impostata la Memoria_automatica_Start/Stop e se sono soddisfatte le condizioni di abilitazione.

7 Istruzioni strutturate passo passo

Qui di seguito sono riportate le istruzioni necessarie per poter realizzare la pianificazione. Per chi ha già dimestichezza sarà sufficiente eseguire i passi numerati. Diversamente, leggere la descrizione dei passi descritti dettagliatamente nelle istruzioni.

7.1 Disarchiviare un progetto esistente

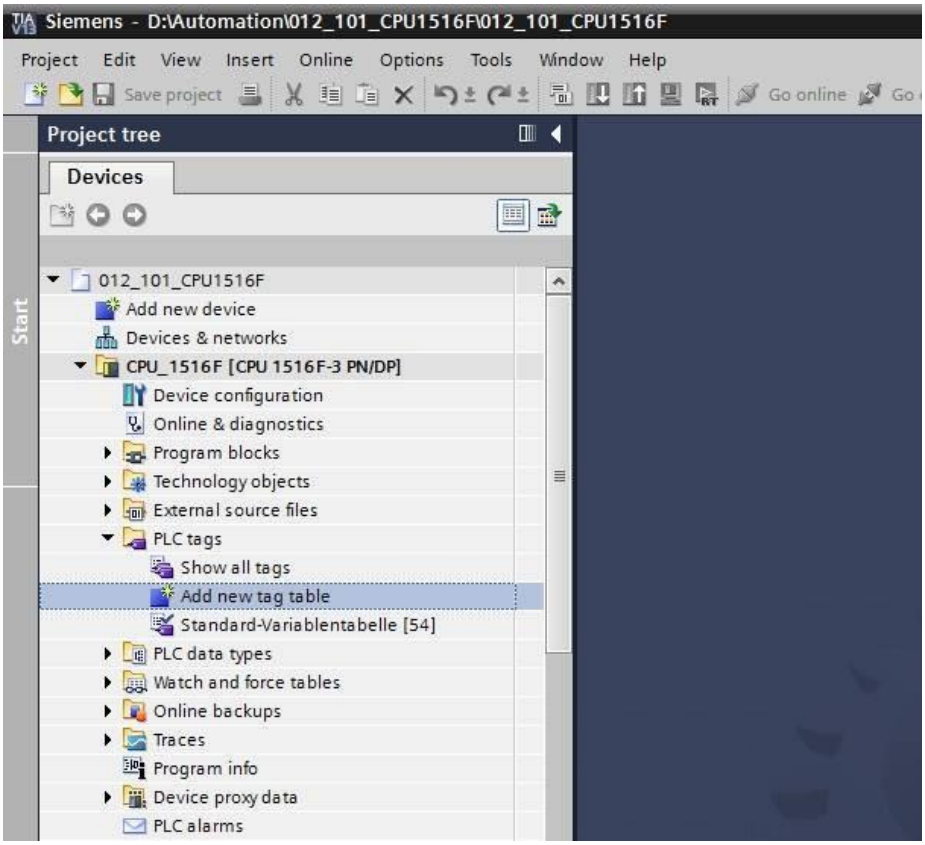
→ Prima di poter iniziare a programmare il blocco funzionale (FB) “MOTOR_AUTO” è necessario un progetto con una configurazione hardware (ad es. SCE_IT_012_101_Configurazione hardware_S7-1516F_R1502.zap). Per disarchiviare un progetto esistente è necessario cercare l'archivio specifico nella vista del progetto con → Project → Retrieve. Quindi confermare la selezione con "Open”. (→ Progetto → Disarchivia → selezionare un archivio .zap → Apri)



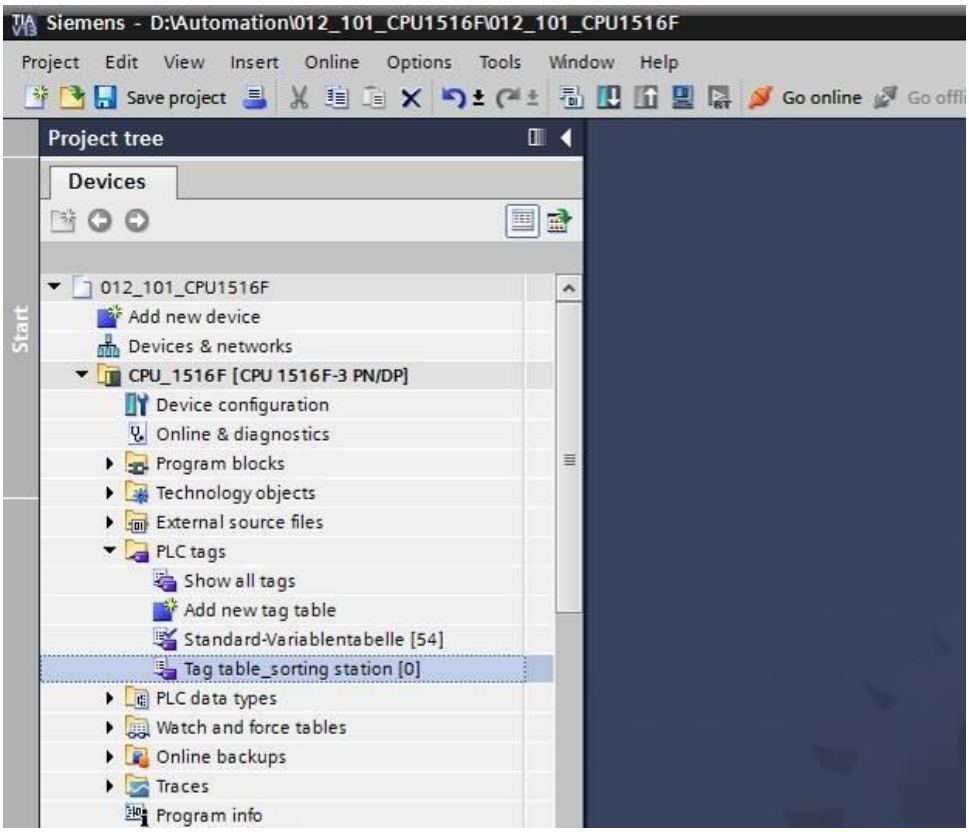
→ Ora è possibile selezionare la directory di destinazione nella quale salvare il progetto disarchiviato. Confermare la selezione con "OK". (→ Directory di destinazione → OK)

7.2 Creazione di una nuova tabella delle variabili

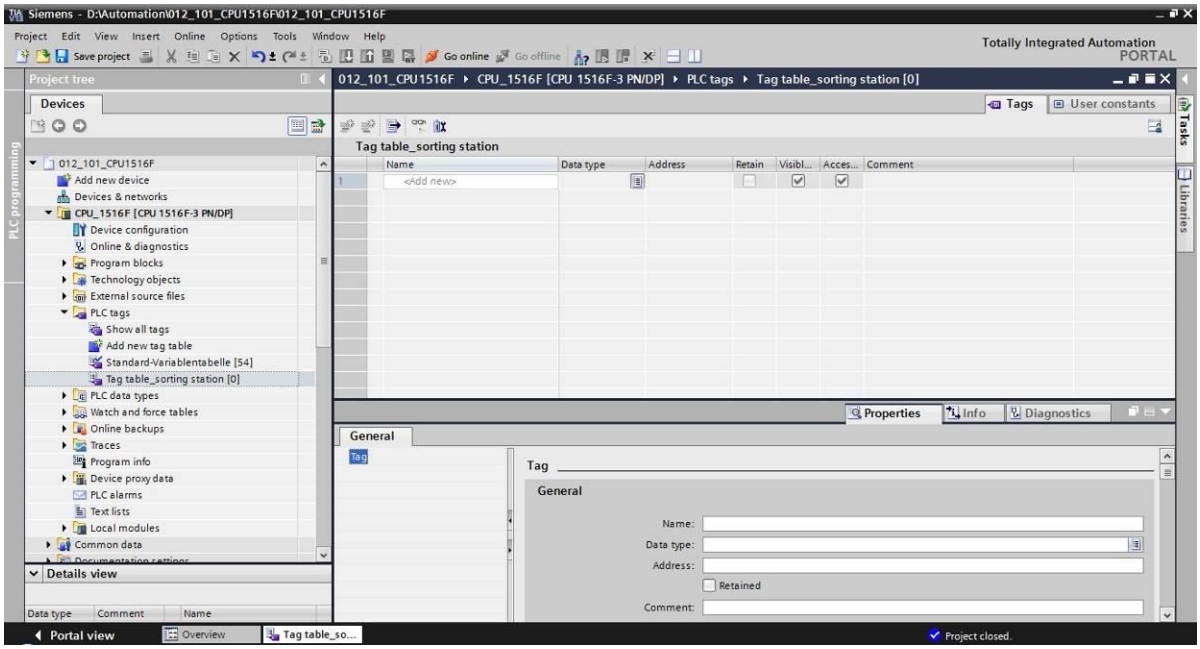
→ Nella vista progetto spostarsi alle → variabili PLC del controllore in uso e creare una nuova tabella delle variabili facendo doppio clic su → “Add new tag table” (Aggiungi nuova tabella delle variabili).



→ Rinominare la tabella delle variabili appena creata in “Tag_table_sorting station” (Tabella_variabili_stazione_smistamento). (→ Clic con il tasto destro del mouse su “Tag_table_1” / Tabella delle variabili_1 → “Rename” / Rinomina → “Tag_table_sorting station” / Tabella_variabili_stazione_smistamento)

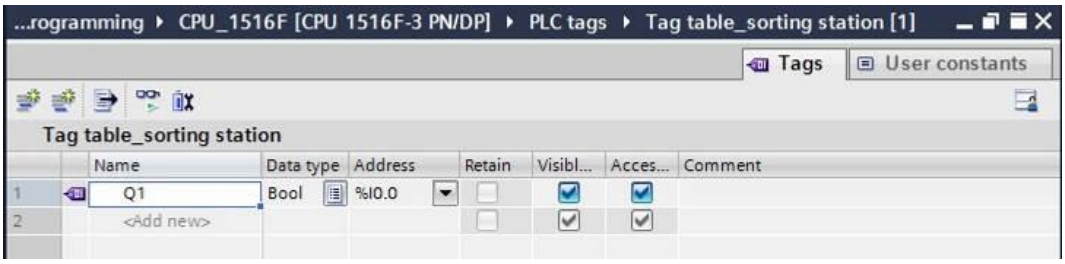


→ Aprire la tabella con un doppio clic. (→ Tag_table_sorting station / Tabella_variabili_stazione_smistamento)

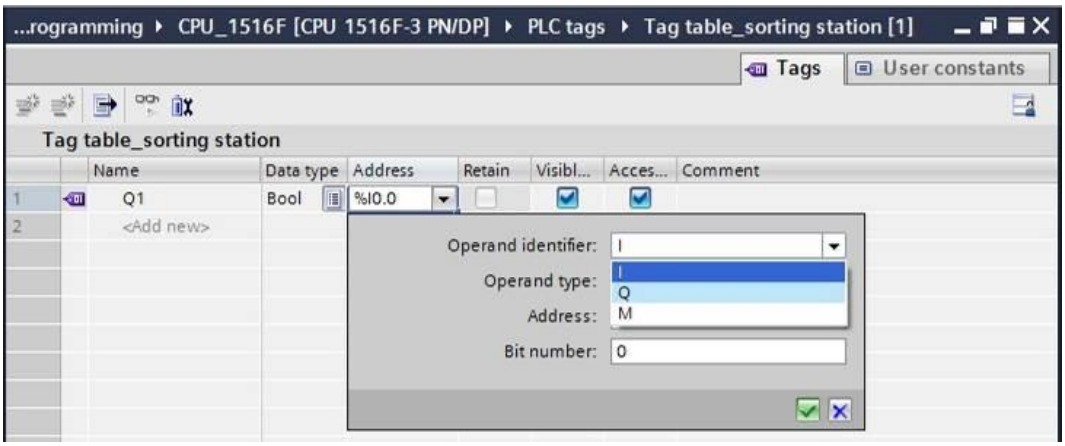


7.3 Creazione di nuove variabili in una tabella delle variabili

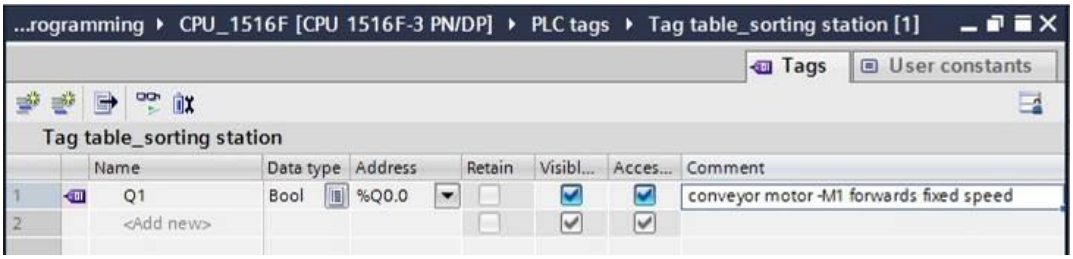
→ Aggiungere il nome Q1 e confermare con il tasto Invio. Se non si è ancora creata una tabella delle variabili, TIA Portal assegna automaticamente il tipo di dati “Bool” e l’indirizzo %E0.0 (I 0.0). (→ <Add new> / <Aggiungi>→ Q1 → Invio)



→ Modificare l’indirizzo in %A0.0 (Q0.0) inserendolo direttamente o dal menu per l’indirizzamento che si apre con un clic sulla freccia della casella di riepilogo. Modificare l’identificatore operando in Q e confermare con il tasto Invio o con un clic sul segno di spunta. (→ %I0.0 → Operand identifier / Identificatore operando → Q →)

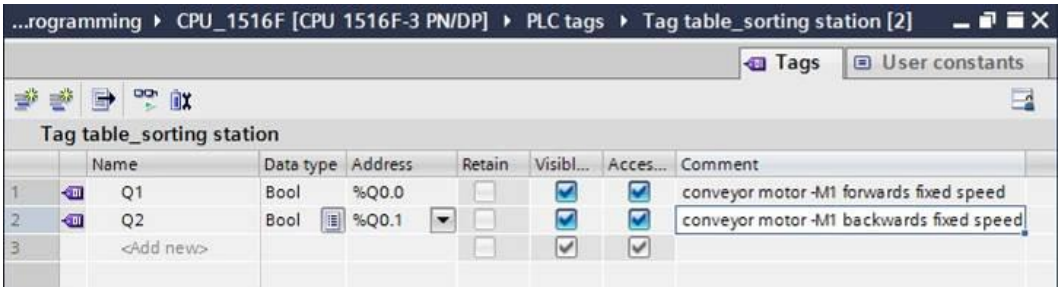


→ Assegnare alla variabile il commento “conveyor motor M1 forwards fixed speed” (motore nastro M1 in avanti numero di giri fisso).



→ Inserire nella riga 2 una nuova variabile Q2. TIA Portal ha assegnato automaticamente lo stesso tipo di dati della riga 1 e incrementato di 1 l'indirizzo, che diventa %A0.1 (Q0.1). Inserire il commento "conveyor motor M1 backwards fixed speed" (motore nastro M1 all'indietro numero di giri fisso).

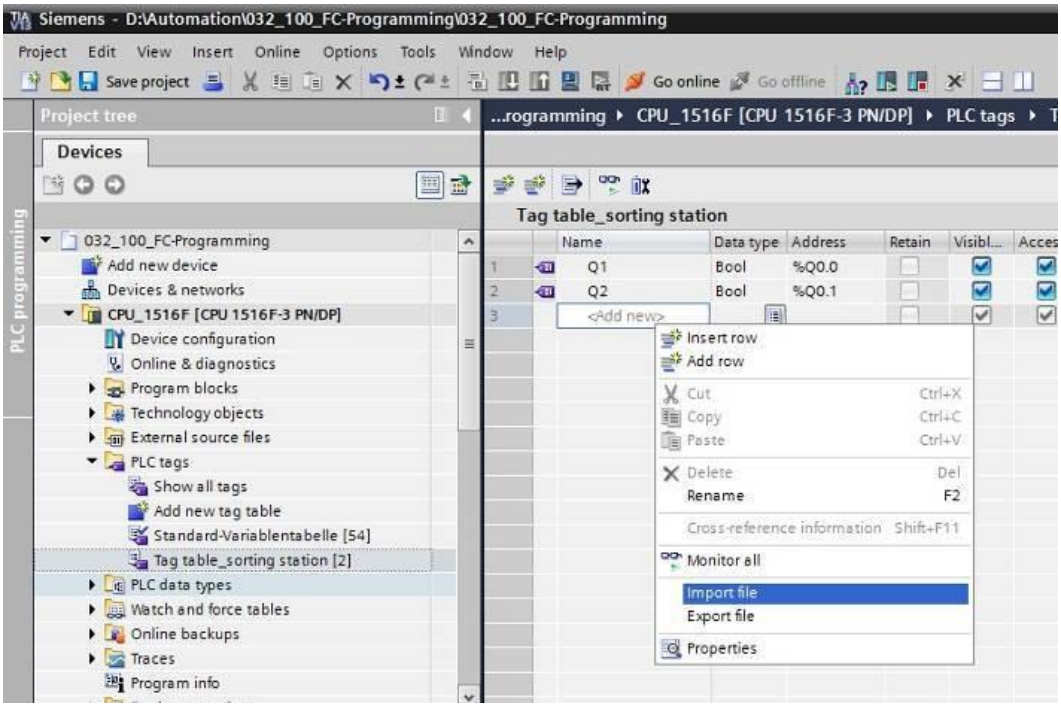
(→ <Add new> / <Aggiungi> → Q2 → Invio → commento → "conveyor motor M1 backwards fixed speed" / motore nastro M1 all'indietro numero di giri fisso)



7.4 Importazione della tabella Tag_table_sorting station / Tabella_variabili_stazione_smistamento

→ Per l'inserimento di una tabella dei simboli esistente fare clic con il tasto destro del mouse su un campo vuoto della tabella "Tag_table_sorting station" (Tabella_variabili_stazione_smistamento). Selezionare nel menu di scelta rapida "Import file".

(→ clic con il tasto destro del mouse in un campo vuoto della tabella delle variabili → File di importazione)



→ Selezionare la tabella dei simboli desiderata (ad es. in formato .Xlsx) e confermare con “Open”.

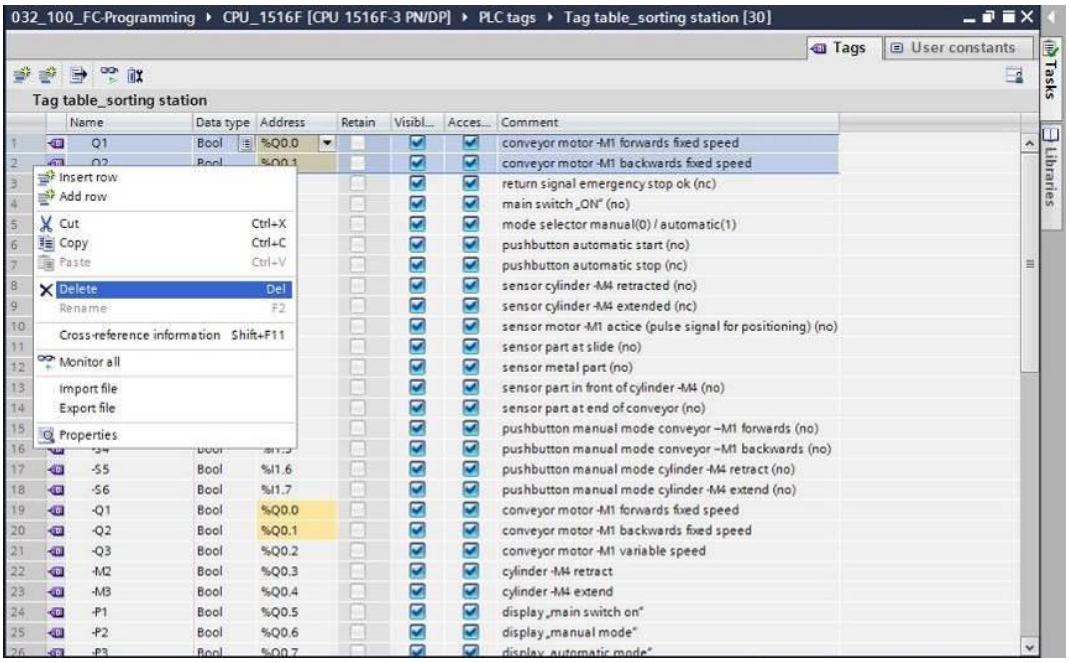
(→ SCE_IT_020-100_Tabella delle variabili impianto di smistamento… → Apri)

→ Al termine dell’importazione viene visualizzata una finestra di conferma che consente di visualizzare il file di protocollo dell’importazione. Fare clic su → OK.

→ Si vedrà che alcuni indirizzi sono evidenziati in color arancione. Significa che sono doppi e i nomi delle rispettive variabili sono stati automaticamente rinumerati per evitare la mancanza di chiarezza.

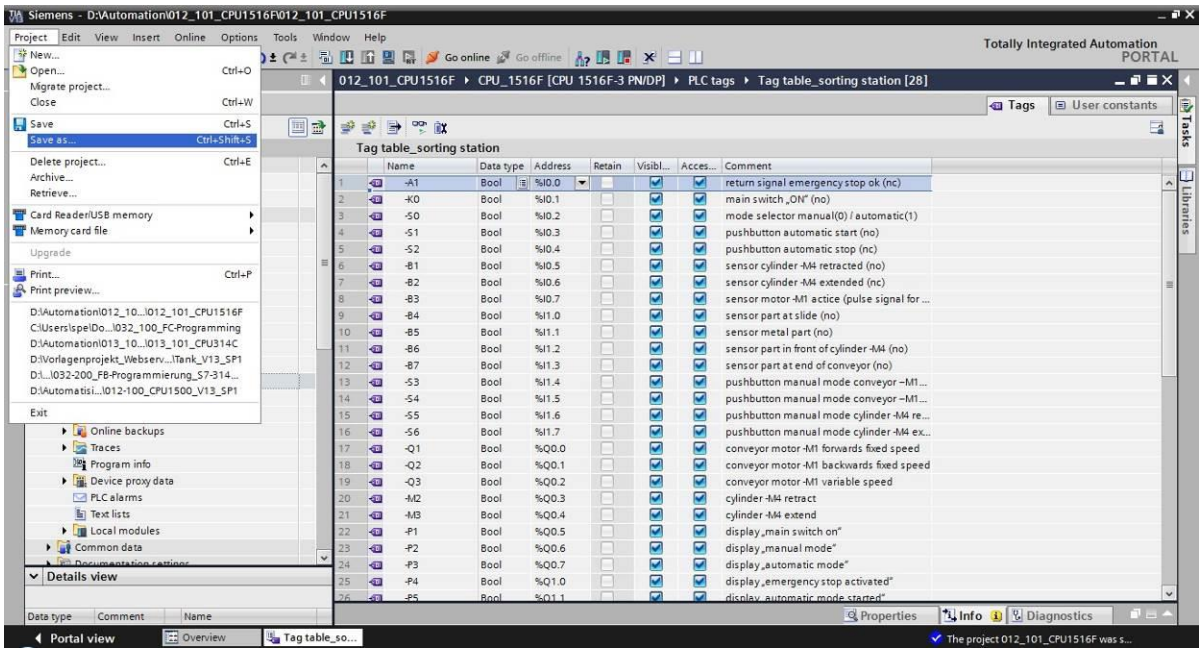
→ Cancellare le variabili doppie selezionando le righe e premendo il tasto Canc sulla tastiera o selezionando il comando “Delete” nel menu di scelta rapida.

(→ Clic con il tasto destro sulla variabile selezionata → Elimina)



→ Ora viene visualizzata una tabella dei simboli completa degli ingressi e delle uscite digitali. Salvare il progetto con il nome 032-100_Programmazione di FC.

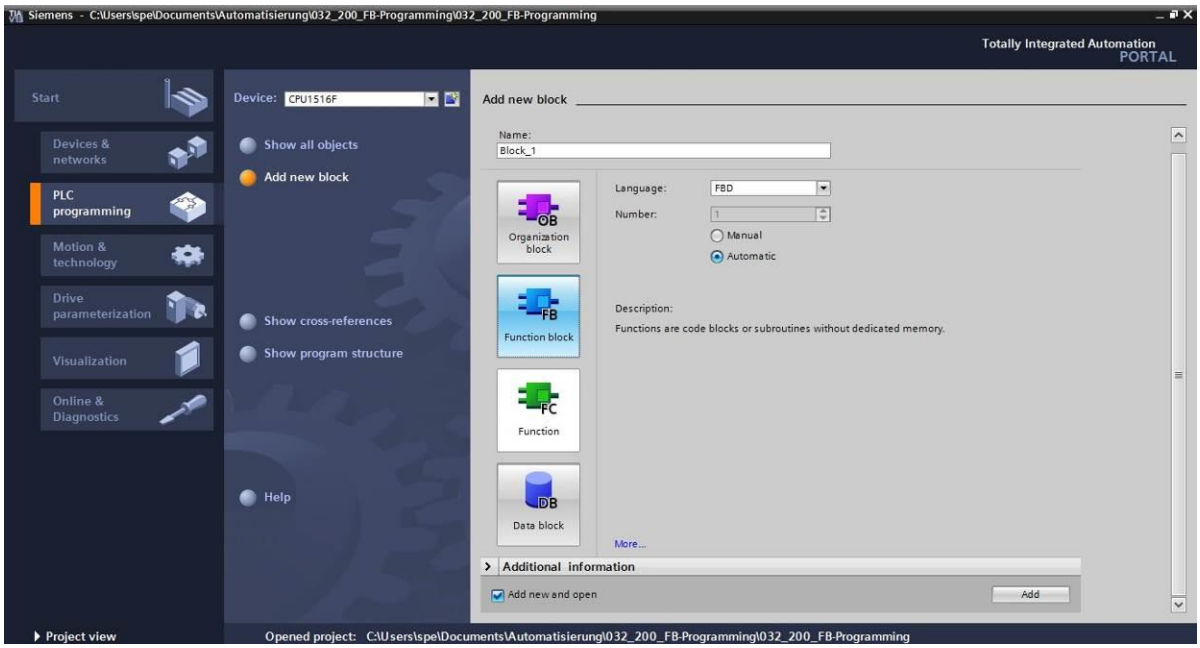
(→ Project / Progetto → Save as… / Salva con nome... → 032-200_Programmazione di FB → Save / Salva)



7.5 Creazione del blocco funzionale FB1 “MOTOR_AUTO” per il motore del nastro in funzionamento automatico

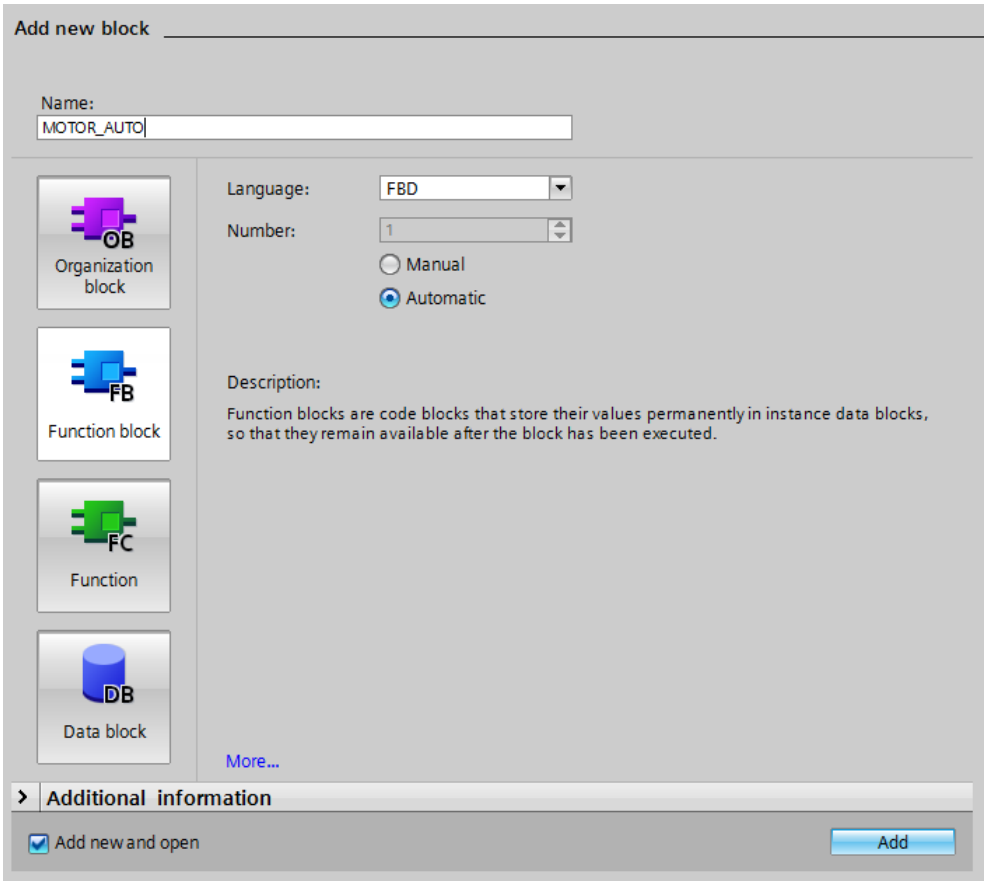
→ Nella vista portale fare clic su “Add new block” nella sezione “PLC programming” per creare un nuovo blocco funzionale.

(→ Programmazione PLC → Inserisci nuovo blocco → )



→ Nominare il nuovo blocco: “MOTOR_AUTO”, impostare il linguaggio FBD e far assegnare il numero automaticamente (Number > Automatic). Spuntare la casella “Add new and open” per accedere automaticamente al blocco funzionale creato nella vista progetto.Fare clic sul pulsante “Add”.

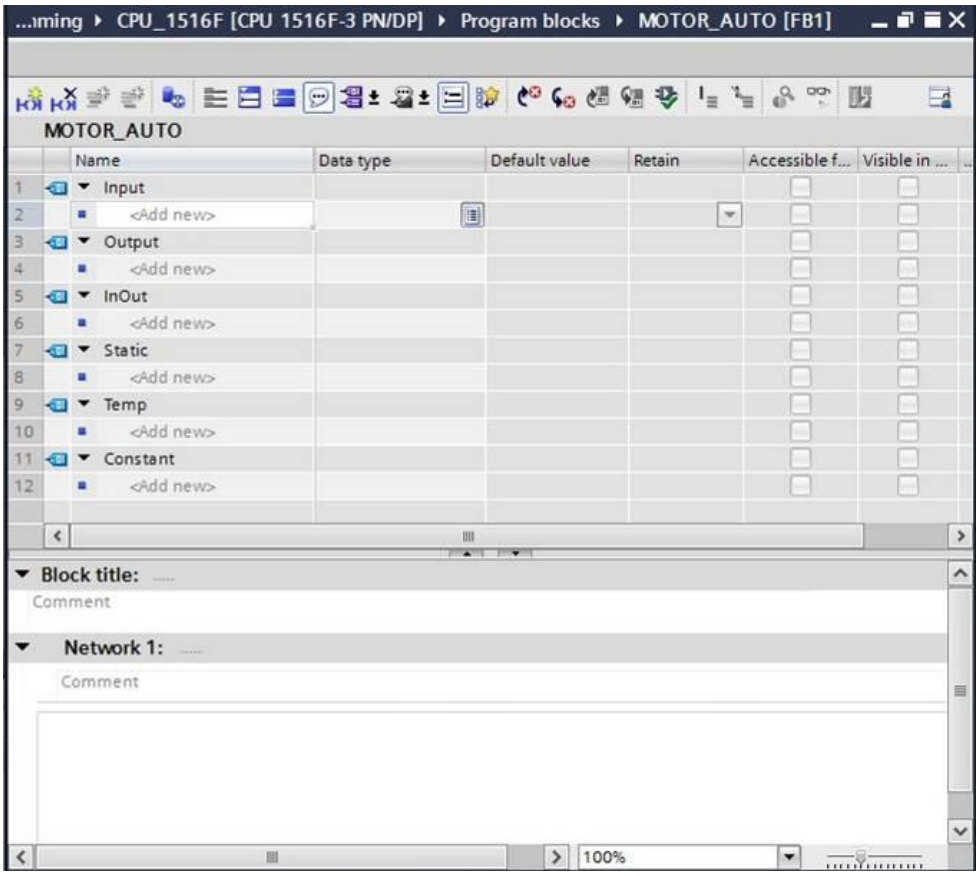
(→ Nome: MOTOR_AUTO→ Linguaggio: FUP → Numero: Automatico → Aggiungi e apri → Aggiungi)



7.6 Definizione dell’interfaccia dell’FB1 “MOTOR_AUTO”

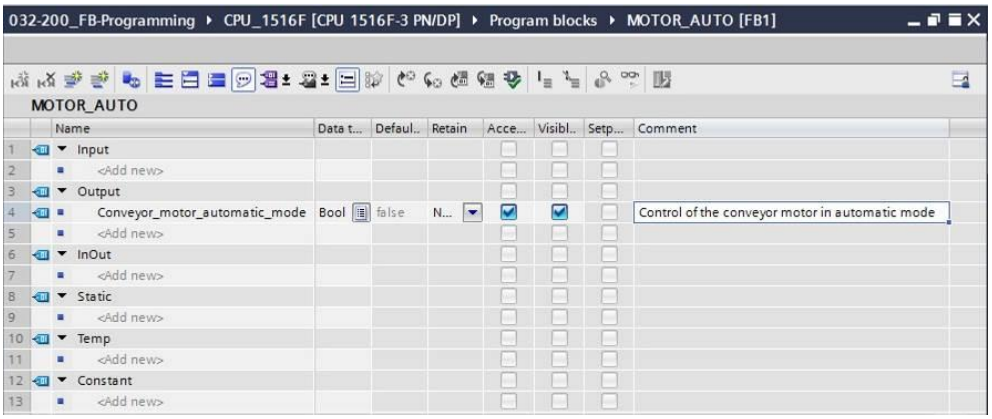
→ Dopo aver fatto clic su “Add new and open” (Aggiungi e apri) si apre la vista progetto con una finestra per la generazione del blocco appena creato.

→ Nella sezione superiore della finestra di programmazione compare la descrizione dell’interfaccia del blocco funzionale.



→ Per il comando del motore del nastro è necessario un segnale di uscita binario. Per questo creeremo prima la variabile Output locale #conveyor_motor_automatic_mode del tipo “Bool”. Assegnare al parametro il commento “Control of the conveyor motor in automatic mode”.

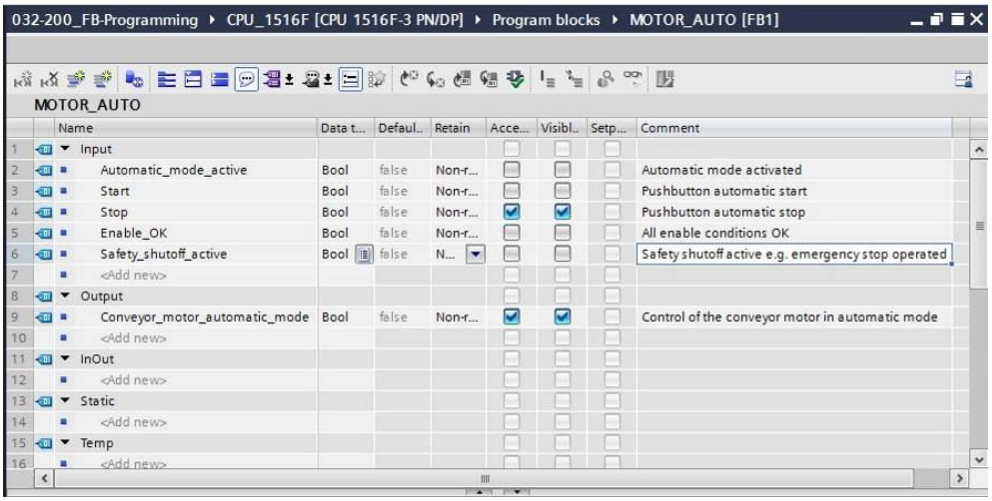
(→ Output: Motore_nastro_automatico → Bool → Comando del motore del nastro in funzionamento automatico)



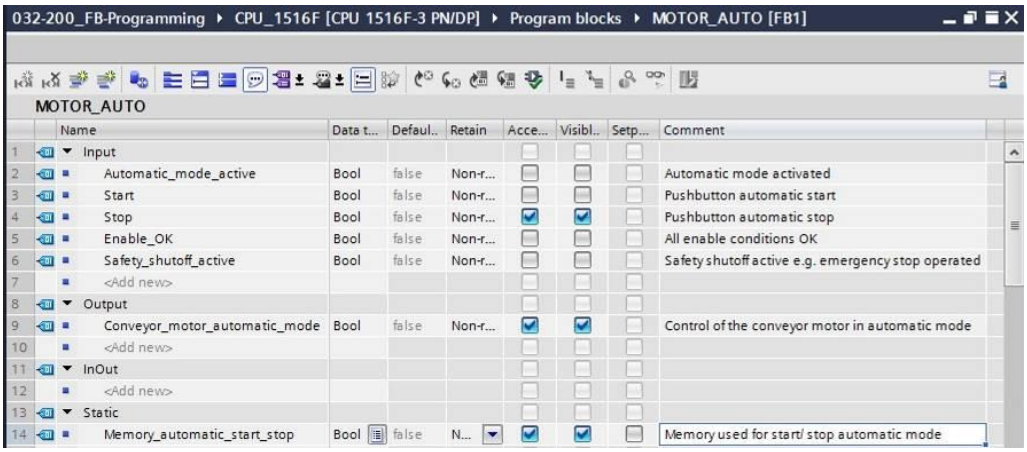
→ Inserire come interfaccia di ingresso Input prima il parametro #Automatic_mode_active e confermare con il tasto Invio o uscendo dal campo di immissione. Viene assegnato automaticamente il tipo di dati “Bool”. Questo viene mantenuto. Successivamente inserire il commento corrispondente “Automatic mode activated”.

(→ Funzionamento_manuale_attivo → Bool → Modo di funzionamento automatico attivato)

→ Inserire alla voce Input gli ulteriori parametri di ingresso binari #Start, #Stop, #Enable_OK e #Safety_shutoff_active e verificarne i tipi di dati. Completare aggiungendo commenti opportuni.

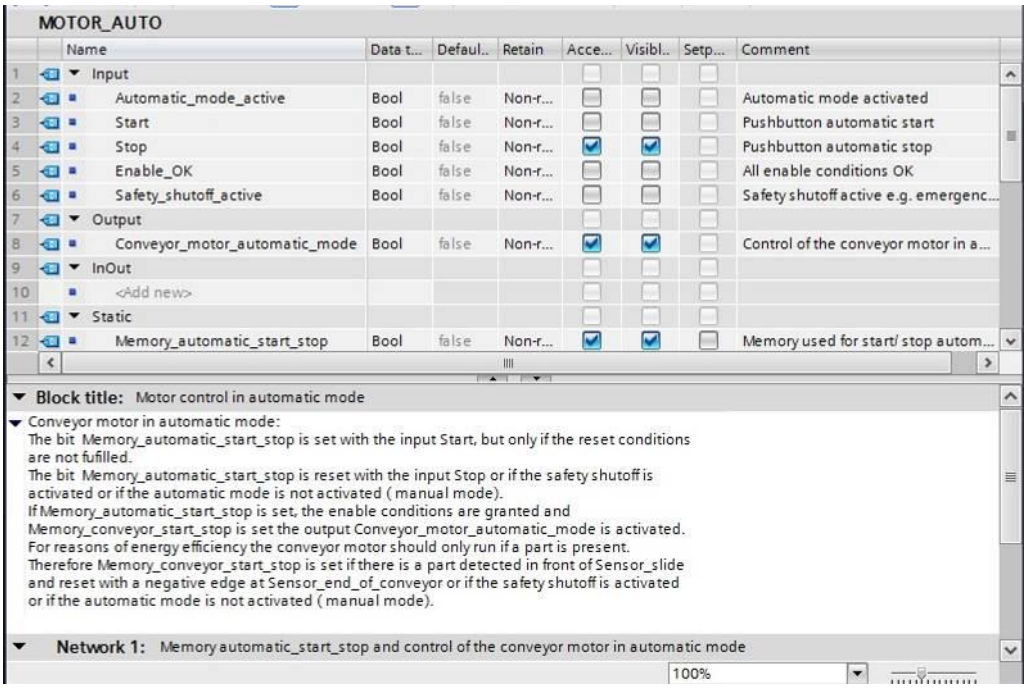


L'avvio e l'arresto del nastro vengono comandati da tasti. Perciò è necessaria una variabile “Static” come memoria. Inserire alla voce Static la variabile #Memory_automatic_start_stop e confermare con il tasto Invio o uscendo dal campo di immissione. Viene assegnato automaticamente il tipo di dati “Bool”. Questo viene mantenuto. Inserire il commento corrispondente “Memory used for start/stop automatic mode”. (→ Memoria_automatica_Start_Stop → Bool → Memoria per le funzioni di avvio e arresto in funzionamento automatico)



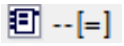
→ Assegnare alla documentazione del programma il titolo del blocco, un commento al blocco e un titolo significativo per il segmento 1.

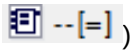
(→ Block title / Titolo del blocco: Motor control in automatic mode / Comando motore in funzionamento automatico → Network 1 / Segmento 1: Memory automatic_start_stop and control of the conveyor motor in automatic mode / Memoria_automatica_Start_Stop e comando del motore del nastro in funzionamento automatico

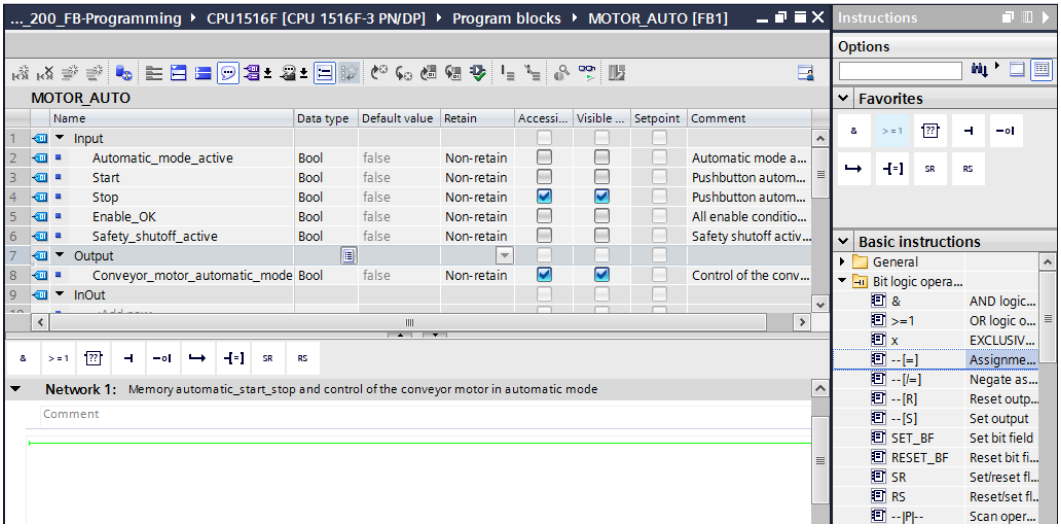


7.7 Programmazione dell'FB1: MOTOR_AUTO

→ Sotto la descrizione dell'interfaccia è visibile nella finestra di programmazione una barra degli strumenti con diverse funzioni logiche e, sotto di essa, un'area con segmenti. Qui abbiamo già definito il titolo del blocco e il titolo del primo segmento. All'interno dei segmenti la programmazione si effettua con l'uso di singoli blocchi logici. La suddivisione in diversi segmenti consente di mantenere la visibilità dell'insieme. Qui di seguito vedremo le varie possibilità di inserire i blocchi logici.

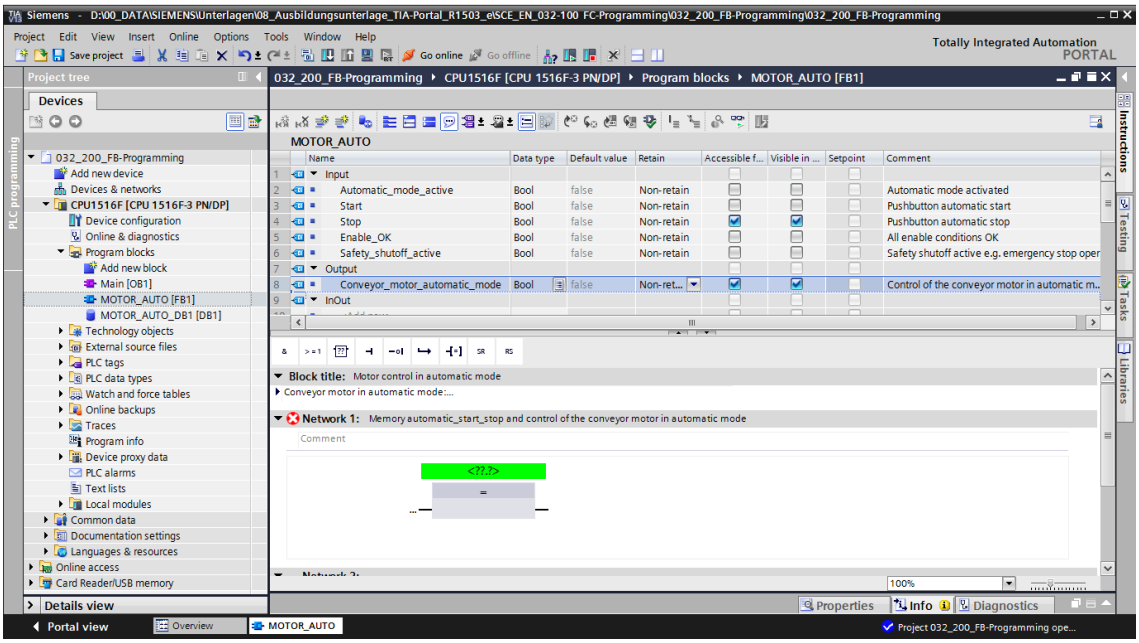
→ Sulla destra della finestra di programmazione è visibile un elenco di istruzioni (Instructions) che si possono utilizzare nel programma. Alla voce → Basic instructions → Bit logic operations cercare la funzione  (assegnazione) e trascinarla nel segmento 1 (compare una linea verde, puntatore del mouse con simbolo +).

(→ Istruzioni → Istruzioni di base → Combinazioni logiche di bit → )

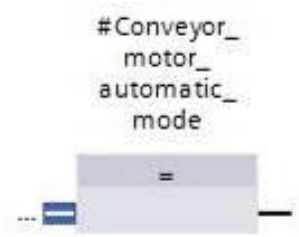


→ Trascinare il parametro Output #Conveyor_motor_automatic_mode su <??.?> sopra il blocco appena inserito. Per selezionare un parametro nella descrizione dell'interfaccia è preferibile acquisirlo dal simbolo blu .

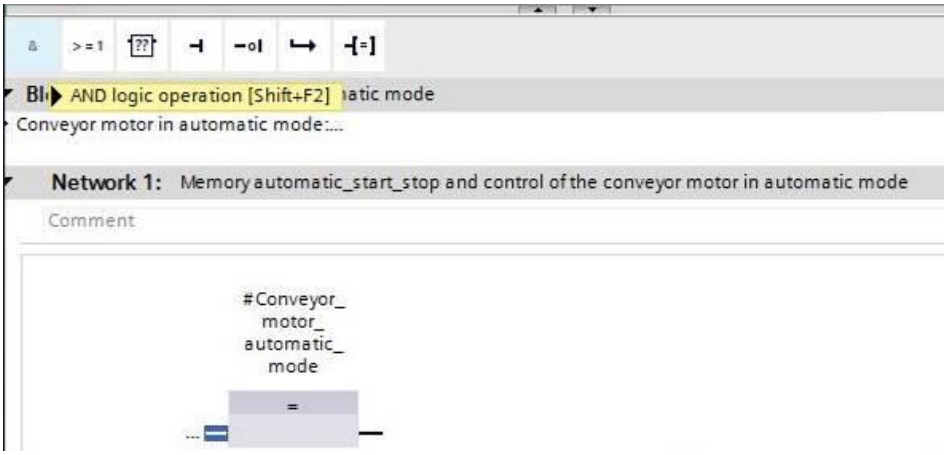
(→ Motore_nastro_automatico)



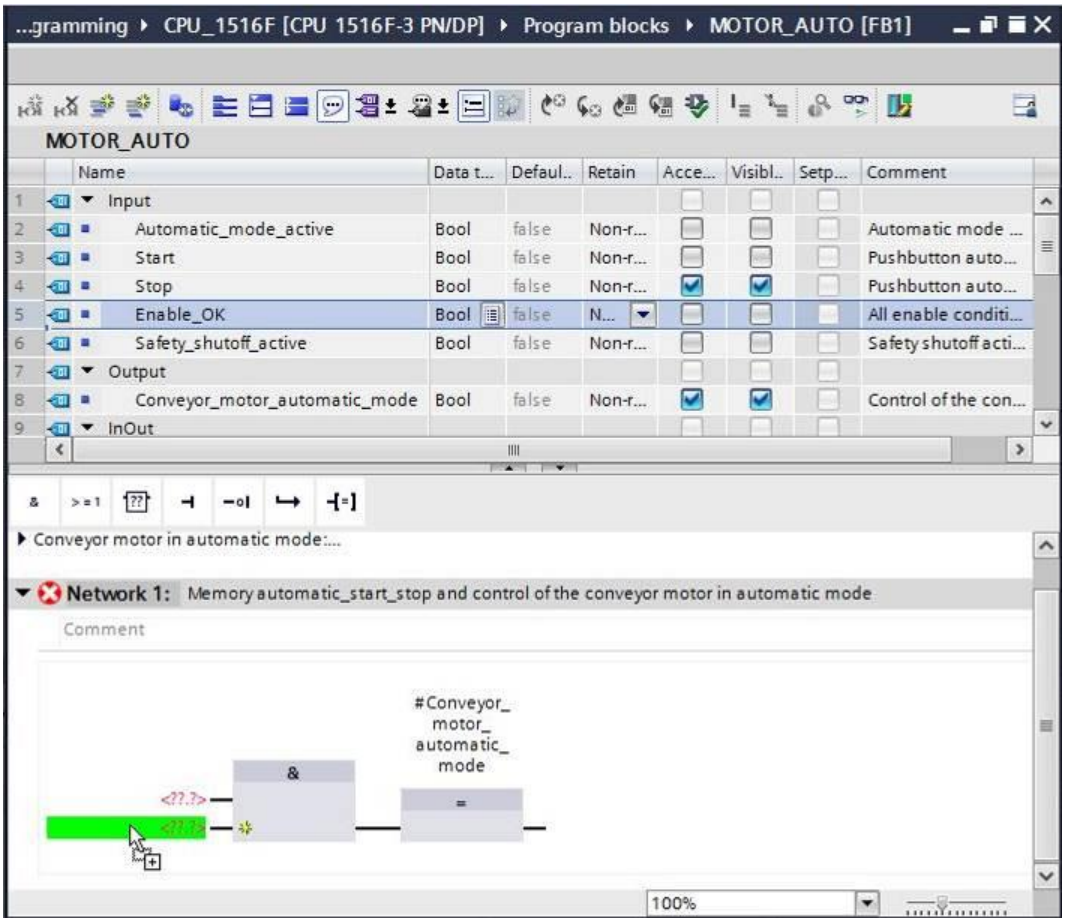
→ In questo modo si stabilisce che il parametro #Conveyor_motor_automatic_mode venga scritto da questo blocco. Tuttavia mancano ancora le condizioni di ingresso perché ciò succeda veramente. All'ingresso del blocco di assegnazione devono essere collegati tramite AND un'istruzione SR Flipflop e il parametro #Enable_OK. Fare clic prima sull'ingresso del blocco in modo che il trattino dell'ingresso abbia lo sfondo blu.



→ Fare clic sul simbolo & nella barra dei simboli logici per inserire una combinazione logica AND davanti al blocco di assegnazione.

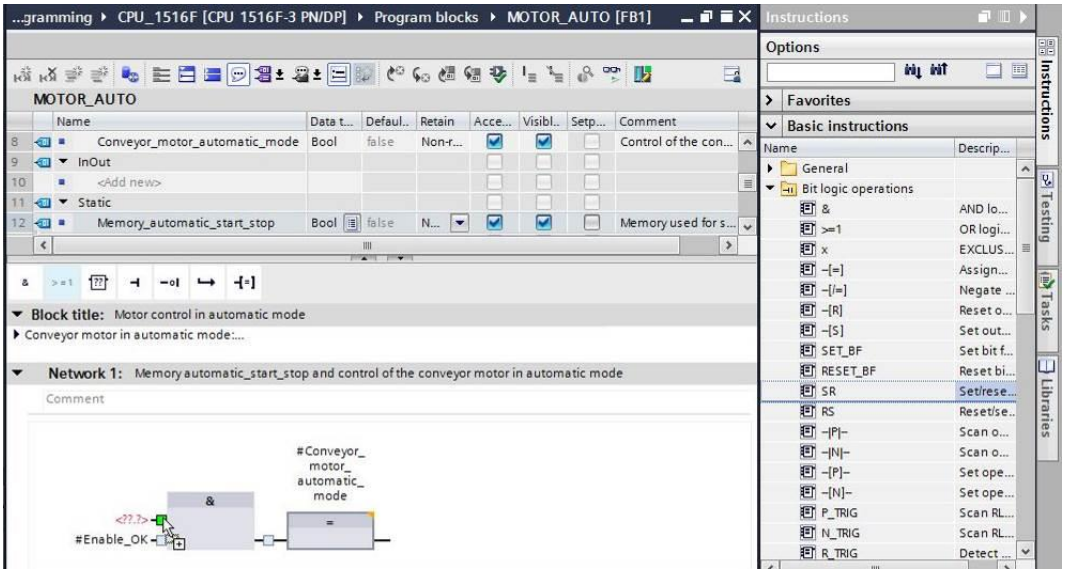


→ Trascinare il parametro Input #Enable_OK sul secondo ingresso della combinazione logica & <??.?>. (→ Abilitazione_OK)

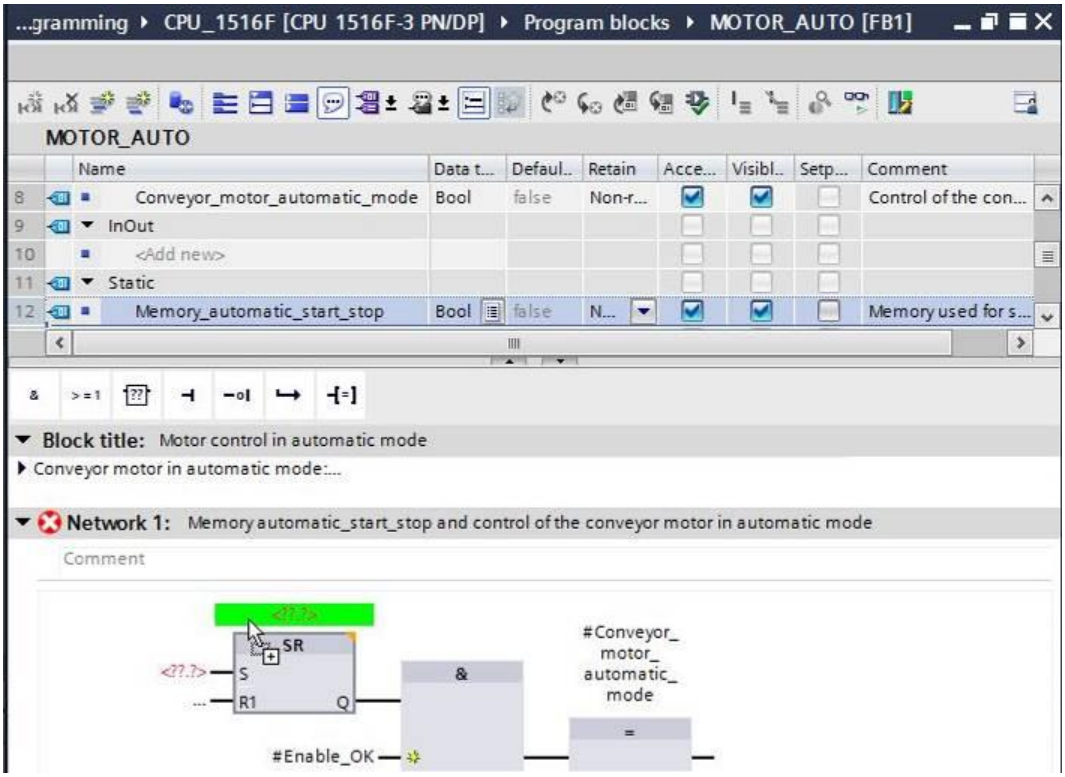


→ Trascinare dall'elenco Instructions → Basic instructions → Bit logic operations la funzione Set/Reset Flipflop SR sul primo ingresso della combinazione logica &.

(→ Istruzioni → Istruzioni di base → Combinazioni logiche di bit → SR →)

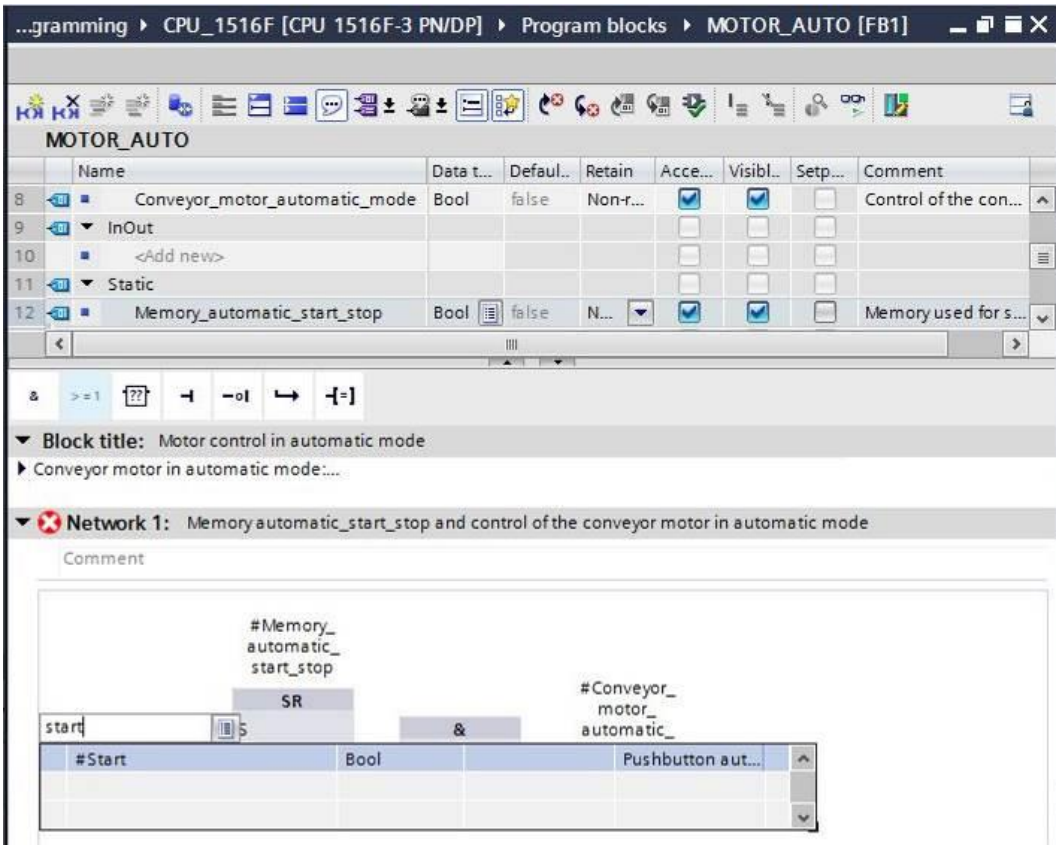


→ Per il flipflop di impostazione/reset è necessaria una variabile di memoria. Trascinare il parametro Static #Memory_automatic_start_stop su <??.?> sopra il flipflop SR. (→ Memoria_automatica_Start_Stop)



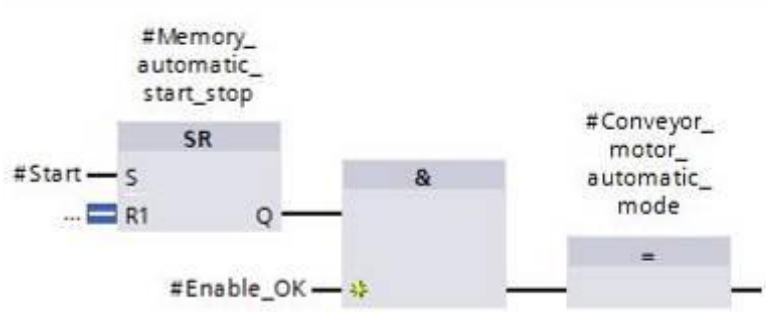
→ Il parametro #Memory_automatic_start_stop deve essere impostato con la variabile di ingresso #Start. Fare doppio clic sull'ingresso S del flipflop SR <??.?> e inserire "Start" nel campo che si apre in modo da visualizzare un elenco delle variabili disponibili che iniziano per "Start". Fare clic sulla variabile #Start e acquisirla con → Invio.

(→ SR-Flipflop → <??.?> → Start → #Start → Invio)

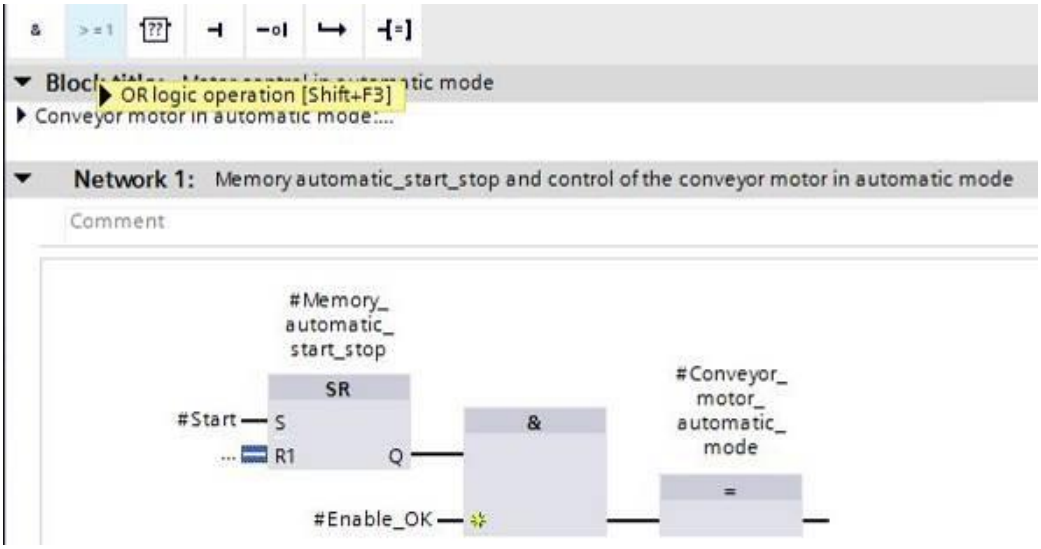


Nota: *con questa variante dell'assegnazione delle variabili esiste il rischio di uno scambio con le variabili globali della tabella delle variabili. Per questo motivo è preferibile scegliere la variante con drag&drop della descrizione dell'interfaccia.*

→ Diverse condizioni devono poter arrestare il nastro. Pertanto, nell'ingresso R1 del flipflop SR è necessario un blocco OR. Fare clic prima sull'ingresso R1 del flipflop SR in modo che il trattino dell'ingresso abbia lo sfondo blu.

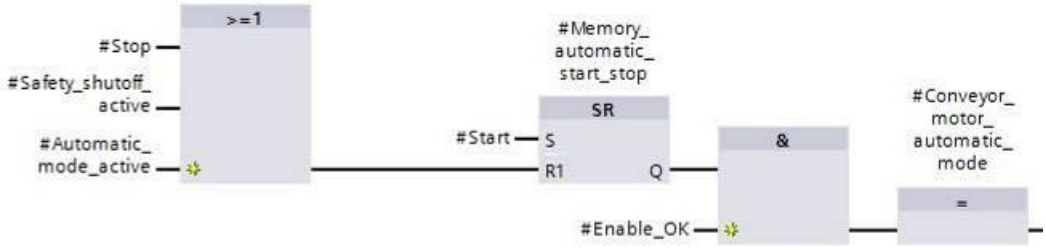


→ Fare clic sul simbolo >=1 nella barra dei simboli logici per inserire una combinazione logica OR.

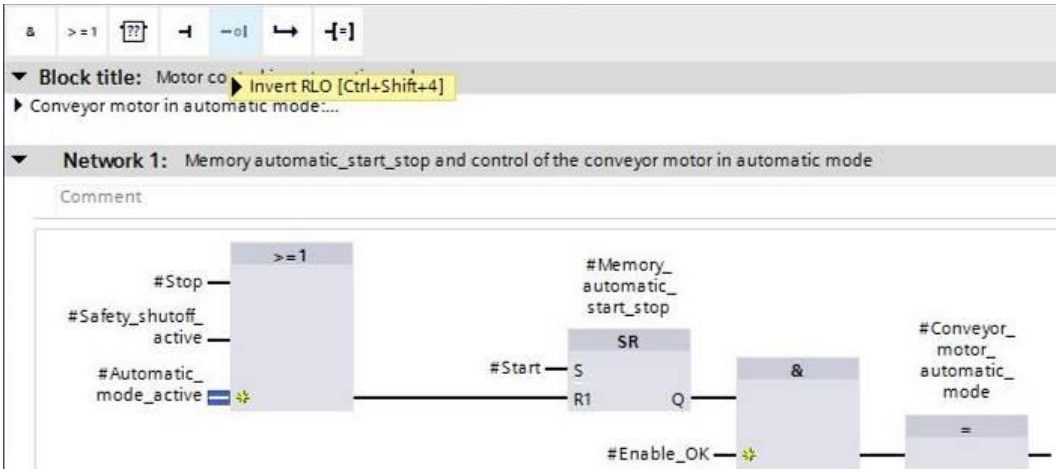


→ Il blocco OR ha dapprima solo 2 ingressi. Per poter collegare un'ulteriore variabile di ingresso fare clic sull'asterisco giallo dell'elemento OR.

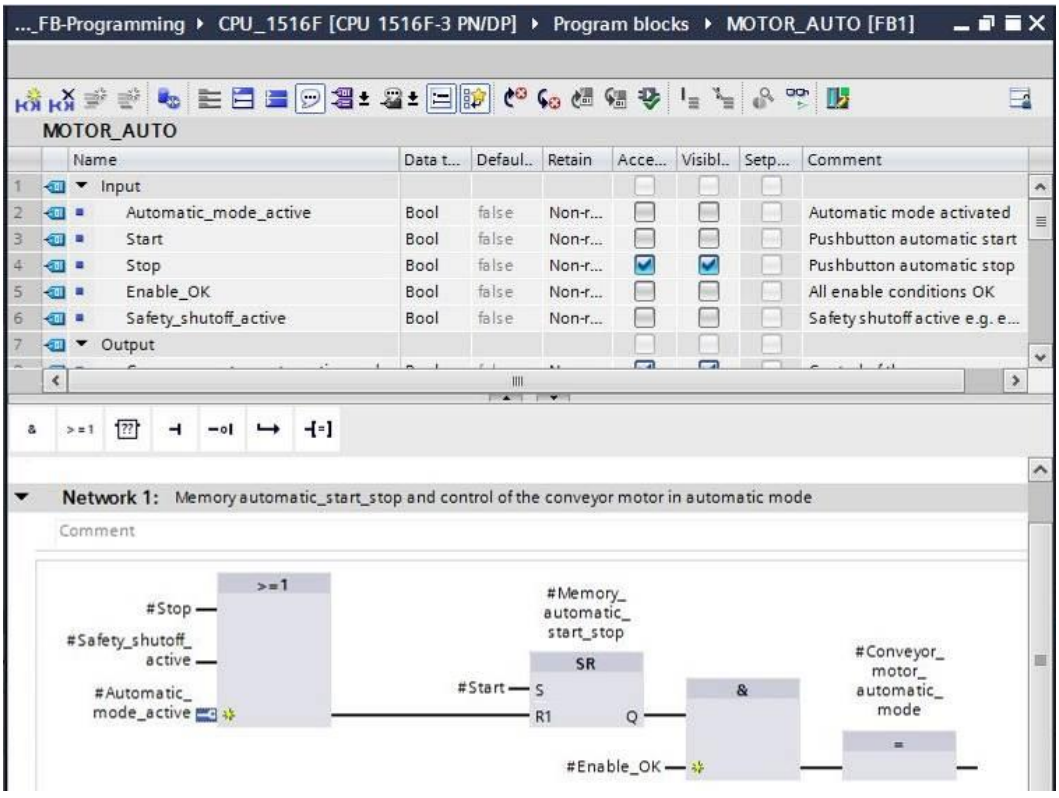
→ Aggiungere ai 3 ingressi dell'elemento OR le variabili di ingresso #Stop, #Safety_shutoff_active e #Automatic_mode_active.



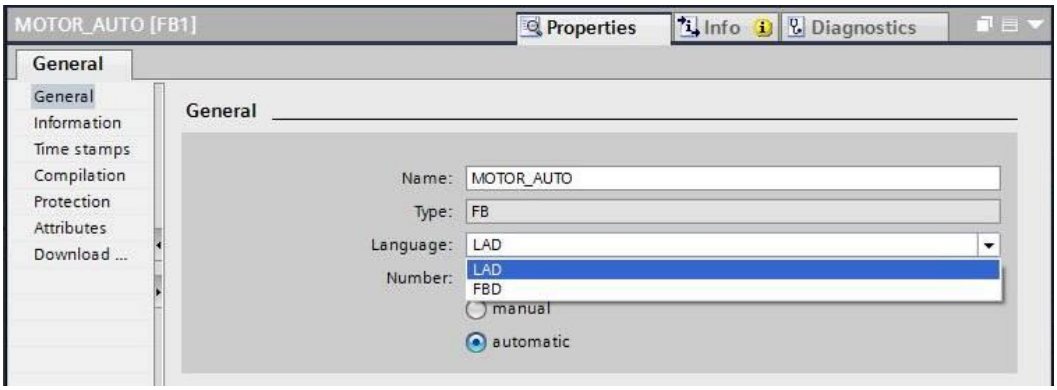
→ Negare l'ingresso collegato con il parametro #Automatic_mode_active selezionandolo e facendo clic su −o| .



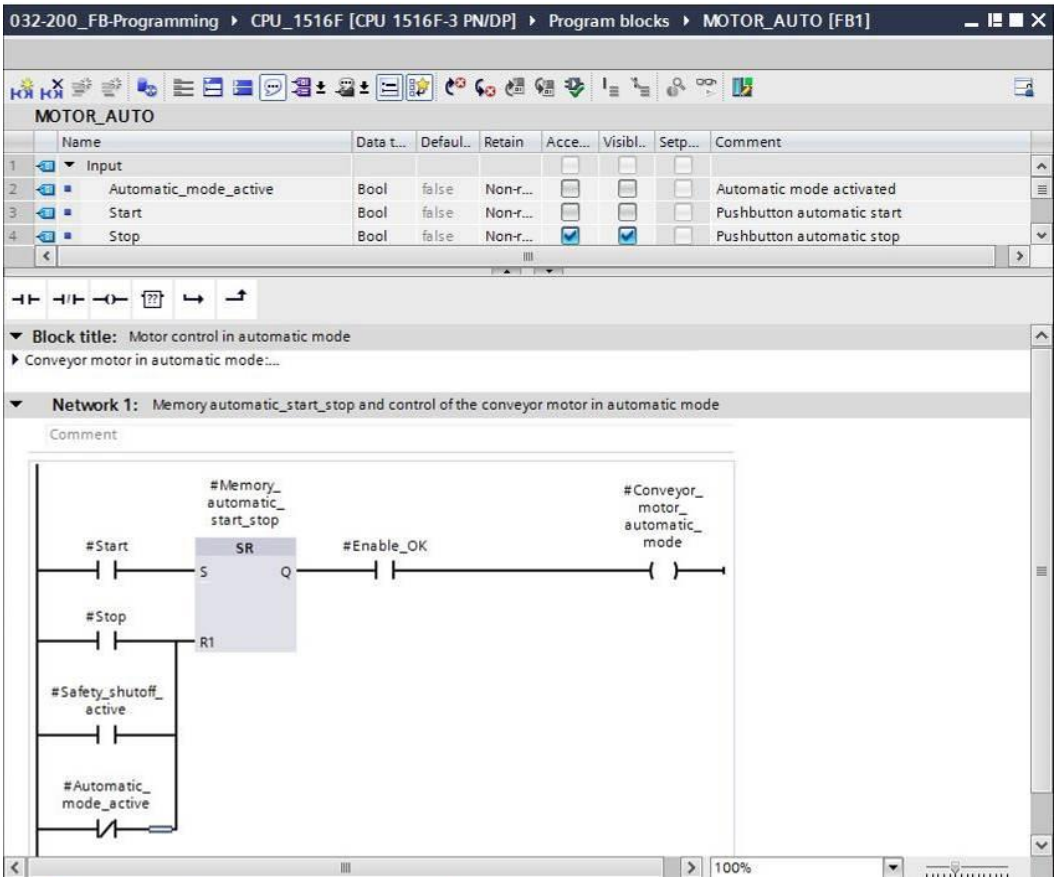
→ Non dimenticare di fare clic su Save project (Salva progetto). Il blocco funzionale finito "MOTOR_AUTO" [FB1] in FUP è rappresentato qui di seguito.



→ Nelle proprietà del blocco (“Properties”) è possibile aprire la scheda “General” e reimpostare “Language” su LAD (KOP, schema a contatti). (→ Proprietà → Generale → Linguaggio: KOP)



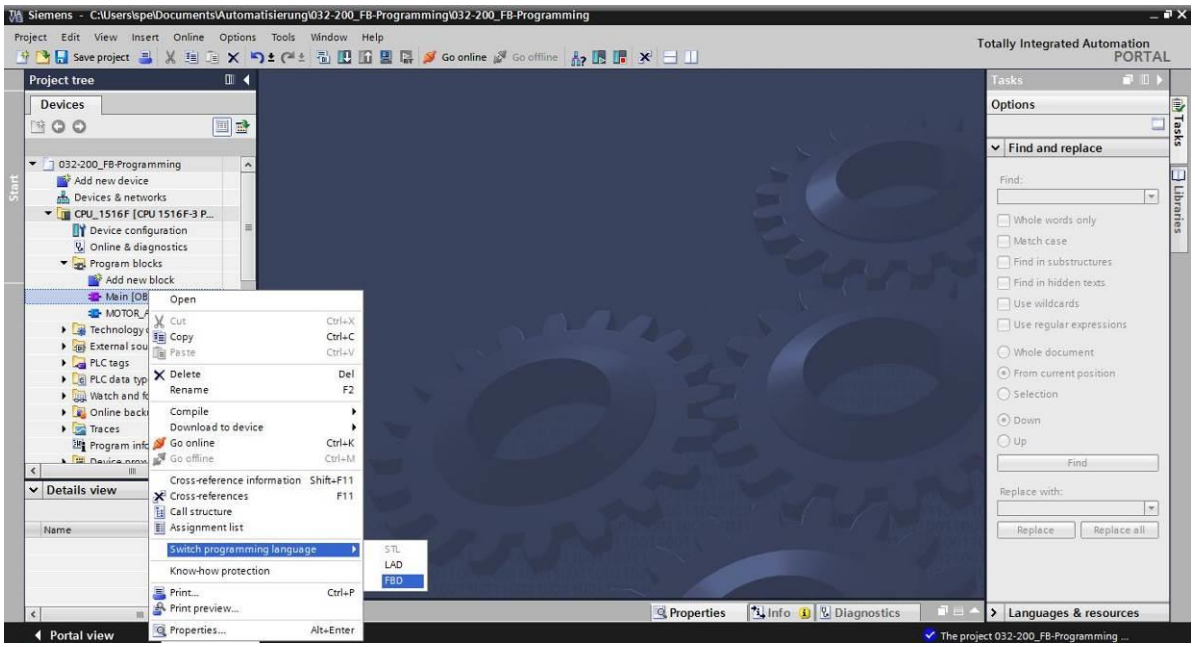
→ In KOP il programma viene visualizzato come segue.



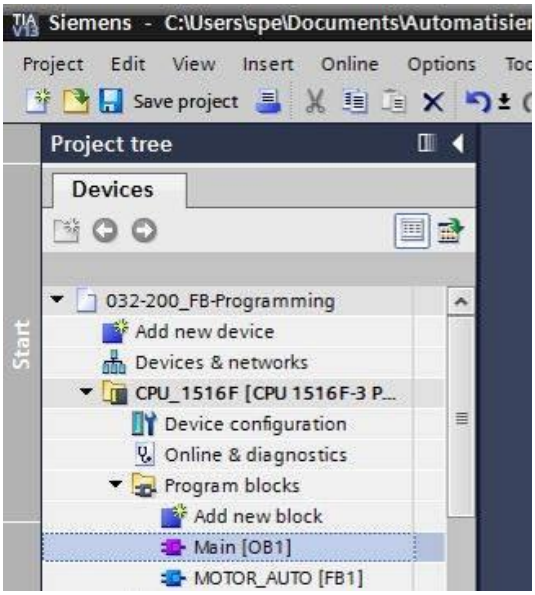
7.8 Programmazione del blocco organizzativo OB1 – comando del movimento del nastro in avanti in funzionamento automatico

→ Prima di programmare il blocco organizzativo “Main[OB1]” impostiamo il linguaggio di programmazione su FUP (schema logico). Prima fare clic con il tasto sinistro del mouse nella cartella “Program blocks” su “Main[OB1)”.

(→ CPU_1516F[CPU 1516F-3 PN/DP → Program blocks / Blocchi di programma → Main [OB1] → Switch program language / Commuta linguaggio di programmazione → FBD / FUP)

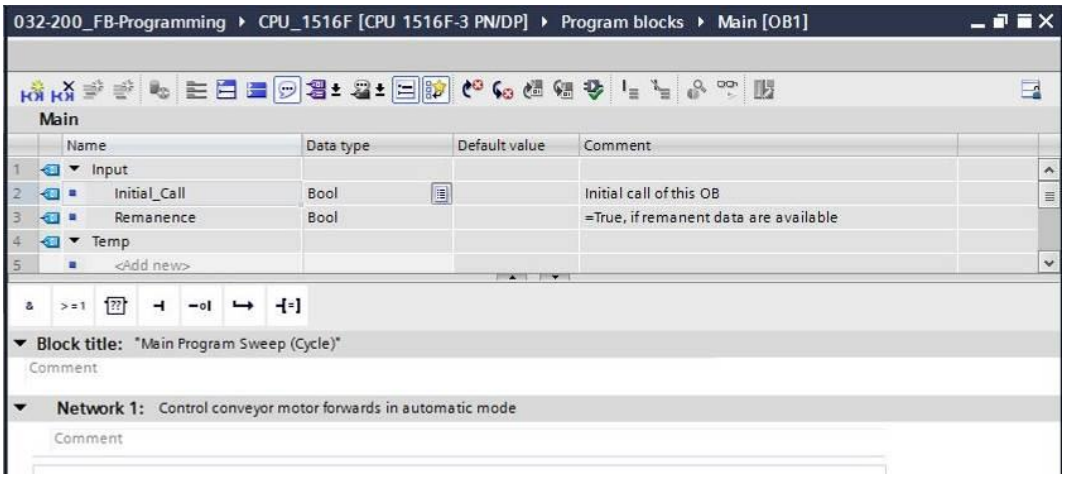


→ Aprire il blocco organizzativo “Main [OB1]” con un doppio clic.

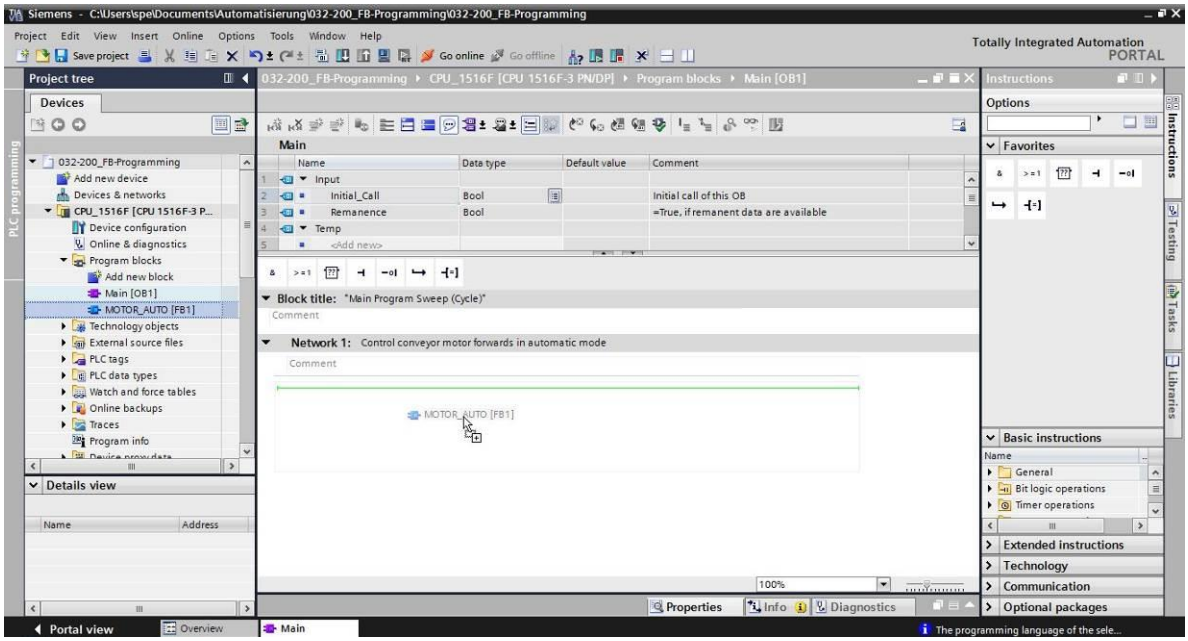


→ Nominare il segmento 1 “Control conveyor motor forwards in automatic mode”.

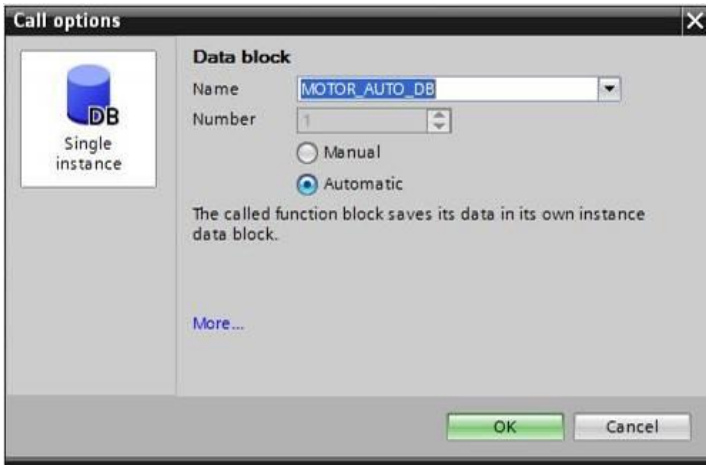
(→ Segmento 1:… → comando del movimento del nastro in avanti in funzionamento automatico)



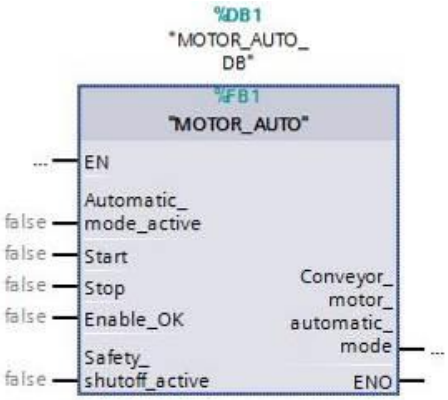
→ Trascinare il blocco funzionale “MOTOR_AUTO [FB1]” nel segmento 1 sulla linea verde.



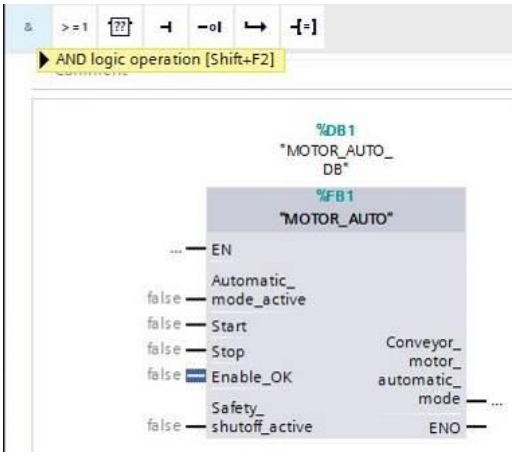
→ Il blocco dati di istanza per questo richiamo dell'FB1 viene generato automaticamente. Assegnare un nome e confermarlo con "OK". (→ MOTOR_AUTO_DB1 → OK)



→ Verrà inserito nel segmento 1 un blocco con l'interfaccia definita precedentemente, il blocco dati di istanza e le connessioni EN ed ENO.

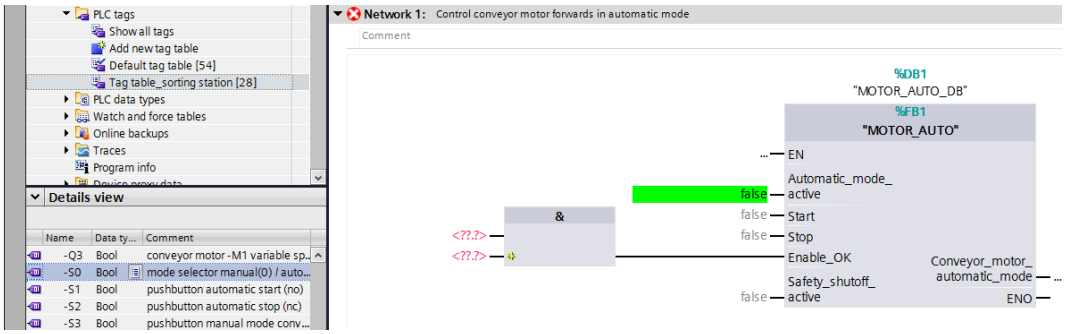


→ Per inserire un AND davanti al parametro di ingresso "Enable_OK" selezionare l'ingresso e inserire l'AND nella barra dei simboli logici con un clic sul simbolo & . (→ &)

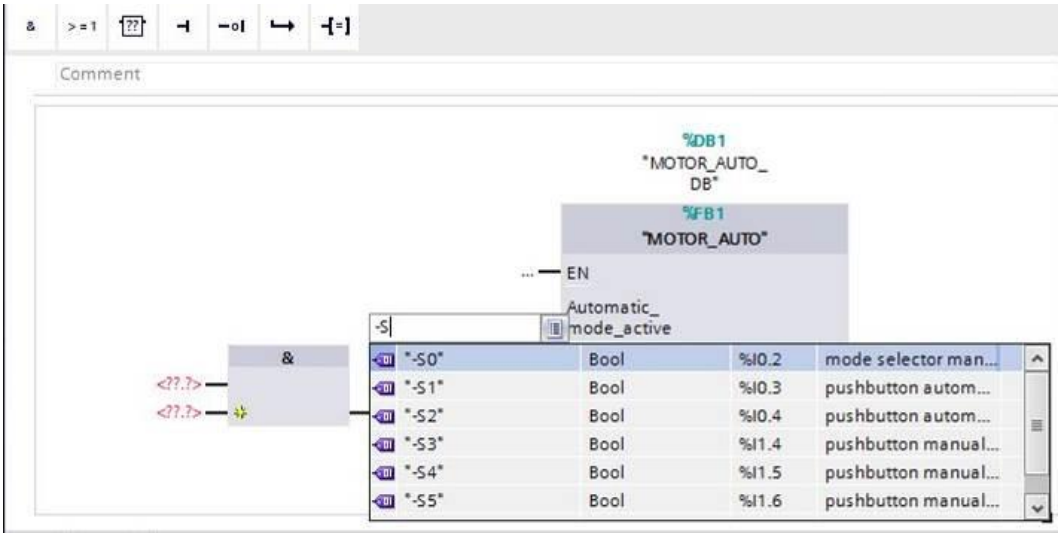


→ Per collegare il blocco con le variabili globali della tabella “Tag_table_sorting station” esistono 2 possibilità:

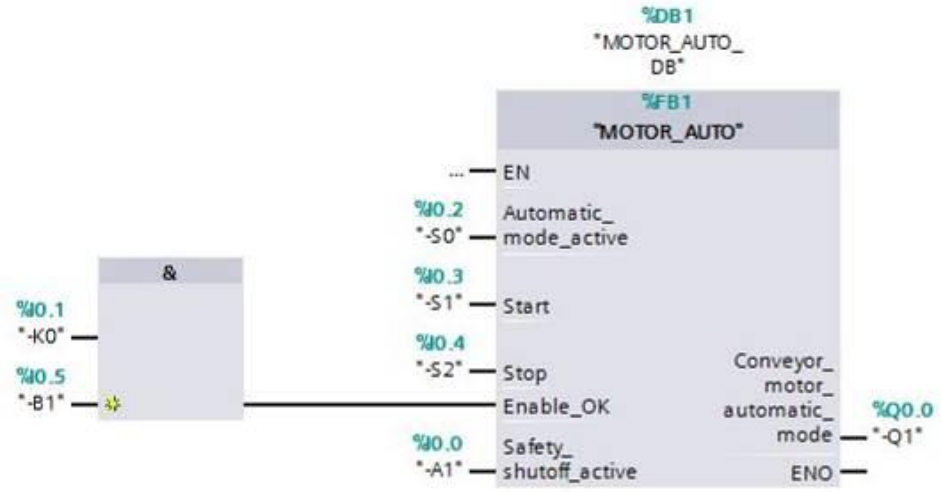
→ Selezionare la tabella “Tag_table_sorting station” nella navigazione del progetto e trascinare la variabile globale desiderata dalla vista “Details view” all’interfaccia della funzione FC1 (→ Tabella_variabili_stazione_smistamento → Vista dettagli → -S0 → Funzionamento_automatico_attivo)



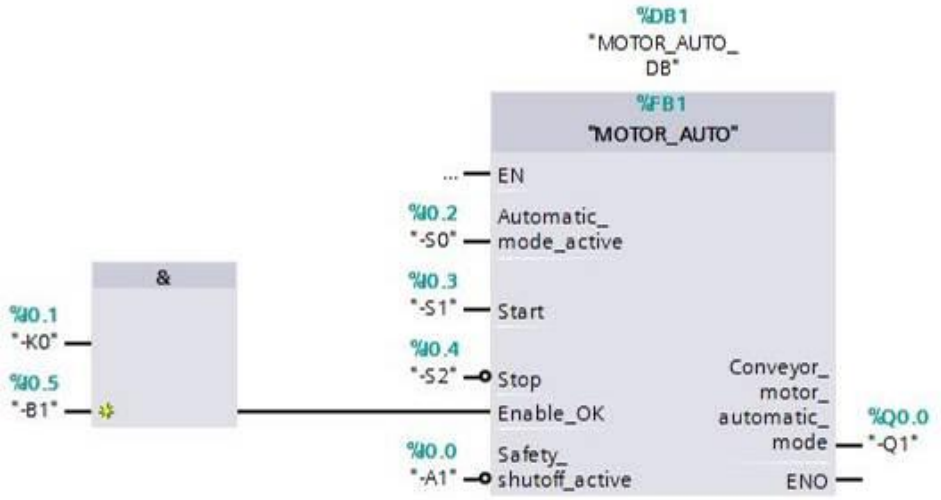
→ In alternativa inserire in <??.?> le lettere iniziali della variabile globale desiderata (ad es. “-S”) e selezionare dall’elenco visualizzato la variabile di ingresso globale “-S0” (%I0.2). (→ Funzionamento_automatico_attivo → -S → -S0)



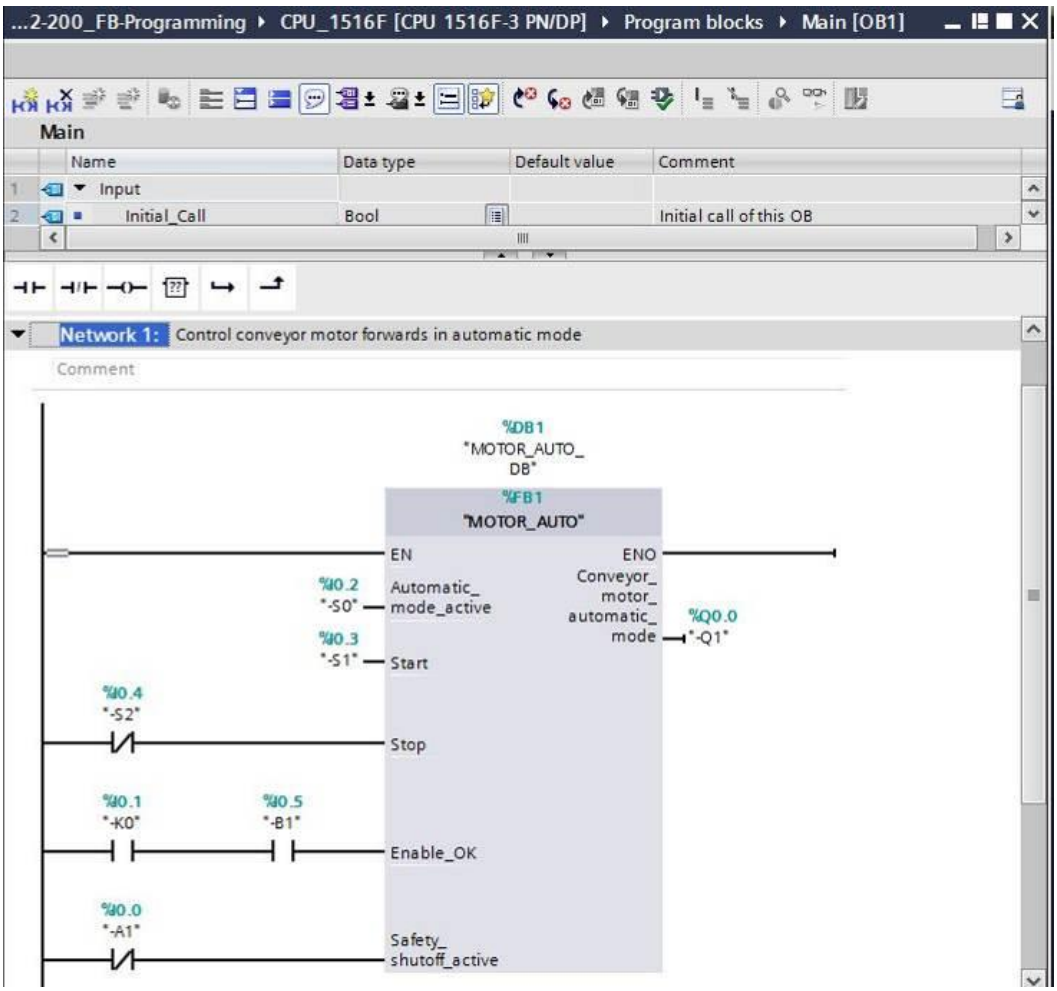
→ Inserire le ulteriori variabili di ingresso “-S1”, “-S2”, “-K0”, “-B1” e “-A1” e immettere la variabile di uscita “-Q1” (%Q0.0) nell’uscita “Conveyor_motor_automatic_mode”.



→ Negare le interrogazioni delle variabili di ingresso “-S2” e “-A1” selezionandole e facendo clic su  . (→ -S2 →  → -A1 → )

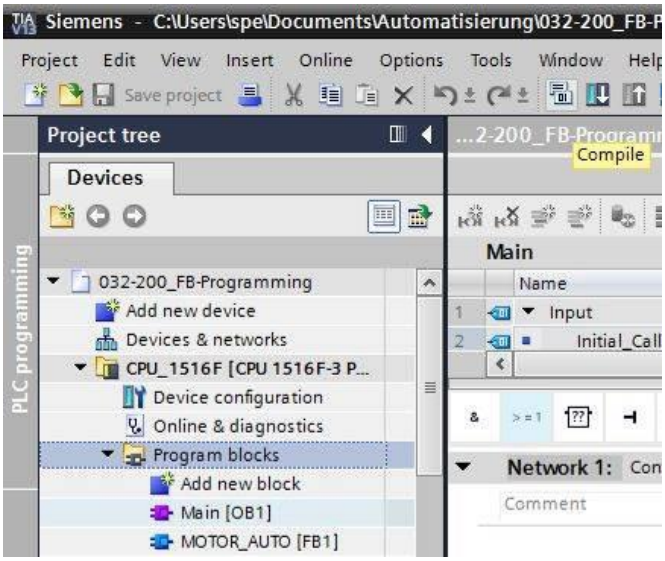


7.9 Nel linguaggio di programmazione KOP (schema a contatti) il risultato compare come segue.

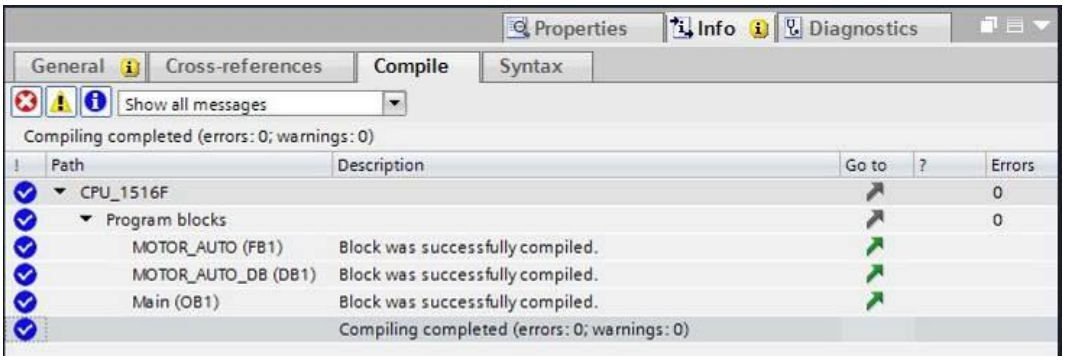


7.10 Salvataggio e compilazione del programma

→ Per salvare il progetto selezionare nel menu il pulsante Save project (Salva progetto). Per compilare tutti i blocchi fare clic sulla cartella “Program blocks” e selezionare nel menu il simbolo per la compilazione. (→ Save project → Blocchi di programma →)

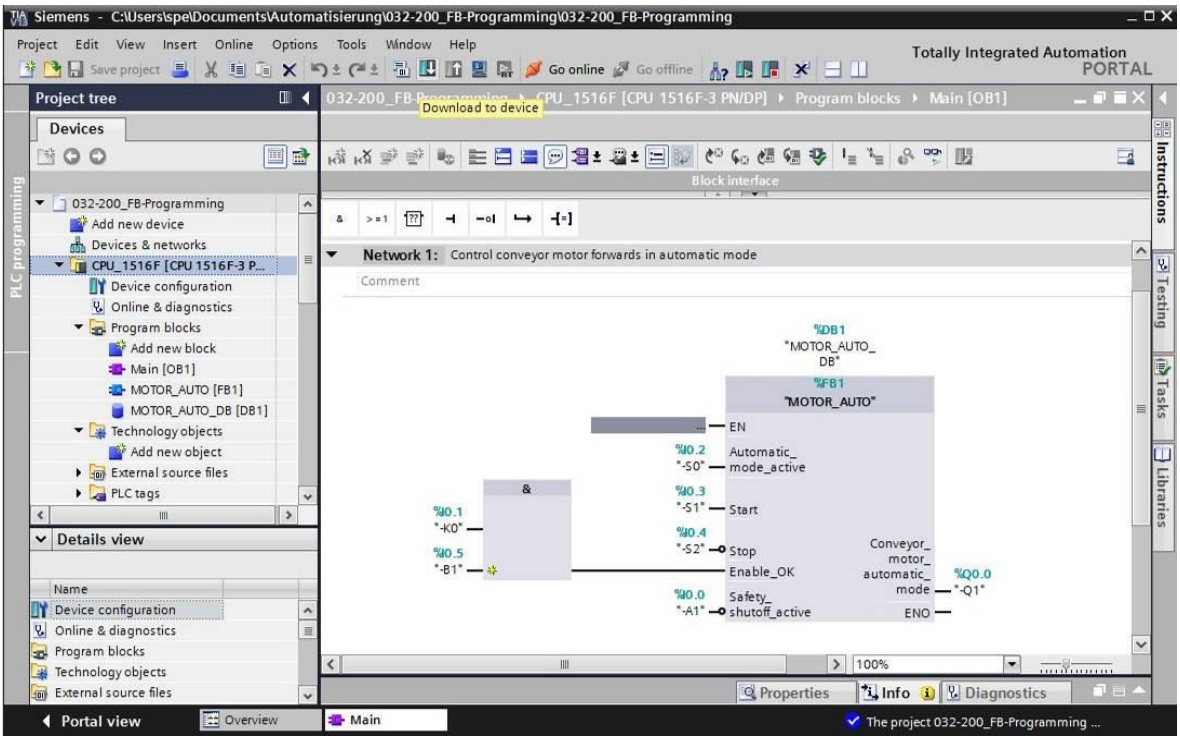


→ Nell’area “Info” “Compile” (Informazioni / Compila) è possibile vedere quali blocchi sono stati compilati senza errori.



7.11 Caricamento del programma

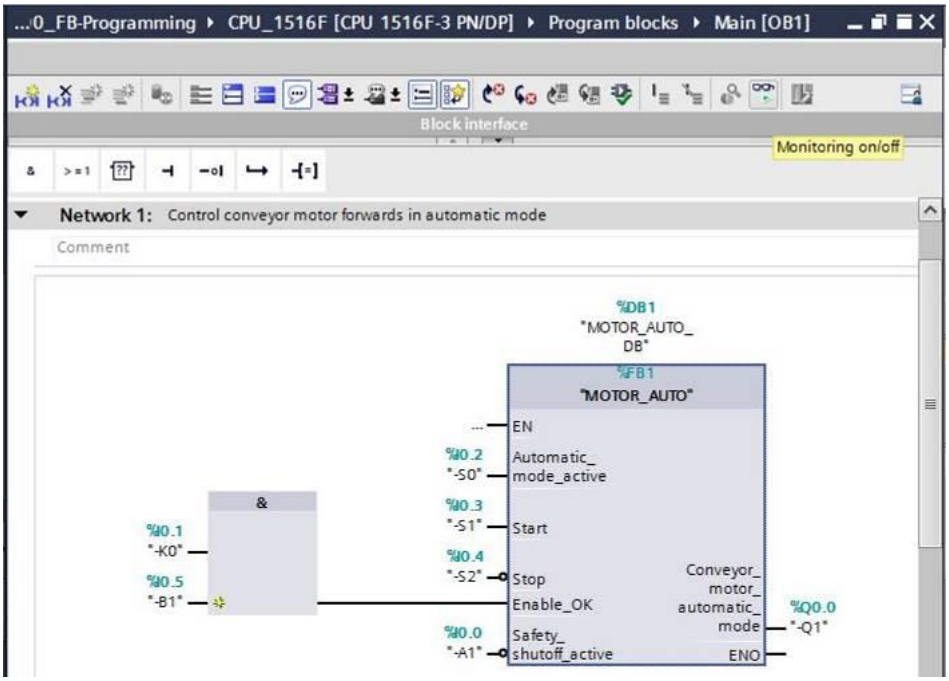
→ Al termine della compilazione è possibile caricare l'intero controllore con il programma creato come descritto nei moduli sulla configurazione hardware. (→)

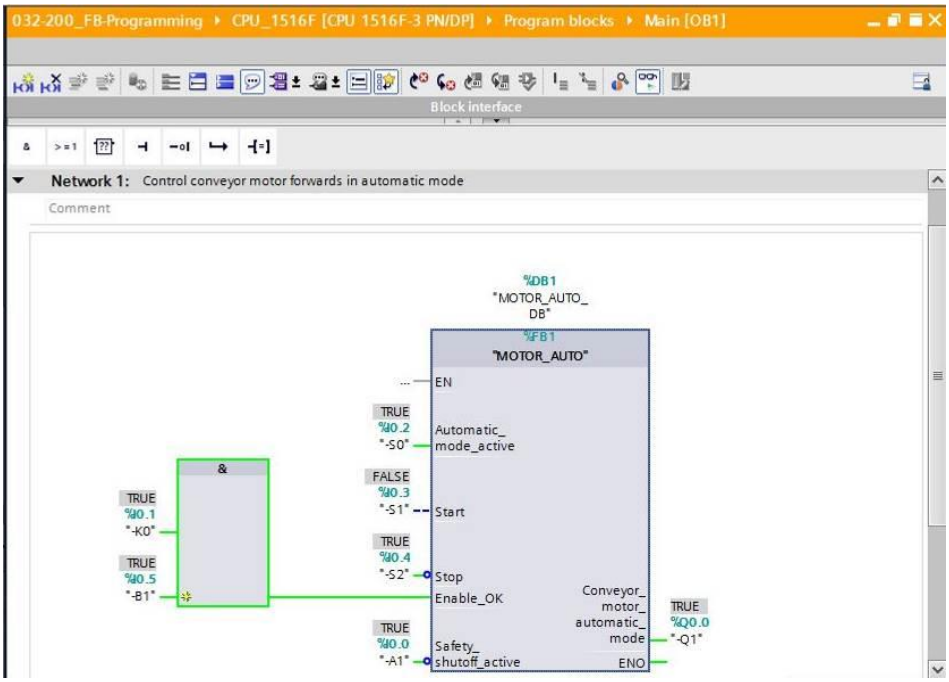


7.12 Controllo dei blocchi di programma

→ Per controllare il programma creato è necessario che il blocco corrispondente sia aperto.

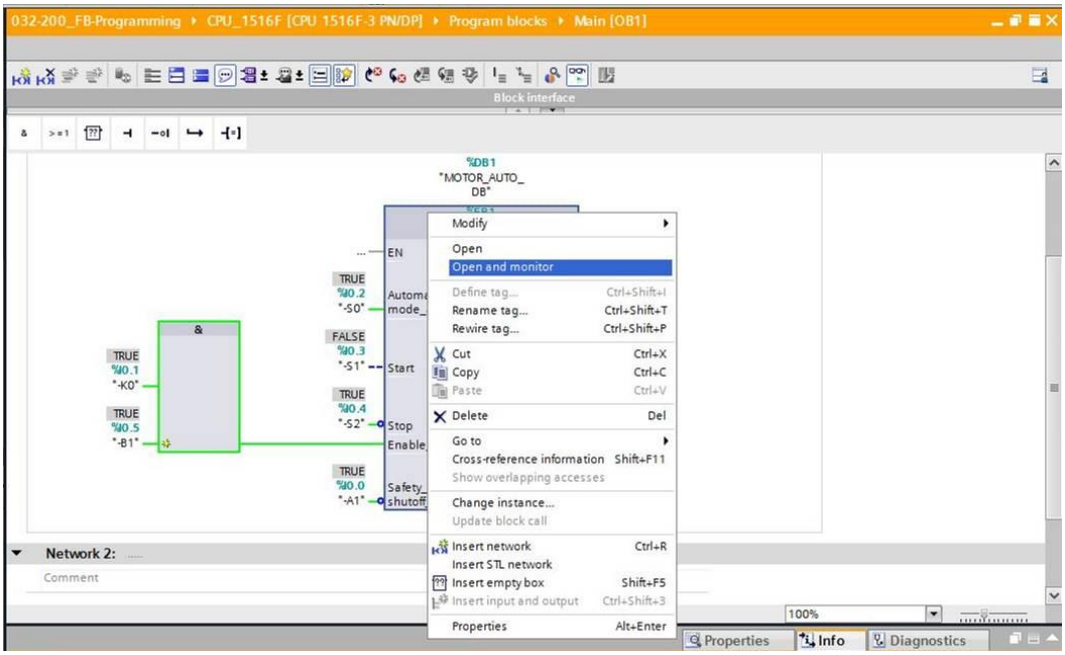
Con un clic sul simbolo  è possibile attivare/disattivare il controllo. (→ Main [OB1] → )

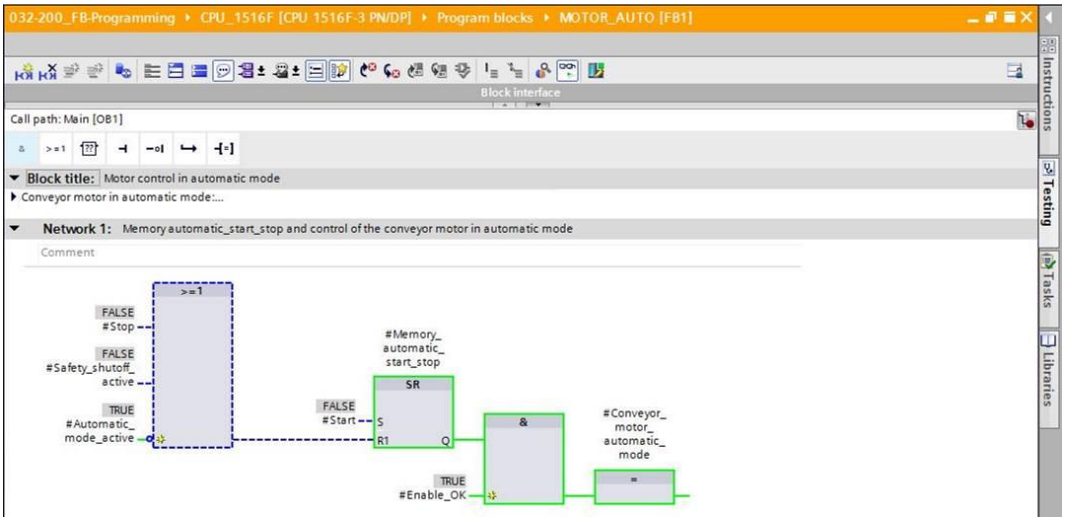




Nota: *il controllo qui avviene in riferimento al segnale e in funzione del controllore. Gli stati del segnale nei morsetti vengono visualizzati con TRUE o FALSE.*

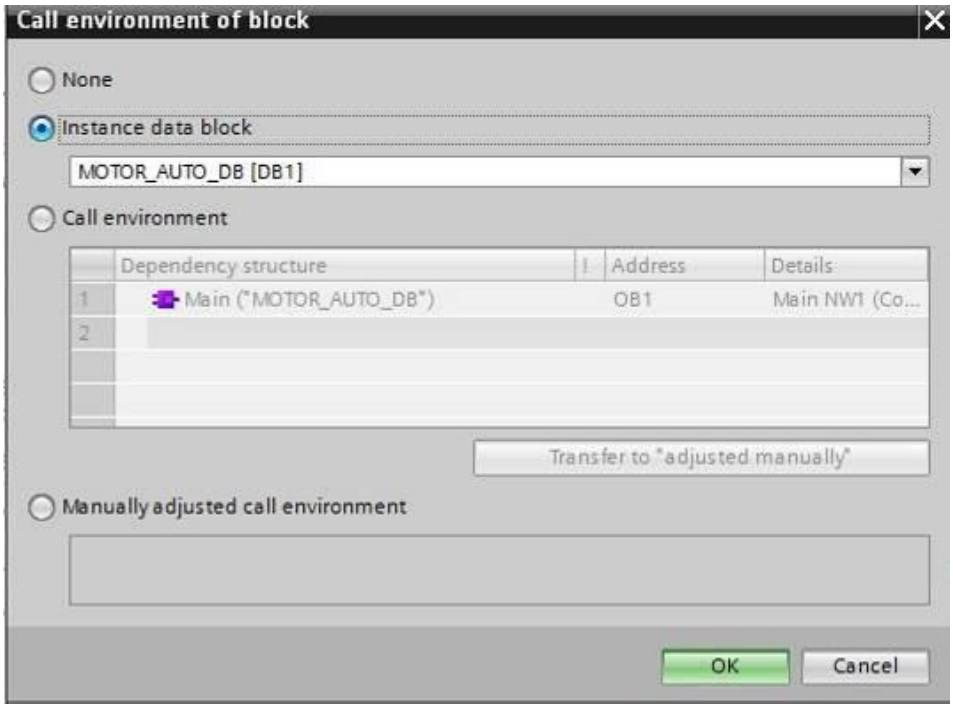
→ Il blocco funzionale “MOTOR_AUTO” [FB1] richiamato nel blocco organizzativo “Main [OB1]” si può aprire direttamente facendo clic con il tasto destro del mouse su “Open and monitor”. (→ “MOTOR_AUTO” [FB1] → Apri e controlla)

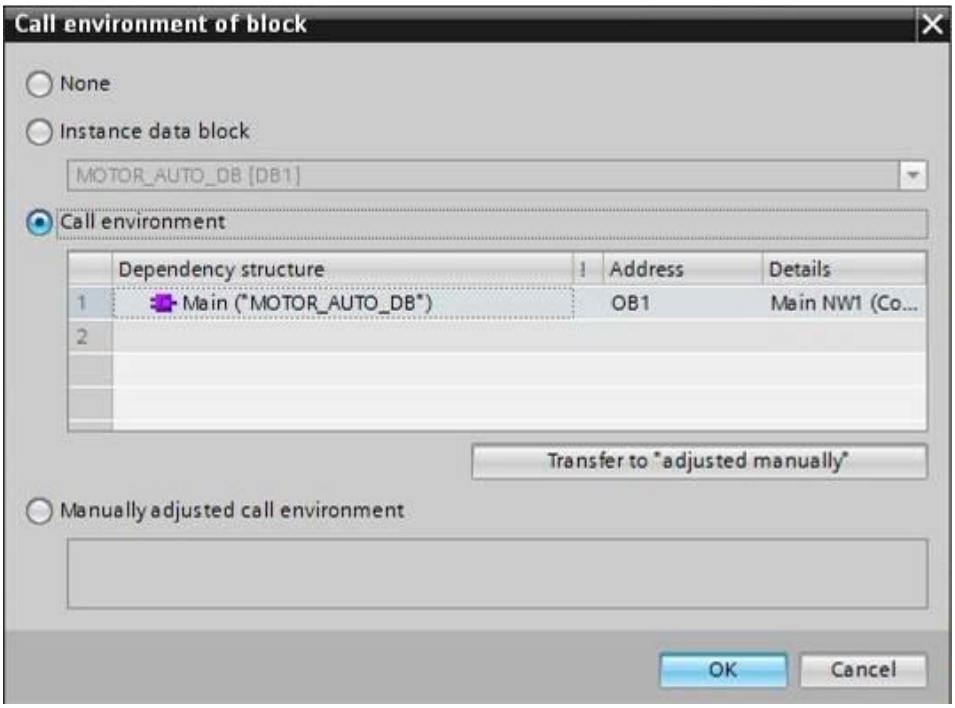




Nota: *il controllo qui avviene in riferimento alla funzione e indipendentemente dal controllore. L’azionamento degli encoder o lo stato dell’impianto vengono rappresentati con TRUE o FALSE.*

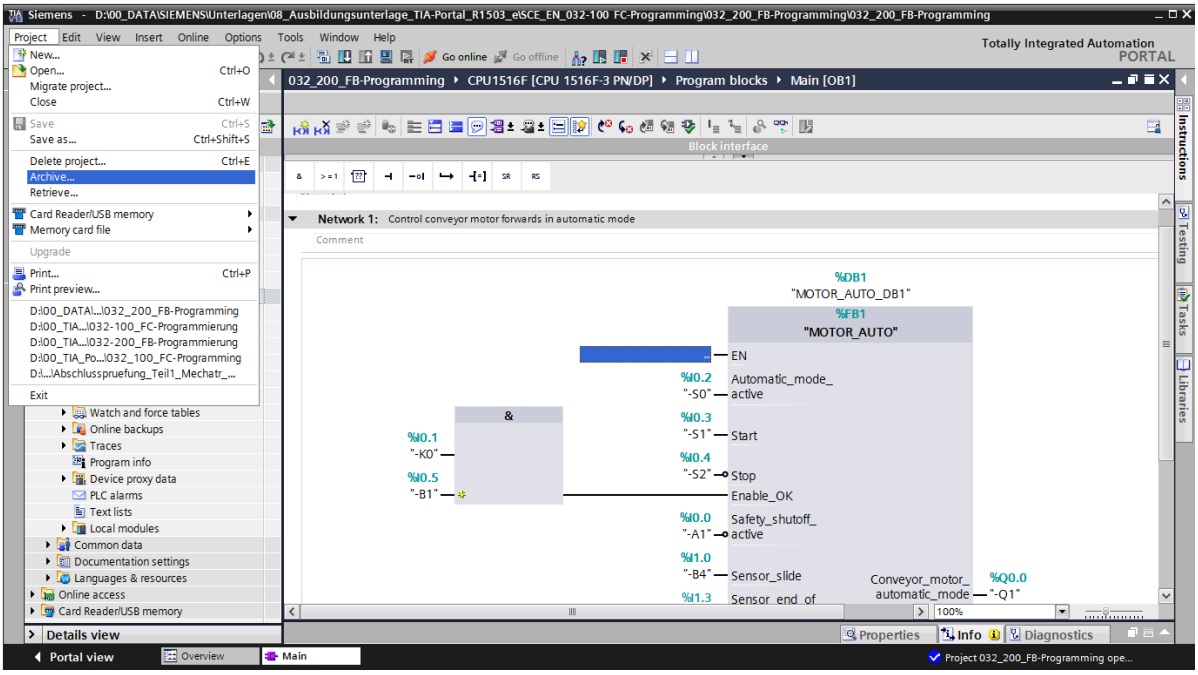
→ Per controllare un determinato punto di applicazione di un blocco funzionale “MOTOR_AUTO” [FB1] richiamato più volte, è possibile utilizzare il simbolo  . Qui esistono due alternative per definire l’ambiente di richiamo: o richiamando l’ambiente stesso o attraverso il blocco dati di istanza. (→  → Instance data block / Blocco dati di istanza → MOTOR_AUTO_DB1 [DB1] → Call environment / Ambiente di richiamo → Address / Indirizzo: OB1 → Details / Dettagli: Main NW1 → OK)





7.13 Archiviazione del progetto

→ Per concludere, vogliamo archiviare il progetto completo. Selezionare nel menu → “Project” il comando → “Archive…”. Selezionare una cartella in cui archiviare il progetto e salvare come “TIA Portal project archives”. (→ Progetto → Archivia → Archivi di progetto del TIA Portal → 032-200_Programmazione di FB... → Salva)



8 Lista di controllo

N.	Descrizione	Controllato
1	Compilazione riuscita senza messaggi di errore	
2	Caricamento riuscito senza messaggi di errore	
3	Accensione impianto (-K0 = 1) Cilindro inserito / conferma attivata (-B1 = 1) Arresto d'emergenza (-A1 = 1) non attivato Modo di funzionamento AUTOMATICO (-S0 = 1) Tasto di arresto automatico non azionato (-S2 = 1) Azionare brevemente il tasto di avvio automatico (-S1 = 1) in seguito si attiva il motore del nastro in avanti numero di giri fisso (-Q1 = 1) e rimane "ON".	
4	Azionare brevemente il tasto di arresto automatico (-S2 = 0) $\rightarrow$ -Q1 = 0	
5	Attivazione dell'arresto d'emergenza (-A1 = 0) $\rightarrow$ -Q1 = 0	
6	Modo di funzionamento manuale (-S0 = 0) $\rightarrow$ -Q1 = 0	
7	Spegnimento impianto (-K0 = 0) $\rightarrow$ -Q1 = 0	
8	Cilindro non inserito (-B1 = 0) $\rightarrow$ -Q1 = 0	
9	Progetto archiviato correttamente	

9 Esercitazione

9.1 Definizione del compito – esercitazione

L'obiettivo di questa esercitazione è di aggiungere al blocco funzionale MOTOR_AUTO [FB1] una funzione di risparmio energetico. Il blocco funzionale così ampliato deve essere pianificato, programmato e testato:

Per risparmiare energia il nastro deve muoversi solo quando è presente un pezzo.

L'uscita Motore_automatico pertanto viene comandata solo se è impostata la Memoria_automatica_Start_Stop, se sono soddisfatte le condizioni di abilitazione e se è impostato Memoria_nastro_Start_Stop.

Il parametro Memoria_nastro_Start_Stop viene impostato se il Sensore_scivolo_occupato segnala la presenza di un pezzo e viene resettato se il Sensore_fine_nastro genera un fronte di discesa o se è attiva la disinserzione di protezione o se non è attivo il funzionamento automatico (funzionamento manuale).

9.2 Pianificazione

Pianificare ora in autonomia la realizzazione del compito.

Nota: *consultare la Guida in linea per informazioni sull'utilizzo del fronte di discesa in SIMATIC S7-1500.*

9.3 Lista di controllo – esercitazione

N.	Descrizione	Controllato
1	Compilazione riuscita senza messaggi di errore	
2	Caricamento riuscito senza messaggi di errore	
3	Accensione impianto (-K0 = 1) Cilindro inserito / conferma attivata (-B1 = 1) Arresto d'emergenza (-A1 = 1) non attivato Modo di funzionamento AUTOMATICO (-S0 = 1) Tasto di arresto automatico non azionato (-S2 = 1) Azionare brevemente il tasto di avvio automatico (-S1 = 1) Sensore scivolo occupato attivato (-B4 = 1) in seguito si attiva il motore del nastro in avanti numero di giri fisso (-Q1 = 1) e rimane "ON".	
4	Sensore fine nastro attivato (-B7 = 1) → -Q1 = 0	
5	Azionare brevemente il tasto di arresto automatico (-S2 = 0) → -Q1 = 0	
6	Attivazione dell'arresto d'emergenza (-A1 = 0) → -Q1 = 0	
7	Modo di funzionamento manuale (-S0 = 0) → -Q1 = 0	
8	Spegnimento impianto (-K0 = 0) → -Q1 = 0	
9	Cilindro non inserito (-B1 = 0) → -Q1 = 0	
10	Progetto archiviato correttamente	

10 Ulteriori informazioni

Per l'apprendimento o l'approfondimento sono disponibili ulteriori informazioni di orientamento, come ad es.: Getting Started, video, tutorial, App, manuali, guide alla programmazione e Trial software/firmware al link seguente:

www.siemens.com/sce/s7-1500