TIA Portal – Programmazione

Rev Digitale 1.1 del 01/07/2020

Per modificare il nome del progetto è sufficiente modificare il nome del file con estensione **.ap15**

Introduzione

TIA Portal è un **ambiente integrato** (nato nel 2010) per programmare tutti i dispositivi di automazione: PLC, motion, hmi, sensori intelligenti. Database unico e dunque più facilmente gestibile ed interrogabile.
TIA Portal è un **ambiente modulare** che consente di installare i moduli che servono. Prima di TIA Portal esistevano più applicazioni differenti, una per i plc 200, una per i plc 300(STEP 7), una per gli hmi (WINCC), una per gli azionamenti dei motori (passo passo, brushless), etc.
Ogni modulo ha una sua licenza.

L'applicazione step7 è quella che consente di programmare i PLC. La programmazione è la stessa sia per PLC 1200 che per i PLC 1500

L'applicazione **STEP 7 basic** consente di programmare soltanto la famiglia dei PLC 1200.
L'applicazione **STEP 7 professional** consente invece di programmare sia i plc 1200 sia i plc 1500.
Una cosa analoga vale per gli **hmi** dove ci sono le versioni basic, advanced e comfot

Dall'ultimo menù **HELP / Software installato** si vede l'elenco delle applicazioni installate
Eventualmente cliccare sull'ultimo pulsante "Ulteriori proprietà"

Tool esterni a TIA Portal

- Licence Manager
- PLC SIM
- Update extender

Le viste

TIA portal si apre "**vuoto**", senza alcun componente o progetto.

L'ambiente dispone di due viste differenti:

- **`Vista Portale`** (apertura di default)
- **`Vista Progetto`**

Tutti i programmatori prediligono la vista progetto
E' possibile impostare la Vista Progetto come vista di apertura agendo sul menù
`Strumenti / Impostazioni / Generale`

Linguaggi e strutture dati

Il linguaggio

Un programma PLC può utilizzare diversi linguaggi

- **KOP** ad oggi il più utilizzato. Strutturato a ladder usa le convenzioni dei circuiti elettrici
- **FUP** (a blocchi) ormai in declino
- **SCL** (Structured Control Language) E' un Linguaggio testuale strutturato in cui sono disponibili le tipiche istruzioni informatiche: IF; FOR, WHILE, DO WHILE, SWITCH. Ottimo per elaborazioni più complesse come cicli, calcoli matematici, elaborazione di stringhe. Sempre più utilizzato ma non ancora diffuso come il ladder

Nei PLC 1500 sono disponibili due linguaggi aggiuntivi:
- **AWL** che è uno pseudoassembler storico di Siemens oggi quasi inutilizzato
- **GRAPH**

Un **programma ladder** è strutturato in righe dette **segmenti**.
In TIA Portal si possono anche inserire più righe all'interno di uno stesso segmento
All'interno del ladder è possibile utilizzare una struttura mista in cui si possono scrivere ad esempio alcuni segmenti in KOP ed altri in SCL

Al momento del download Il **codice** viene memorizzato in forma testuale all'interno della cosiddetta "Memoria di caricamento" costituita da una Memory card, in modo da poter essere eventualmente letto in qualsiasi momento da un PC.
Nel momento in cui il PLC passa dallo stato di STOP allo stato di RUN il programma viene automaticamente compilato e salvato nella memoria di lavoro (RAM)

Le variabili

Le variabili possono essere memorizzate all'interno di diverse aree di memoria:

- **Immagine degli ingressi**
- **Immagine delle uscite**
- **merker**
- **DB (data block)**

Tutte le aree di memoria, nei PLC Siemens, sono sempre **organizzate in bytes** numerati in modo crescente a partire dal byte 0, cioè **l'offset è SEMPRE espresso in BYTE rispetto al Byte 0.**
M13 indica il byte 13 (quattordicesimo) all'interno dell'area di memoria dei merker

Immagine di ingressi e uscite

E' l'area in cui viene copiato il valore degli ingressi prima dell'esecuzione e programma ed in cui il programma provvede a salvare temporaneamente il valore delle uscite.
Visibili da Configurazione Hardware / Modulo desiderato / Proprietà / IO

I = ingressi	da **I0.0** fino a **I3.7**
Q = uscite	da **Q0.0** fino a **Q3.7**

Come per tutte le aree di memoria, anche l'indirizzamento di ingressi e uscite è riferito al byte (partendo da 0). Cioè **l'offset è SEMPRE espresso in BYTE rispetto al Byte 0**
I0.5 significa byte 0, quinto bit (cioè il sesto ingresso)
I3.7 significa byte 3, settimo bit (cioè il 32° ingresso)

Merker

L'area relativa ai **merker** è un'area che esiste storicamente fin dai primi plc Siemens.
- E' un'area unica di ampiezza fissa **NON ORGANIZZABILE**.
  Nei PLC 1500 sono numerati da 0 a 8000.
- Si tratta di un'area globale visibile da tutti i blocchi di codice.
- Nel momento del passaggio STOP / RUN il valore dei merker viene azzerato
- Le variabili definite all'interno dei merker possono essere ritentive / non ritentive. (ritentive significa che mantengono il valore tra una variazione STOP RUN del PLC). Non è possibile però rendere ritentiva una singola variabile dei merker, ma soltanto una certa area iniziale, cioè ad esempio tutti i merker fino al merker 30. Pulsantino in alto nell'area dei merker

L'utilizzo dei merker è ampiamente sconsigliato (anche da Siemens stesso) che ne limita l'uso a variabili di sistema o temporanee

DB

L'area relativa alle DB è un'area molto più ampia rispetto ai merker (da 1 M a 20MB).

- Quest'area è **organizzabile** in più DB con nome e scopo diversi.
- Come per i merker si tratta di un'area globale visibile da tutti i blocchi di codice.
- La ritenzione può essere applicata alle singole variabili
- Riguardo al passaggio STOP / RUN, le variabili DB hanno una colonna START VALUE. che viene automaticamente caricato all'interno della variabile al momento del passaggio STOP / RUN

E' obbligatorio assegnare un nome ad ogni DB
Ogni singola DB può avere una sua struttura dati

Editing delle DB

- Ogni variabile interna alla DB deve avere un **nome** un **tipo** ed eventualmente uno **start value**
- Le variabili all'interno della DB devono essere dichiarate in modo continuativo con l'indirizzo assegnato automaticamente in modo sequenziale. Se voglio utilizzare il byte 20, devo prima definire con un apposito nome tutte le variabili precedenti.
- Quando si è online con modalità **test** attiva (occhialini). Viene mostrata una colonna aggiuntiva che visualizza il valore corrente di tutte le variabili della DB.
  Con doppio click è possibile anche modificare il valore della variabile.

Colonne aggiuntive

DB_stati

	Nome	Tipo di dati	Offset	Valore di avvio	A ritenzione	Accessibile da HMI/OPC UA	Scrivibile da HMI/OPC UA	Visibile in HMI Engineering	Valore di impostazione
1	Static								
2	sAuto	Bool	0.0	false	☐	☑	☐	☑	☐
3	sStop	Bool	0.1	false	☐	☑	☐	☑	☐
4	sMan	Bool	0.2	false	☐	☑	☐	☑	☐
5	sHome	Bool	0.3	false	☐	☑	☐	☑	☐
6	sAttesaStop	Bool	0.4	false	☐	☐	☐	☐	☐
7	sEmergenza	Bool	0.5	false	☐	☐	☐	☐	☐

A ritenzione : In corrispondenza di uno STOP / RUN mantengono il valore precedente, altrimenti assumono lo start value.

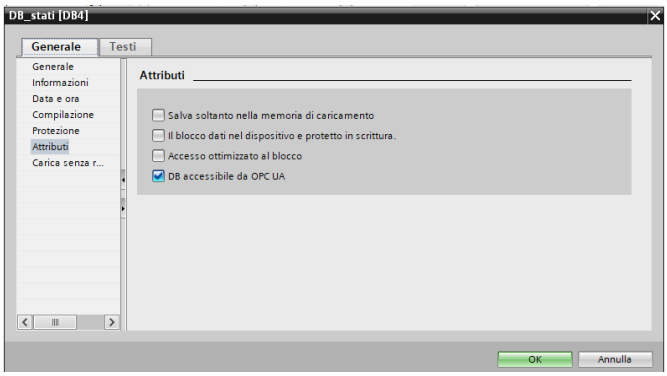
Valore di impostazione : Valore assunto dalle variabili non ritentive in caso di STOP / RUN

Accessibile da HMI/OPC : Variabile accessibile in lettura da un OPC-UA client

Scrivibile da HMI/OPC : Variabile accessibile in scrittura da un OPC-UA client

Visibilità della DB da un OPC-UA client

Facendo tasto destro su una DB, all'interno delle **Proprietà / Attributi** c'è una spunta "**DB accessibile da OPC UA**" senza la quale nessuna variabile della DB sarà visibile da OPC-UA.



Per consentire la comunicazione tramite le DLL microsoft occorre ancora selezionare la CPU / **Protezione e Security** / Livello di Accesso :

- Selezionare la prima voce **Full Access**
- Connection Mecanism: permit PUT/GET

La notazione simbolica

In TIA Portal è consigliata una programmazione attraverso una notazione **simbolica**
Ad ogni variabile si può assegnare un nome detto **simbolo**
E' raccomandato definire sempre le variabili prima del loro utilizzo
L'accesso alle variabili deve essere eseguito tramite simbolo e non tramite indirizzo assoluto.
Gli indirizzi fisici sono preceduti da un **%** davanti che fa capire al compilatore che si tratta di un indirizzo fisico e non di una variabile.

I vantaggio della notazione simbolica sono:

- La leggibilità
- il fatto che se vado a cambiare l'indirizzo fisico della variabile, automaticamente l'indirizzo verrà modificato all'interno dell'intero programma
- Per assegnare il simbolo ad un **merker** occorre apre la finestra "Variabili PLC", creare una nuova "**tabella delle variabili"** (nella versione inglese "PLC Tags") chiamata ad esempio **merker,** ed inserire all'interno tutte le variabili merker che si intendono utilizzare
- Per assegnare un simbolo ad una variabile di una **DB** è sufficiente definirlo all'interno della DB in fase di creazione
- Per assegnare un simbolo ad un **Input o Output** lo si può fare all'interno della "**tabella delle variabili**" oppure meglio ancora selezionando il modulo sulla "**configurazione hardware"** e poi sulle proprietà. La mappatura di ingressi e uscite costituisce in genere il primo step nella stesura del codice di una applicazione PLC

Laddove non c'è un simbolo assegnato ad un certo indirizzo, la prima volta che utilizzo quell'indirizzo, automaticamente TIA gli assegna un simbolo del tipo TAG_1, TAG_2 etc

Accesso ottimizzato alle DB

Le DB per default sono **ottimizzate**, **cioè di default non è possibile accedere alle variabili di una DB tramite indirizzo ma solo ed esclusivamente tramite il simbolico**.
Facendo tasto destro su una DB, all'interno delle **Proprietà / Attributi** c'è una spunta "**Accesso Ottimizzato al blocco**". Togliendo questa spunta, all'interno della DB compare una nuova colonna denominata **offset** e diventa possibile accedere alla variabile anche attraverso il suo indirizzo assoluto di memoria. Dopo aver tolto la spunta occorre ricompilare il progetto
Nei vecchi PLC esistevano SOLO DB di tipo non ottimizzato

L'accesso non ottimizzato è stato mantenuto per consentire l'interfaccia con terze parti che hanno necessità di conoscere gli offset delle variabili del plc.

E' comunque consigliato di utilizzare sempre solo il simbolico che, oltre alla leggibilità, è più performante rispetto all'indirizzamento assoluto in quanto in fase di compilazione produce un codice altamente ottimizzato. Qualunque cosa è oggi fattibile con il simbolico :

- I protocolli come ad esempio OPC UA non utilizzano più gli offset ma il simbolico
- Se si desidera accedere ad un singolo bit di una variabile intera o word si può utilizzare il simbolo seguito dalla seguente sintassi:

nomeVariabile.x0	bit 0 della variabile
nomeVariabile.b0	byte 0 della variabile
nomeVariabile.w0	word 0 della variabile

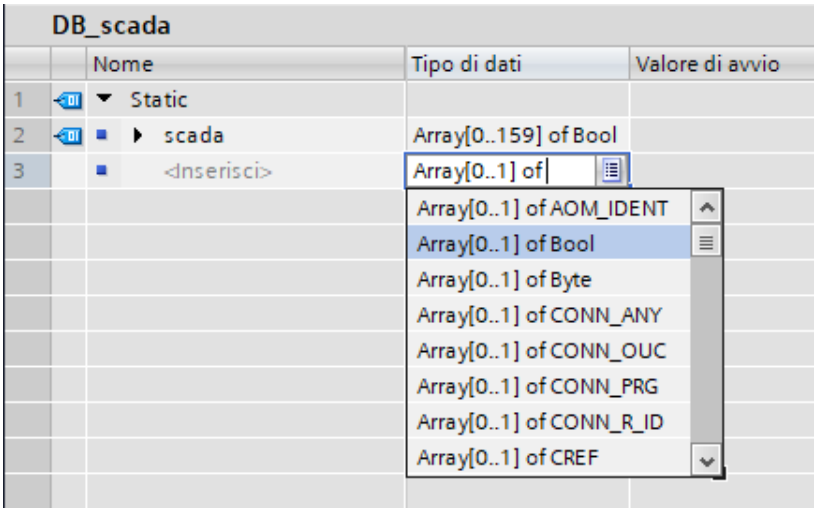
Tipi di Dati

Le variabili di un PLC possono essere definite come :

- BIT
- BYTE (0 – 255)
- WORD (2 bytes / 16 bit) Intero senza segno da 0 a 65000
- INT (2 bytes / 16 bit) intero con segno, tra ±32000)
- DWORD (32 bit) double word
- DINT (32 bit) double int
- LWORD (64 bit) long word
- LINT (64 bit) long int
- Real (32 bit)
- Long Real (64 bit)
- **Time** tipo da assegnare alle variabili di tipo “tempo” (ad esempio per memorizzare in un merker il tempo di conteggio di un certo timer)

Bisogna fare attenzione al fatto che alcuni blocchi (ad esempio gli Operatori di Confronto e gli Operatori in Virgola Fissa) accettano in ingresso soltanto **INT** e non **WORD** o viceversa.
Nelle ultime versioni viene praticamente sempre eseguita una conversione implicita del tipo.

In TIA Portal sono disponibili anche gli Array:



Sintassi per l’indirizzamento assoluto

Merker

M4.0	byte 4 bit 0
MB4	intero byte 4
MW4	bytes 4 e 5
MD4	bytes 4 5 6 e 7

DB

DB12.DB**X**0.3	primo byte della DB12, bit n. 3
DB12.DB**B**0	intero primo byte della DB12
DB12.DB**W**0	prima word della DB12 (bytes 0 e 1)
DB12.DB**D**0	prima double word della DB12 (bytes 0 1 2 e 3)

Valori immediati

I valori immediati possono essere utilizzati in modo diretto all'interno di una istruzione oppure in fase di configurazione di una DB

I valori immediati devono essere scritti in modo diverso a seconda del tipo di dato:

BYTE	**B#16#**0A	Il 16 indica che il valore è espresso in formato esadecimale
WORD	**W#16#**CC0A	Il 16 indica che il valore è espresso in formato esadecimale
DWORD	**DW#16#**0	Il 16 indica che il valore è espresso in formato esadecimale
INT	0	Normale numero decimale
DINT	**L#**0	Long Int

Per i timer si usano valori nel seguente formato: S5**T#5S** (5 sec).
In TIA Portal si utilizza solo più T# (non c'è più S5 davanti)
Il valore S5**T#10MS** significa 10 msec (intervallo minimo di conteggio).

I principali blocchi di programmazione

Per aggiungere nuovi blocchi al programma utilizzare
Blocchi di Programma/ Inserisci Nuovo Blocco.

OB organization block (blocco organizzativo)
FC function
FB function block (base per una libreria)
DB data block

I blocchi OB

Blocchi Organizzativi di sistema già scritti che effettuano operazioni particolari.
Non possono essere richiamati dall'utente, ma vengono lanciati direttamente dal plc:
E' il PLC che decide quando questi moduli devono essere eseguiti.

Le ultime versioni di TIA Portal consentono di parametrizzare il numero di ognuna delle seguenti OB

OB1 e Tempo di Ciclo

Rappresenta il **Programma Principale** richiamato ciclicamente all'infinito.
Ogni nuovo progetto contiene soltanto il blocco **OB1**.
La durata del tempo di ciclo non è mai regolare e stabile. Dipende da:

- dal numero di istruzioni inserite all'interno dell'OB1 medesimo
- dal numero di moduli di campo da leggere
- dalle operazioni che vengono eseguite (possono esserci delle IF che saltano alcune porzioni di codice)
- il ciclo può essere interrotto da operazioni asincrone (ad esempio legate alla periferia remota) e dunque può durare molto più del previsto.

OB100 - startup

viene eseguita una sola volta ad ogni STOP RUN del plc. All'interno della OB è possibile leggere gli ingressi e scrivere le uscite. Infatti la OB viene eseguita solo DOPO l'aggiornamento degli ingressi

OB cyclic interrupt (30 - 38)

Sono OB richiamate "**a intervalli regolari"** sulla base di un orologio hw.
Con queste OB si ha la garanzia che il codice venga eseguito esattamente nel tempo impostato, senza essere influenzato dai tempi di scansione.
Il tempo può essere impostato nella finestra di destra al momento della creazione

Allo scadere del tempo impostato, viene interrotta l'esecuzione di OB1, viene avviato l'OB a tempo e, alla fine dell'esecuzione delle istruzioni contenute nell'OB a tempo, si torna all'esecuzione del programma normale, dal punto in cui era stato interrotto.
Generalmente si cerca di programmare in questi OB il minimo indispensabile.

Su cpu di livello inferiore, è disponibile solo **OB35** che, di default, è configurato per l'avvio ciclico ogni **100ms**. Nella configurazione hardware della cpu si può modificare questo parametro fino ad un minimo di 10 msec. La configurazione può essere fatta dalle Proprietà della CPU.
Se si imposta un tempo troppo basso, che non si riesce a rispettare, viene avviato l'**OB82.**
Se l'OB82 non esiste, la cpu va in stop.

OB hardware interrupt (40 – 48)

Dette anche **realtime** (RT) o **SYNC**. richiamato in corrispondenza di uno specifico evento hw, legato ad esempio all'accensione di un ingresso digitale. Quando il sensore va a 1, automaticamente il PLC richiama l'OB indicato senza attendere l'esecuzione dell'OB1.

Se il segnale del sensore dura meno del ciclo OB1 il PLC potrebbe non accorgersi che è arrivato il segnale. Legge 0 prima del ciclo e poi di nuovo 0 al termine del ciclo. Per rilevare questo picco occorre una scheda di ingresso che sappia fare questo lavoro (le schede **HF** heigh frequency hanno processori veloci) e poi si asegna questo ingresso ad un OB di tipo RT

Ogni OB ha una sua priorità. Quando si verifica un evento, se è in esecuzione una OB con priorità inferiore, questa viene interrotta e viene mandata in esecuzione la OB con priorità superiore.
OB1 ha una priorità non modificabile che è la priorità minima, e dunque interrompibile da tutti.
La priorità è definibile all'interno delle proprietà delle OB in fase di definizione

Disponibili solo nei PLC1500 e non nei 1200
Nei PLC 300 se l'OB non c'era la cpu andava in stop. Ora non più

OB 80 time error interrupt

watch dog Richiamata quando viene superato un tempo max di esecuzione liberamente impostato. Va bene anche vuota, purchè esista. Nei PLC 300 se l'OB80 non esisteva, in caso di superamento del tempo max, la cpu andava in stop.

I blocchi FC

Fun**C**tion. E' il blocco più semplice. Non si può scrivere tutto il codice all'interno di OB1. Le FUnction sono semplici funzioni richiamabili dal programma principale o da un'altra funzione.
Tutto ciò che non viene richiamato da OB1 è come se non ci fosse: il plc non lo vede.
Una function:

- Può ricevere uno o più parametri dal chiamante
- Può restituire uno o più risultati al chiamante, impostati durante l'esecuzione della FC medesima
- Può accedere alle variabili globali definite all'interno delle DB o dei merker
- **non ha variabili interne**, cioè non ha blocchi dati di istanza, non ha memoria.

Il passaggio di parametri ad una FC

Per ogni FC si possono definire dei parametri:

- in **Ingresso** (cioè che compariranno sul lato sinistro del blocchetto), a cui il chiamante potrà passare dei valori
- in **Uscita** (cioè che compariranno sul lato destro del blocchetto) dai quali il chiamante potrà leggere i risultati restituiti dalla funzione
- in **Ingresso_Uscita** (cioè che compariranno sia sul lato sinistro del blocchetto sia sul lato destro). E' l'equivalente del passaggio di un parametro per riferimento, in cui il chiamato può elaborare il dato ricevuto e restituirlo come parametro al chiamante
- di tipo **Temp**, cioè sostanzialmente variabili locali alla funzione che vengono riazzerati in corrispondenza di ogni chiamata
- di tipo **Const**, cioè leggibili ma non modificabili dalla procedura

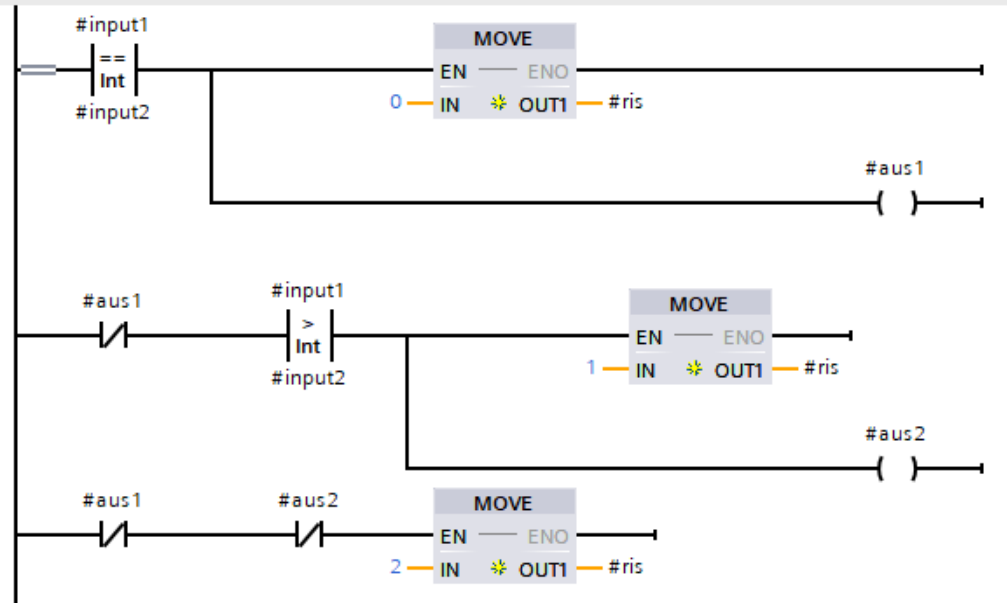
Tutti questi parametri vengono salvati in un'area di memoria di sistema non accessibile al programmatore.

Esempio

FC che esegue il confronto fra due numeri e restituisce:
0 se sono uguali
1 se il primo maggiore,
2 se il secondo è maggiore

Blocco_1

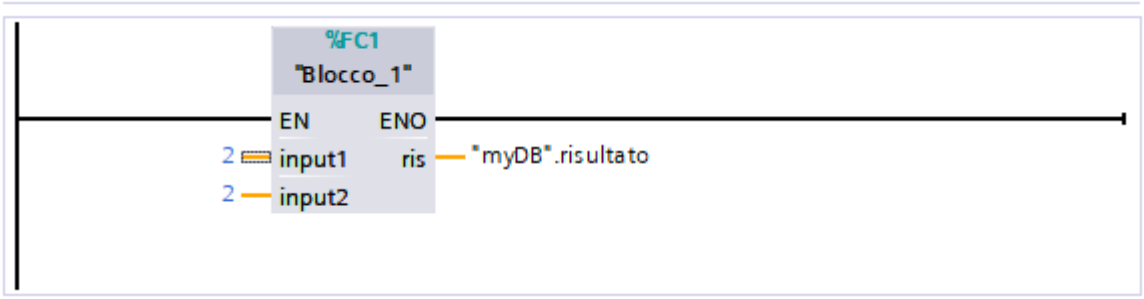
	Nome	Tipo di dati	Valore di default	Commento
1	Input			
2	input1	Int		
3	input2	Int		
4	<Inserisci>			
5	Output			
6	ris	Int		
7	<Inserisci>			
8	InOut			
9	<Inserisci>			
10	Temp			
11	aus1	Bool		
12	aus2	Bool		



Notare come tutti i parametri sia di ingresso vengano visualizzati con davanti un **cancelletto** che indica appunto che si tratta di una variabile locale. Inoltre non hanno l'icona viola

Richiamo di una FC

Per richiamare una FC è sufficiente trascinarla all'interno di un segmento vuoto



I blocchi FB

Functional **B**lock. Blocchi funzionali. **FB** è una FC con memoria.
Rispetto alla FC :

- Tutte le variabili (input, output, temp, const) vengono salvate all'internodi una DB dedicata che funge da "**DB di istanza**" che viene automaticamente ricreata al momento dello stop / run.
- Oltre alle variabili temporanee che vengono azzerate dopo ogni chiamata, sono ora disponibili anche variabili **statiche** che mantengono il valore tra una chiamata e l'altra
- Tutte le variabili definite all'interno della DB di istanza (input, output, statiche), ad eccezione delle temporanee, avendo visibilità globale, saranno visibili ed accessibili da OB1 o qualsiasi altra funzione
- A differenza delle FC, nel caso delle FB è possibile assegnare ai parametri di input uno start value che verrà automaticamente assegnato al parametro in corrispondenza di uno stop /run nel caso in cui il programmatore non abbia specificato, in fase di chiamata, un valore diretto. In programmatore può comunque modificare in fase di chiamata il valore da assegnare al parametro, senza necessità di eseguire uno stop/run

Chiamate differenti possono passare alla FB una DB differente, facendo in modo che la FB operi su DB diverse a secondo dei contesti

Si supponga di dover gestire più serbatoi tutti identici fra loro.

Prima soluzione

Creo una DB per ogni serbaoio
Creo una FC per ogni serbatoio che usa i dati della relativa DB
Svantaggi : devo copia incollare più volte la stessa FC variando in ogni FC le variabili utilizzate indirizzandole sulla DB opportuna

Seconda soluzione

Usiamo una unica FC e dichiariamo tutte le variabili che servono come variabili locali all'interno del blocco oppure come variabili di input
In corrispondenza del richiamo occorrerà passare le variabili necessarie.
OB1 richiamerà più volte la stessa FC passandogli di volta in volta le variabili sulle quali la FC dovrà lavorare. Comunque scomodo da scrivere

Terza soluzione

Uso una FB. Struttura dei dati locali e codice interno sono gli stessi della soluzione 2
Nel momento in cui richiamo la FB da OB1, step7 automaticamente mi richiede di creare ed assegnare una DB di istanza dedicata. Il programmatore, se vuole, anziché passare i parametri in fase di chiamata, li può andare a scrivere direttamente all'interno della DB di istanza.
In fase di chiamata verranno visualizzati i valori di start value, che però a run time verranno sovrascritti dalle MOVE scritte dal programmatore

Consigli:
All'interno di FC e FB non usare mai variabili globali, ma sempre solo variabili locali / statiche. In questo modo le FC / FB diventano riutilizzabili per applicazioni future e/o costruire una libreria.

Le tabelle di controllo (VAT)

Sono tabelle che consentono di monitorare Run Time ed eventualmente forzare qualunque variabile del PLC.

- I forzamenti possono essere eseguiti anche dalle righe di codice mediante tasto destro oppure, ad esempio, dall'interno di una DB
- E' possibile monitorare anche Ingressi e Uscite.
- E' possibile impostare (tramite il campo Modify Value) e selezionare tramite apposito check più variabili contemporaneamente, che verranno forzate in corrispondenza del click sul pulsante "**fulmine**" che invia il valore e poi subito lo rilascia
- **NON** è possibile forzare gli ingressi

Le tabelle di controllo sono residenti sul PC e NON vengono downloadate su PLC

Molto comodo il **5° pulsantino** : **nascondi tutte le colonne di comando** che danno solo fastidio.
Penultimo pulsantino: **controlla tutto** consente di controllare tutte le variabili presenti nella VAT

Per forzare gli ingressi è possibile:

- Fare uso di una tabella di forzamento. In questo modo gli ingressi vengono settati **PRIMA** della scrittura nell'ara delle immagini di processo. Attenzione che il forzamento rimane attivo fino a quando non viene esplicitamente disattivato
- Fare uso di PLC SIM

Le principali istruzioni del linguaggio KOP

Sopra il KOP editor è posizionata una barra delle istruzioni preferite che può essere personalizzata aggiungendo le istruzioni utilizzate più frequentemente

I dati possono arrivare al plc attraverso diversi canali:

- Attraverso il bus di backplane collegato ai moduli di campi (dati ciclici)
- Attraverso le interfacce relative alla periferia decentrata profinet o profibus
- Attraverso un pannello operatore (dati asincroni, non ciclici)

Bobina ( )

Viene portata ad 1 quando le condizioni di ingresso sono vere e ritorna a 0 quando le condizioni ritornano ad essere false.
Può essere richiamata in un unico punto del programma. Se fosse richiamata in più punti, l'ultimo richiamo maschererebbe i precedenti, che dunque non avrebbero più nessun effetto.

Le istruzioni SET (S) e RESET (R)

SET Consente di attivare una bobina autoritentiva che rimane attiva anche quando le condizioni di ingresso ritornano ad essere false.
Detto anche **Latch**

RESET Consente di spegnere una bobina autoritentiva
Detto anche **Unlatch**

Timer

Nei vecchi PLC 300 richiedevano aree di memoria dedicate.
In TIA Portal questi tmer sono ancora presenti per ragioni di compatibilità, ma sono stati "nascosti" all'interno della sottocartella "Legacy". I nuovi timer di TIA portal sono tutti standardizzati IEC e sono sostanzialmente delle FB che, in corrispondenza di ogni istanza, utilizzano una specifica DB di istanza (la prima libera). **Le DB assegnate ai timer vengono memorizzate all'interno dei Blocchi di sistema.** I nuovi timer IEC di TIA Portal sono in tutto quattro : TON, TONR, TOF, TP

TON Timer On Delay. Ritardo all'inserzione.
Nel momento in cui le condizioni di ingresso diventano vere inizia il conteggio. Se le condizioni rimangono vere per tutto il tempo di conteggio, al termine del conteggio viene generato il DONE, che rimane ad uno fino a che le condizioni di ingresso rimangono vere.
Se le condizioni di ingresso diventano false durante il conteggio, il tempo viene riazzerato ed il DONE non viene generato
In pratica genera un Done con un certo ritardo rispetto al verificarsi di un fronte di salita in ingresso

TONR è un temporizzatore con memoria (ton ritentivo). Se le condizioni di ingresso diventano false il timer NON si resetta e, quando le condizioni ritorneranno vere, il timer riprenderà il conteggio dal valore a cui era arrivato.

TOF Timer Off Delay genera immediatamente il DONE quando le condizioni di ingresso vanno a 1. Quando le condizioni di ingresso diventano false, mantiene l'uscita attiva per il tempo indicato e poi torna a 0. Es: Dopo che il motore si è spento la ventola continua a girare per 20 sec

TP Timer Pulse nel momento in cui le condizioni di ingresso diventano vere da istantaneamente l'impulso di uscita che mantiene attivo per il tempo stabilito. Generatore di impulsi.

Per tutti questi timer se l'uscita del timer non è utilizzata, il timer NON si attiva al conteggio

Come detto i vecchi timer s5Time (SE, SS, etc) sono disponibili all'interno della sottocartella **Legacy** . Erano dei timer hardware, con un numero max di timer utilizzabili pari a 256 ed un tempo massimo di conteggio pari a **299 minuti**
I nuovi timer IEC possono invece essere utilizzati in numero pressochè illimitato e contano sia in positivo che in negativo fino a **24 giorni**

Struttura di funzionamento dei timer IEC:

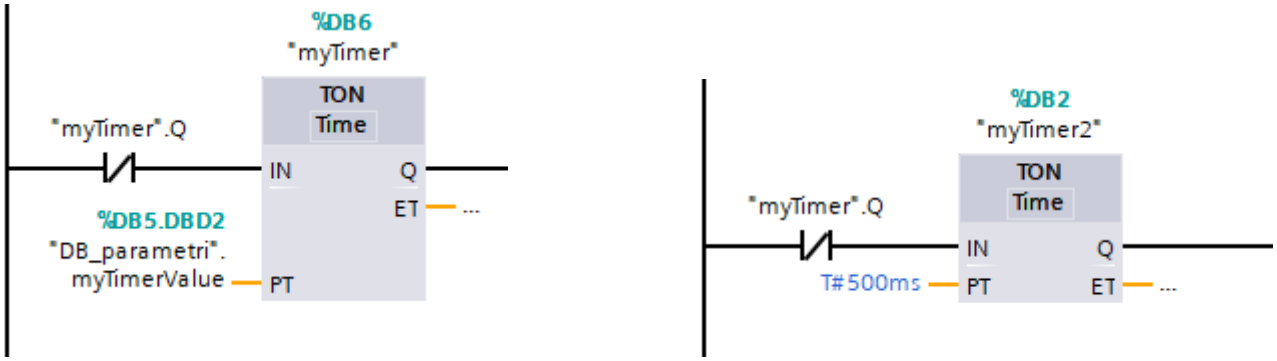
Parametri	Dichiarazione	Tipo di dati	
		S7-1200	S7-1500
IN	Input	BOOL	BOOL
PT	Input	TIME	TIME, LTIME
Q	Output	BOOL	BOOL
ET	Output	TIME	TIME, LTIME

IN condizioni di ingresso per l'avvio del timer
PT tempo di conteggio
Q tipico DONE attivato al termine del conteggio
ET valore istantaneo del conteggio

Funzionamento del TON

(equivalente al vecchio SE):

- quando l'ingresso **IN** va a 1 comincia a contare il tempo indicato all'interno di **PT** e si resetta se **IN** ritorna a 0.
- **PT** è una variabile di tipo **TIME** che può essere memorizzata all'interno di un merker o di una DB (es DB_parametri.myTimerValue) oppure può anche essere scritta in modo diretto in formato **T#1s500ms**
- Nel momento in cui termina il conteggio attiva l'uscita **Q** (done) che rimane attiva fino a quando **IN** non ritorna a 0.
- Il campo **ET** restituisce una variabile sempre di tipo TIME contenente il valore attuale di conteggio



L'istruzione MOVE

Consente di **copiare** un byte (o word o double word o float) da una certa posizione di memoria ad un'altra. E' esattamente analoga alla move dei linguaggi assembly

Operatori Aritmetici

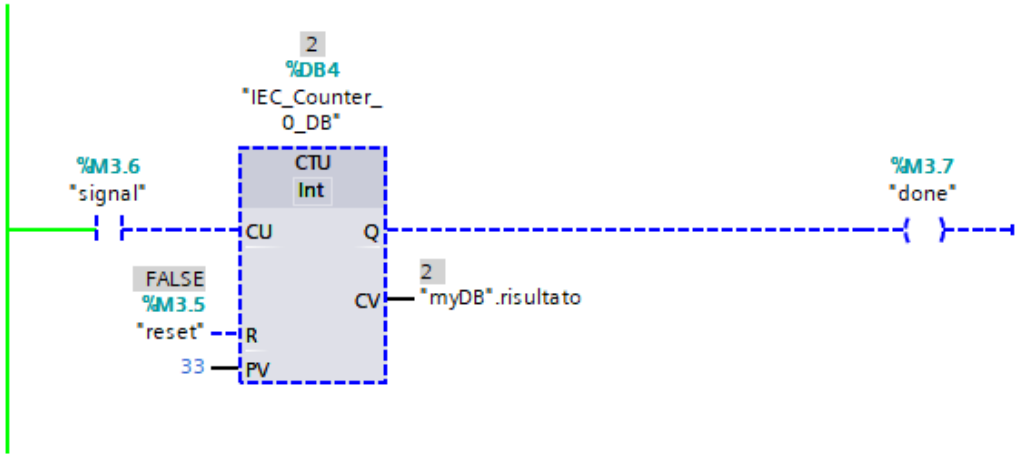
somma, sottrazione, moltiplicazione, divisione, confronto.
Questi comandi **NON lavorano** soltanto sul fronte, nel senso che l'azione viene eseguita ad ogni scansione in cui la condizione antecedente risulta vera. Per cui se vuole eseguire l'azione **UNA VOLTA SOLA** occorre necessariamente creare un fronte.

I Contatori

Esattamente come i timer sono anch'essi delle FB che necessitano di una apposita DB di istanza
Incrementano di 1 il conteggio ogni volta che la condizione a monte è vera.
A differenza degli operatori aritmetici, i contatori lavorano sui fronti di salita dell'ingresso

I nuovi counter IEC sono in tutto 3: CTU, CTD e CTUD

CTU conta in avanti partendo da 0 e quando raggiunge il valore impostato setta un bit di "Done" che rimane ad 1 finchè il contatore non viene riazzerato manualmente con fronte di salita sul pin di reset. Se mentre è ad uno arrivano altri fronti di salita sull'ingresso, questi non vengono più sentiti
L'esempio riporta un counter CTU che dopo 33 impulsi di **`signal`** attiva il **`done`**



CTD conta all'indietro partendo dal valore impostato e quando raggiunge lo zero attiva il bit di done

CTUD incrementa il conteggio in corrispondenza di un fronte di salita sul pin CU e decrementa il conteggio in corrispondenza di un fronte di salita del pin CD

A differenza dei temporizzatori tutti i counter funzionano anche se non si utilizza l'uscita Q

L'istruzione ONE SHOT

Talvolta in un programma risulta necessario creare di fronti di salita, cioè fare in modo che l'uscita venga attivata per una singola scansione nel momento in cui le condizioni di ingresso diventano vere e poi ritorni immediatamente a zero (ad esempio nel momento in cui si deve incrementare di 1 una **variabile intera** tramite l'istruzione ADD)

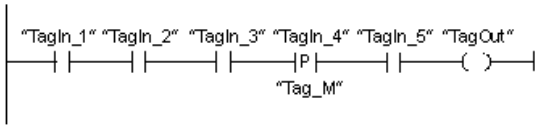
A tale fine sono disponibili le istruzioni **|P|** (fronte di salita) e **|N|** (fronte di discesa)

L'istruzione **|P|** deve essere applicata ad una specifica variabile di tipo bit e chiude il contatto per una singola scansione nel momento in cui il valore della variabile passa da 0 a 1

L'istruzione **|N|** deve essere applicata ad una specifica variabile di tipo bit e chiude il contatto per una singola scansione nel momento in cui il valore della variabile passa da 1 a 0

Entrambe queste istruzioni hanno bisogno di un bit di appoggio (nell'esempio **Tag_M**) in cui in pratica salvano il valore del bit d controllare. Ad ogni scansione confrontano il valore corrente del bit con il valore salvato nella variabile di appoggio, riuscendo in questo modo ad individuare i fronti.

L'esempio seguente mostra il funzionamento dell'istruzione:



L'operando "TagOut" viene impostato se vengono soddisfatte le seguenti condizioni:

- Gli operandi "TagIn_1", "TagIn_2" e "TagIn_3" hanno lo stato di segnale "1".
- Nell'operando "TagIn_4" è presente un fronte di salita. Lo stato di segnale dell'interrogazione precedente viene salvato nel merker del fronte "Tag_M".
- Lo stato di segnale dell'operando "TagIn_5" è "1".

NB: **Ogni variabile di appoggio dei one shot può essere utilizzata UNA SOLA VOLTA all'interno dell'intero programma**

Nota
Sempre riguardo al one shot sono anche disponibili due istruzioni di uscita **(P)** e **(N)** le quali sono sostanzialmente delle semplici bobine che vengono chiuse per una singola scansione nel momento in cui tutte le condizioni antecedenti diventano vere.

Vanno bene nel caso in cui un segnale digitale debba essere attivato per una singola scansione.

- Come primo parametro (sopra) si mette **l'uscita digitale** da controllare
- Come secondo parametro (sotto) si mette un bit di appoggio utilizzato per salvare lo stato complessivo delle condizioni antecedenti

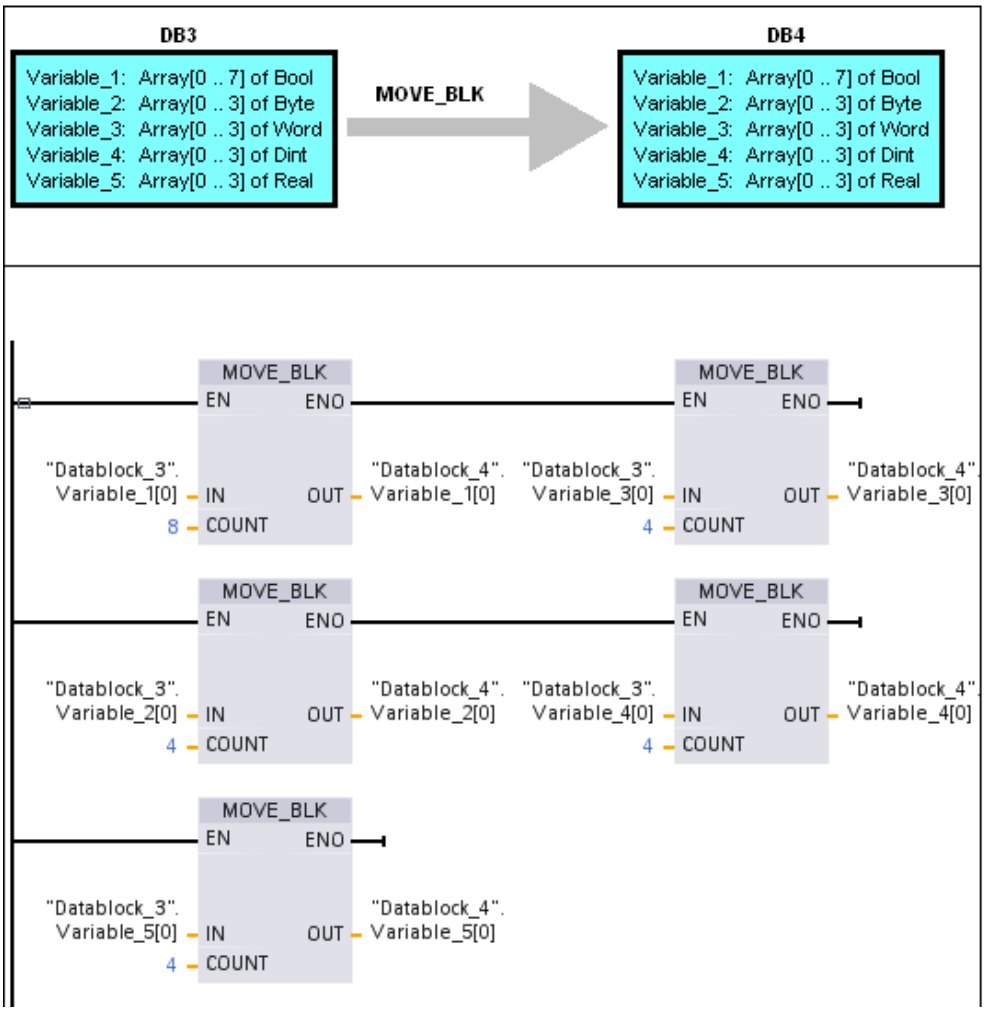
L'istruzione CALCULATE

E' molto flessibile:

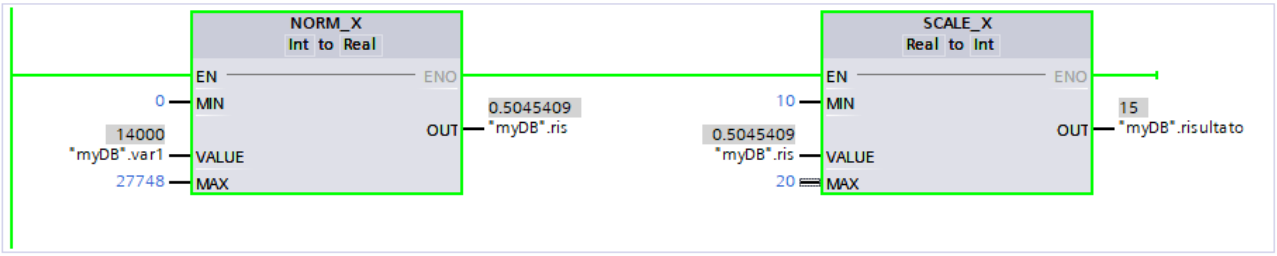
- Può avere un numero arbitrario di parametri (Cliccando sulla stellina gialla posso aumentare a piacimento il numero degli ingressi da utilizzare)
- Può implementare una qualunque operazione algebrica fra gli operandi.
  Facendo doppio click sul tab centrael si apre una finestra di scrittura dell'operazione da eseguire; ad esempio : OUT := IN1 + IN2 + IN3
- Il tipo indicato in alto (int, real, etc) indica il tipo di dati in cui vengono eseguite le operazioni. Ad esempio calculate_real significa che i calcoli vengono eseguiti in real ed anche il risultato sarà un real. Se i parametri sono di tipo diverso viene eseguito un cast automatico; in tal caso davanti all'ingresso compare un quadratino grigio che significa che quella variabile di ingresso verrà automaticamente convertita in REAL.

L'istruzione MOV_BLK

L'istruzione MOV_BLK consente di copiare blocchi di dati.
E' utilizzabile solo ed esclusivamente su singole variabili di tipo Array.



Conversione di un valore Analog Input in un valore con significato



Supponendo che la variabile var1 provenga da una analogia ed abbia un valore compreso tra 0 e 17748, **NORM_X** provvede a scalare questo valore in un real compreso tra 0 e 1 (nell'esempio 0.5045)
SCALE_X provvede a sua volta a scalare il real di appoggio restituito dalla funzione precedente in un nuovo INT compreso ad esempio tra 10 e 20 (nell'esempio 15)

PLC SIM

Aprire il progetto **PLC** su TIA Portal ed assegnare al PLC un qualsiasi indirizzo IP.
Scrivere alcune istruzioni di prova.

Con PLCSIM chiuso cliccare il pulsantino **Avvia Simulazione** (posizionato subito dopo i pulsanti di download e upload). Questo comando automaticamente apre PLC SIM

In automatico si apre anche la solita finestra di download.
Cliccare su **AVVIA RICERCA**
Viene automaticamente individuato l'emulatore a cui viene assegnato l'indirizzo IP indicato nella configurazione della CPU.
CARICA

Da TIA Portal oppure da PLC SIM avviare la CPU portandola in RUN

TIA Portal si comporta come se avesse una CPU fisica collegata, quindi è possibile il debugger online come se fossimo collegati ad un vero PLC.
Tutto ciò **INDIPENDENTEMENTE** dalla creazione di un progetto su PLCSIM.

Creazione di un progetto su PLCSIM

Una volta avviato PLCSIM come sopra indicato, pulsantino in alto a destra consente di ingrandire PLC SIM e creare un progetto

Lo scopo della creazione di un nuovo progetto è quello di definire una tabella di controllo degli ingressi e consentire l'avanzamento del programma.
Da questa tabella è possibile **forzare gli ingressi** esattamente come dalla tabella di forzamento di TIA Portal. Le uscite sono invece in sola visualizzazione

Se non mi serve forzare ingressi / uscite è inutile creare / aprire il progetto PLC SIM con il tastino in alto a destra. Il forzamento lo posso fare da PLC

Project / create new project
All'interno di Configurazione Dispositivo si vede la CPU in esecuzione come se fossimo su un dispositivo reale. PLCSIM però non aggiorna i led relativi a ingressi e uscite

Creare allora una tabella dei simboli impostando il nome diretto del simbolo oppure anteponendo il nome della DB in caso di variabili contenute nelle DB. Attenzione che sono visibili soltanto le variabili dichiarate "accessibili da HMI" all'interno della DB del PLC.

Il copia incolla dalla tabella di contrllo del plc sembra problematico. Conviene impostare manualmente le variabili di controllo, ma è abbastanza veloce

Nel caso di ingressi o bit di memoria, cliccando sul check box della colonna **bit**, il bit corrispondente viene automaticamente portato a 1.
Cliccando su stati.bAuto l'impianto si avvia.
Non è possibile portare a 1 i bit di uscita, in quanto il loro valore vieen sovrascritto dal PLC

Altre operazioni consentite con PLC SIM

IN PLC SIM è possibile :

- definire delle sequenze: ad esempio M0.0 va a 1 per 3 sec poi va 0 per 4 sec etc
- definire degli eventi, ad esempio un evento di simulazione di errore hardware (es estrazione di un modulo di IO). Novità di TIA Porta V16
  Tabella Eventi / aggiungi nuova tabella eventi.
  LADDR indica il modulo su cui intendo simulare l'errore

  OB83 "Pull or Plug of module" è una OB del PLC che interrompe la OB1 e viene richiamata al suo posto in caso di estrazione di un modulo. Lì dentro posso scrivere una riga del tipo : always true -> set bit M9.0
  Il bit M9.0 rimarrà ad 1 fino quando non si risolve il problema
- Esiste anche un **PLC SIM Advanced** che integra anche il web server consultabile tramite browser. PLC SIM base non c'è l'ha

Aperture successive del progetto

Avviare SEMPRE l'emulatore partendo da PLCSIM chiuso.
Una volta avviato l'emulatore ingrandire la finestra ed aprire il progetto precedentemente creato con tutte le sue variabili.

Ricerche

Quando si è all'interno del codice, per vedere rapidamente un elenco di tutti i punti in cui viene utilizzata una variabile, è sufficiente .

- Selezionare la variabile
- Tasto destro / **vai al punto di applicazione**

Per avere invece la Lista completa degli utilizzi di un indirizzo occorre invece utilizzare :

Strumenti / Dati Riferimento / Filtra consente di impostare un Filtro di Ricerca. Ad esempio una singola DB. Volendo ricercare gli utilizzi della DB 10 selezionar DB ed impostare 10
Strumenti / Dati Riferimento / Visualizza / Riferimenti Incrociati vengono mostrati gli utilizzi di tutte le variabili impostate mediante il filtro precedente.

Diagnostica di un Fault

Quando il PLC per qualche ragione va in fault, per vedere eseguire una diagnostica si possono utilizzare diversi approcci :

- Utilizzare il monitor di bordo del PLC
- Utilizzare TIA Portal per interrogare il buffer di diagnostica della CPU.
  - voce "**ONLINE & Diagnostica**" nella cartella del PLC di lavoro oppure
  - tasto destro sul PLC / ONLINE & Diagnostica /
  HMI: c'è un tool di diagnostica che è sufficiente trascinare all'interno di una qualsiasi pagina. Questo tool copre l'intera pagina e visualizza anche lui il buffer di diagnostica del PLC
- Web server

Archiviazione di un progetto

Progetto / Archivia
Progetto / Disarchivia

Copia di un blocco da un vecchio progetto

Visualizza / progetti di riferimento
Nella finestra “progetti di riferimento” apro il progetto dal quale voglio copiare un blocco di codice. Il progetto viene aperto in sola lettura. Posso aprire i file e copiare quello che mi server o trascinare un intero file.

Note sulla simbologia (codifica IEC)

H sono i segnalatori
Emergenza è un pulsante a fungo Normalmente Chiuso