TUTORIAL per serie TPAC

☎ 011/9664616

M7000_14
03/11

1.0 INDICE

2.0 GENERALITA'

TPAC ha diverse periferiche che possono essere divise in quattro sezioni (vedere i manuali utente per i dettagli):

IO locali
Input digitali PNP 0-24V
Output digitali PNP 0-24V
Ingressi analogici; frequenza di campionamento max 500 campioni al secondo
Uscite analogiche, risoluzione 12 bit, frequenza di aggiornamento max 500Hz
encoder incrementali massimo conteggio 2^{28}
rpm
IO remoti
2 canali CAN 1 Mb/sec con protocollo CAN OPEN.
1 RS485 o 1 RS232, 115kbit al secondo
porte TPAC
1 LAN 10Mbit al secondo
1 USB 1.0 master
Human Machine Interface
LCD 320 x 240 STN a colori
Touch screen
Tastiera a 16 tasti

Abbiamo visto le porte del TPAC, ora si descriverà come usare ogni porta in un progetto PAC.
Il ciclo PLC più breve può essere di 1ms con ± 30µs di accuratezza. Il tempo ciclo è un parametro configurabile dall'utente.

Un programma TPAC è composto da due parti:

Un programma PLC creato utilizzando il software LogicLab di Axel.
Interfaccia uomo macchina (HMI) sviluppata usando il programma PageLab di Axel

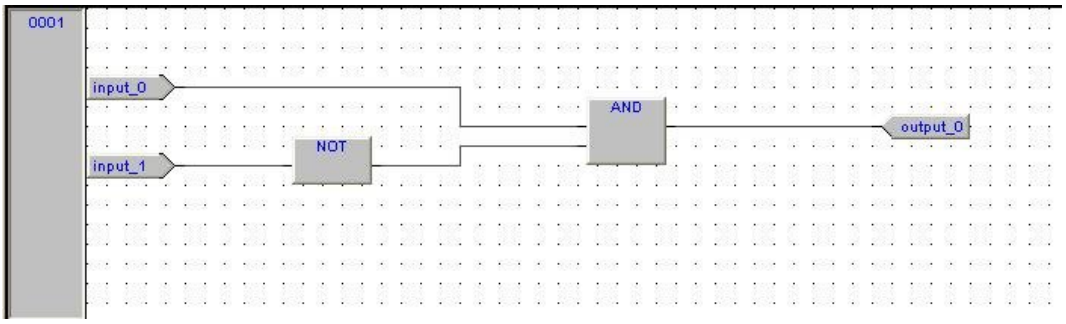
Un programma PLC può essere sviluppato nei seguenti linguaggi di programmazione derivanti dallo standard IEC 61131-3:

FBD	Functional Block Diagram	Grafico	Tipo schema elettrico
LD	Ladder	Grafico	Schema Ladder
SFC	Sequential Function Chart	Grafico	Diagramma degli stati
ST	Structured Text	Testuale	Linguaggio Pascal-like
IL (AWL)	Instruction List	Testuale	Linguaggio Assembler-like

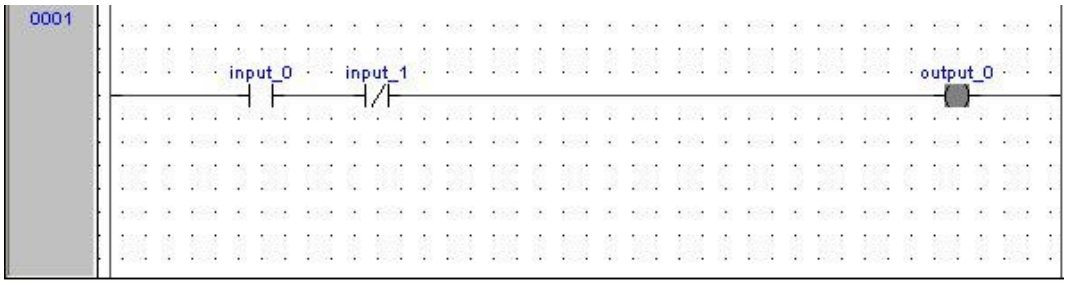
Per spiegare i vari linguaggi di programmazione, sviluppiamo alcuni semplici programmi.

Il programma che realizziamo imposta l'uscita output_0 quando l'ingresso logico input_0 è 1 e l'ingresso input_1 è 0.

2.1 ESEMPIO FBD



2.2 ESEMPIO LD



2.3 ESEMPIO ST

```
IF INPUT_0 AND (NOT INPUT_1 ) THEN
        OUTPUT_0:=1;
ELSE
        OUTPUT_0:=0;
END_IF;
```

2.4 ESEMPIO IL

```
START:
        LD      INPUT_1
        XOR     INPUT_1
        AND     INPUT_0
        ST      OUTPUT_0
```

Successivamente vedremo in dettaglio alcuni di questi linguaggi (vedere il capitolo: “Sviluppare un semplice programma”).

3.0 INSTALLARE LA SUITE DI SVILUPPO

TPAC è provvisto di un CD, nel quale sono inseriti i software che l'utente deve installare su un personal computer con sistema operativo Windows.

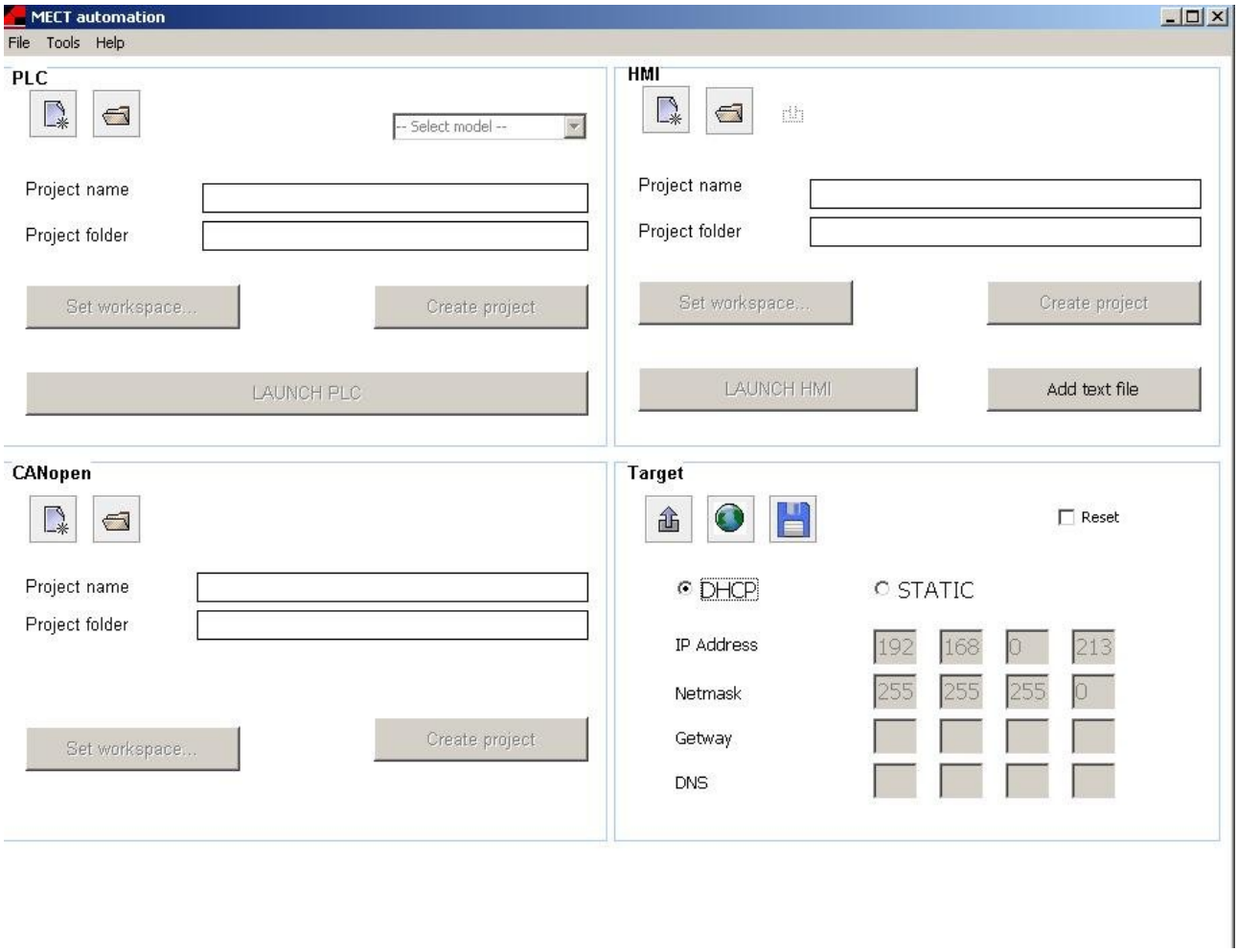
I software forniti sono:
Mect Dashboard
Axel LogicLab: software PLC
Axel PageLab: software Human Machine Interface
Axel LLComp: software di comunicazione per scaricare il programma PLC nel TPAC
Trama Netbuilder: configuratore di rete CANOpen
Microsoft Windows .net framework 2.0: aggiornamento di windows necessario se già non disponibile su PC.
SUN J2SE runtime di java.

Per installare l'intera suite, è necessario eseguire dal CD il programma TPAC.exe. Una installazione guidata aiuterà l'utente.

Al completamento dell'installazione un' icona TPAC apparirà sul desktop del PC. Per la registrazione dei software installati vedi "Appendice 3".

4.0 CONFIGURAZIONE DEL TPAC

Prima di iniziare a sviluppare un progetto è necessario configurare il TPAC, cliccando sull'icona TPAC.



Si aprirà una finestra nel programma **Dashboard** che presenterà quattro sezioni.

Per creare un nuovo progetto nella sezione denominata **PLC** cliccare sull'icona nuovo progetto e selezionare il modello di hardware per il quale si sta creando il progetto. Il tasto **"Imposta directory"** consente di impostare il percorso in cui verrà creato il progetto. Nel campo **"Nome Progetto"** indicare il nome di attribuire al progetto quindi cliccare il tasto **"Crea Progetto".** Per avviare l'ambiente di programmazione premere **"Avvia PLC"**.

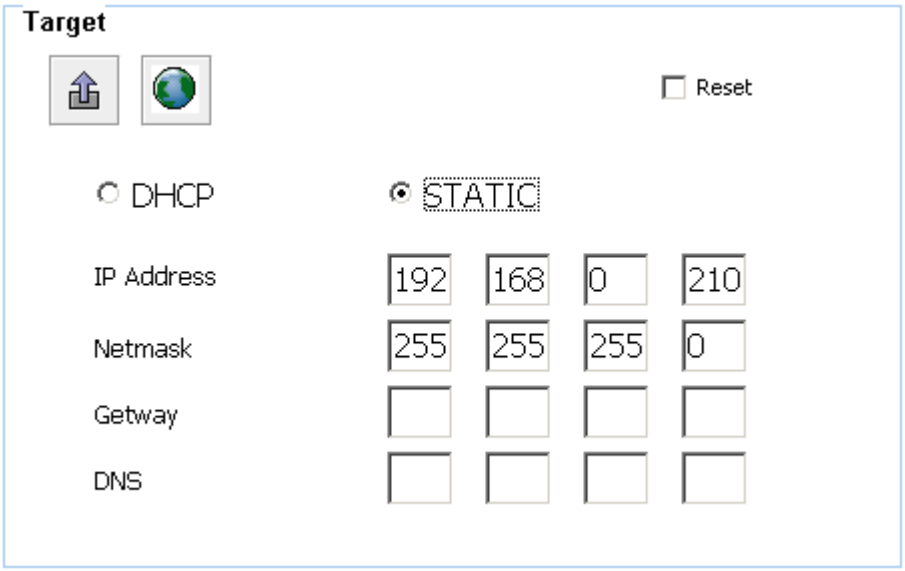
Nel riquadro Target sono disponibili i pulsanti:

- **Export current configuration** (salva su chiave USB/File system la configurazione di rete corrente o la configurazione di reset)
- **Launch web browser** (Lancia un web browser collegandosi al TPAC via LAN utilizzando l'indirizzo specificato nella sezione **Net configuration**)

Nello stesso riquadro è possibile impostare la configurazione LAN per l'hardware selezionato ed eventualmente selezionare il flag per il reset del sistema nel caso si presenti la necessità di ripristinare le impostazioni iniziali.

L'impostazione dell'indirizzo di rete per il TPAC è obbligatoria in quanto tutti i software di programmazione impiegati comunicano con il dispositivo attraverso la rete LAN. Se nella rete locale è presente un server DHCP che assegna automaticamente l'indirizzo IP al TPAC, scegliere DHCP.

Importante: se nella rete è presente un server DHCP, è necessario che l'amministratore di rete comunichi l'indirizzo IP assegnato al TPAC, altrimenti non sarà possibile utilizzarlo.
Se non è presente un server DHCP o si vuole assegnare l'indirizzo IP al TPAC in modo statico, scegliere l'opzione STATIC, quindi inserire l'indirizzo IP da impostare.



Per il ripristino delle impostazioni iniziali vedere il capitolo "Reset del sistema".

Le impostazione di rete o la configurazione di reset possono essere salvati su un dispositivo USB o sul file system tramite il tasto **"Export current configuration"** che creerà rispettivamente un file "net.conf" contenente la configurazione per la rete LAN o "reset.conf" per le impostazioni di reset.

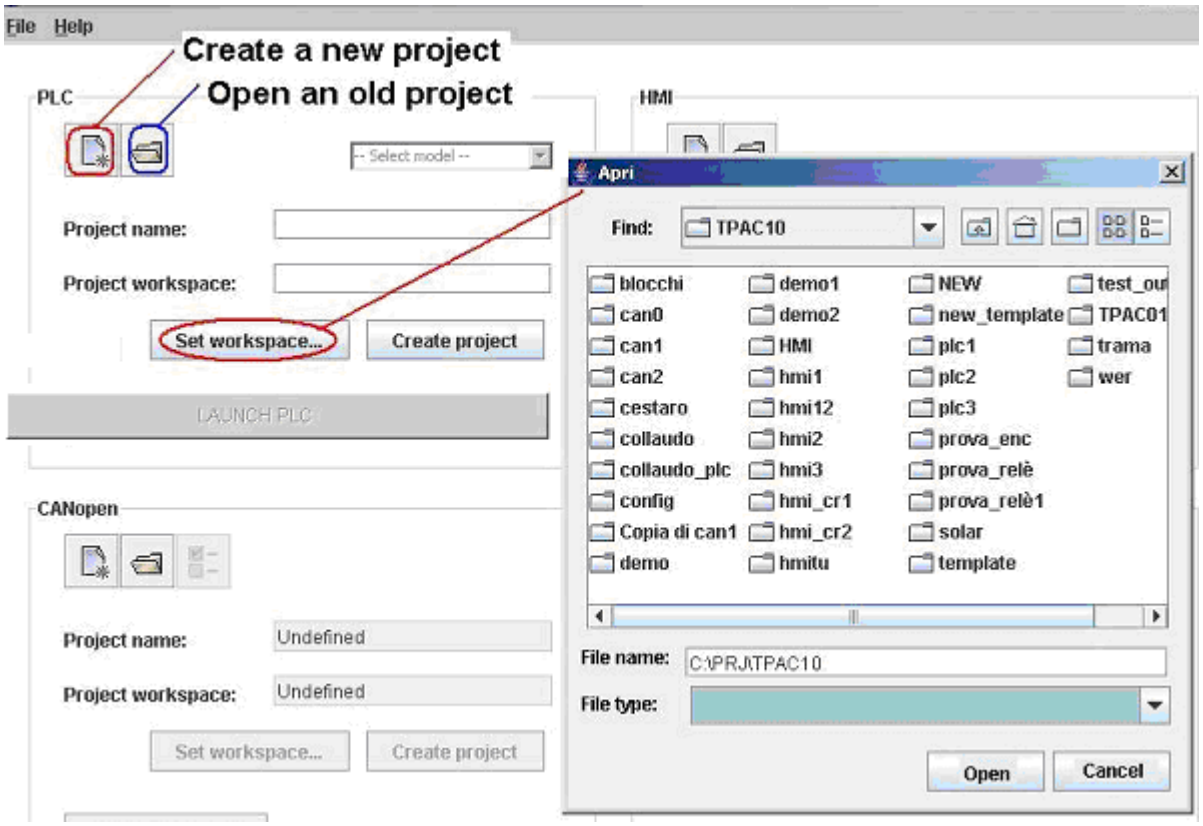
Dopo aver salvato le impostazioni relative alla rete LAN su dispositivo USB, per configurare il TPAC, è sufficiente accenderlo inserendo il dispositivo USB nella relativa nella porta. Il TPAC caricherà i parametri di configurazione che saranno attivi dalla successiva accensione. Ora il dispositivo è pronto per essere programmato.

5.0 SVILUPPARE UN SEMPLICE PROGRAMMA

TPAC può essere programmato nei cinque linguaggi dello standard IEC 61131-3. In questo tutorial, si svilupperà un semplice programma con alcuni di questi linguaggi.
Prima di sviluppare un qualsiasi programma si deve definire il progetto.

5.1 CREARE UN NUOVO PROGETTO

Per partire con un nuovo progetto, si deve utilizzare la dashboard come descritto al paragrafo "Configurazione del TPAC". Dalla finestra principale, usare la sezione PLC selezionando il tipo di hardware cui il progetto fa riferimento.. Premendo il pulsante New project (quello cerchiato in rosso) e successivamente il pulsante Set workspace, apparirà una finestra per selezionare la cartella nella quale si desidera inserire il nuovo progetto.

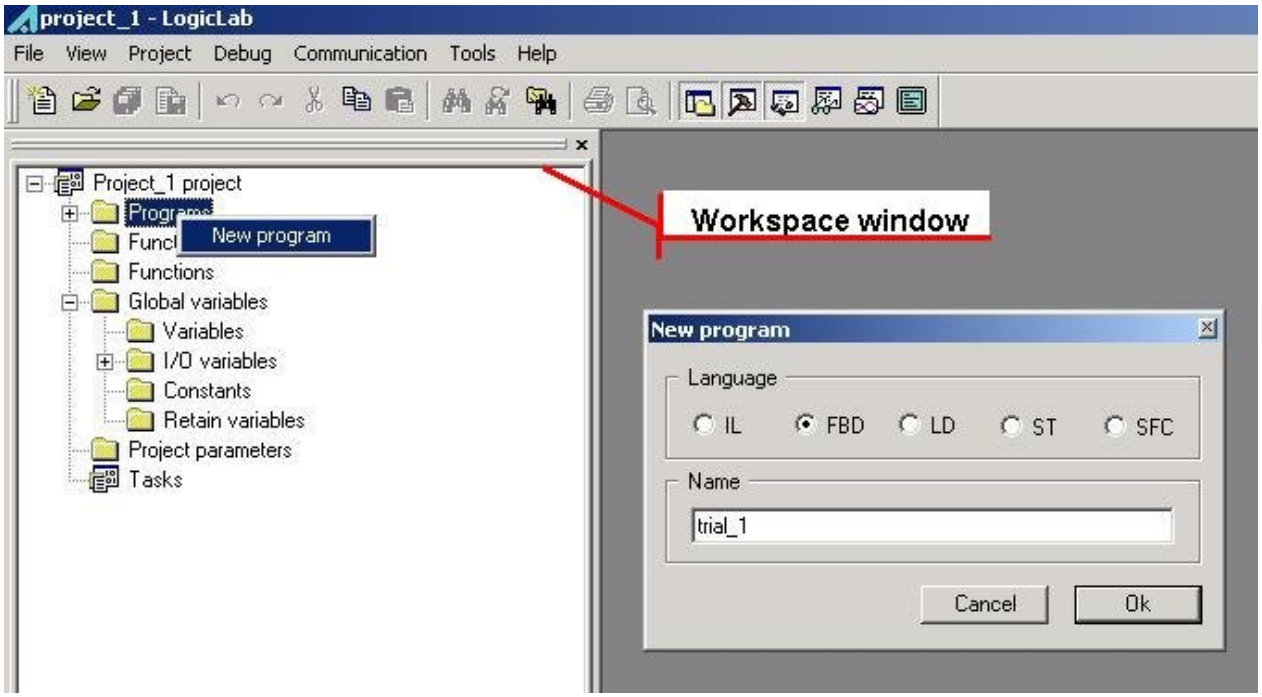


Selezionata la cartella premere Open, specificare il nome del progetto da creare digitando nel campo **Project name** il nome del progetto. Premere il pulsante **Create project** per completare la creazione del progetto.

Per iniziare la programmazione vera e propria premere il pulsante **Avvia PLC**: l'ambiente di programmazione PLC sarà attivato.

5.2 PASSI COMUNI PER CREARE UN NUOVO PROGRAMMA

Per creare un nuovo programma selezionare Programs nella finestra Workspace window (parte sinistra della finestra principale), quindi premere il tasto destro del mouse per visualizzare il tag New program . Cliccandolo, sarà disponibile la finestra New program .



Questa finestra permette di selezionare quale linguaggio di programmazione utilizzare per il programma PLC che si intende realizzare. Si inizierà usando un linguaggio grafico, quindi uno testuale.
Scegliere come linguaggio del nuovo programma FDB (Function Block Diagram), inserire nel campo Name il nome che si vuole assegnare al programma e premere OK.

5.3 DESCRIZIONE DEL PROGRAMMA

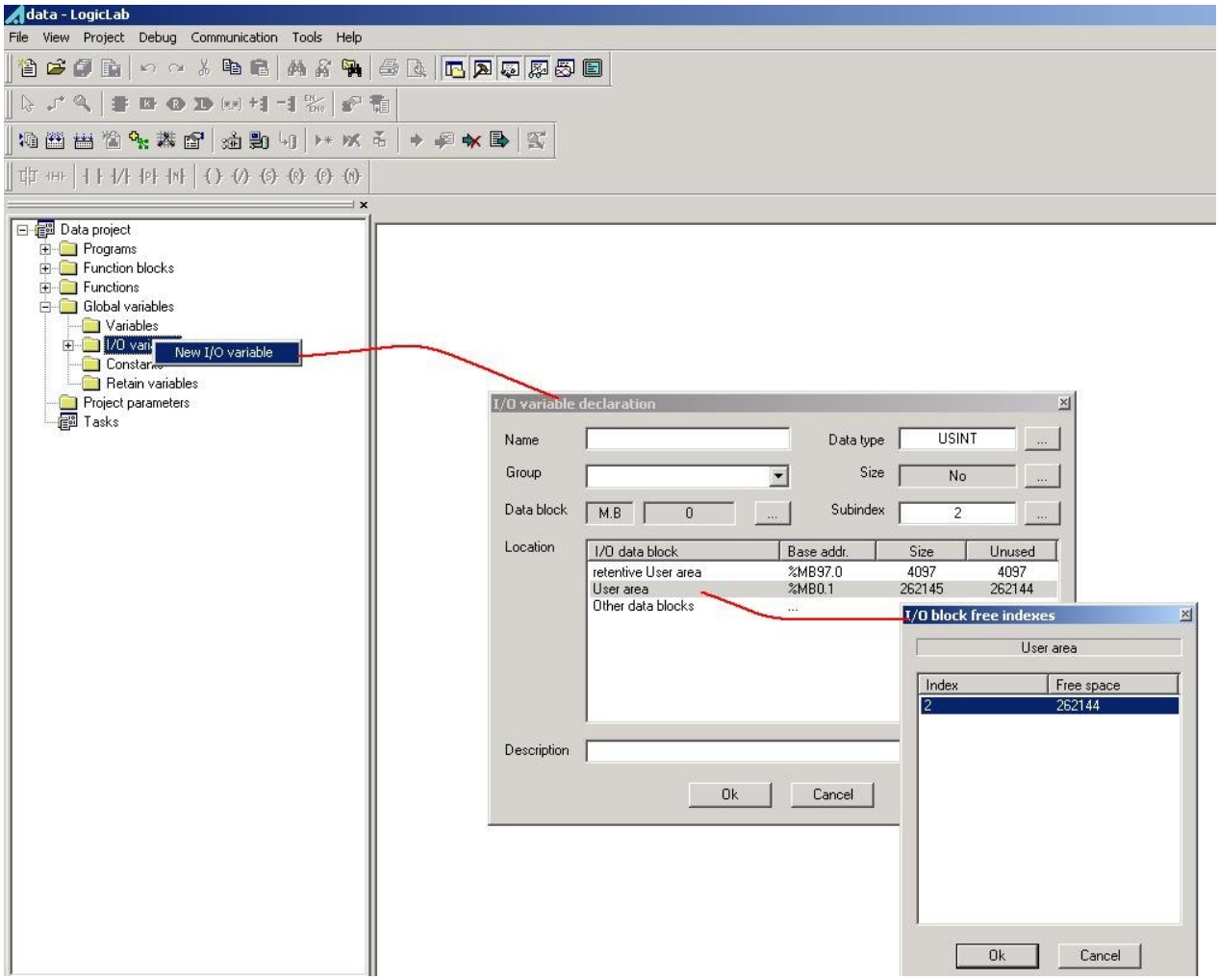
Il progetto che si svilupperà in questo tutorial utilizza un regolatore di pressione on-off. La pressione in una stanza deve rimanere compresa tra due soglie. Se la pressione sale oltre la soglia massima (threshold 1), la si deve ridurre azionando l'uscita output_0 aprendo la valvola di scarico. Se la pressione va al di sotto della soglia minima (threshold_2), la si deve incrementare azionando l'uscita output_1. Un ingresso digitale abilita o meno la regolazione.

5.4 INSERIMENTO DEGLI IO

Il primo passo è la definizione degli ingressi e delle uscite del PLC. Dalla descrizione del progetto si deduce che sono necessari un ingresso analogico per la misura della pressione, due uscite digitali per il comando delle valvole ed un ingresso digitale per l'abilitazione del sistema di regolazione.
Il PLC ha accesso ai pin di IO attraverso le variabili di ingresso e uscita. Queste variabili sono già definite nel progetto. Si assume di voler mappare l'uscita output_0 del progetto con l'uscita fisica del TPAC DO_1, l'uscita output_1 con l'uscita fisica DO_2, l'ingresso enable con l'ingresso fisico DI_1 e l'ingresso analogico con l'ingresso fisico ANALOG_INPUT_1.

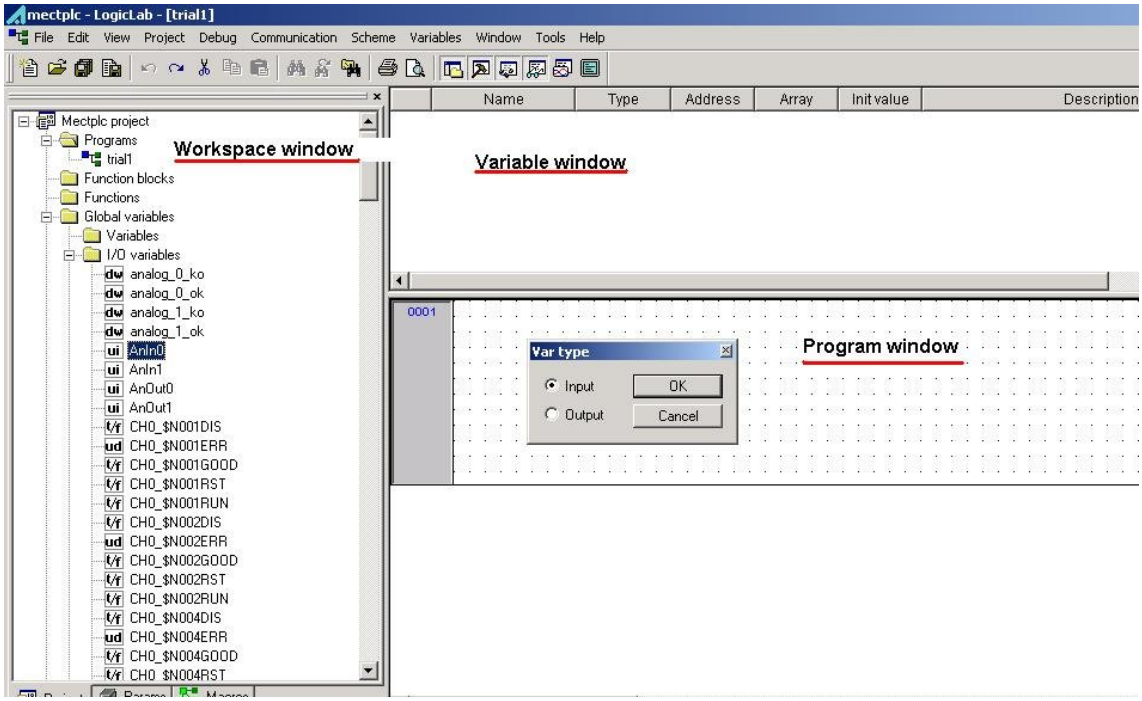
Successivamente si devono definire le variabili threshold_1 e threshold_2. Nella workspace window si selelzioni la cartella I/O variables, premendo con il tasto destro del mouse, sarà visibile il tag New I/O variable. Cliccandolo la finestra I/O variable declaration diventerà attiva.

Premendo sulla riga **User area**, si evidenzierà la finestra che indica le zone di memoria libere: premendo due volte sulla riga mostrata, automaticamente la variabile sarà assegnata a quella locazione di memoria (ciò permette di evitare sovrapposizioni delle variabili in memoria).
Inserendo nel campo Name della finestra I/O variable declaration il nome della variabile (threshold_1) e premendo OK, la variabile sarà creata e quindi utilizzabile nel programma. Seguire lo stesso procedimento per la variabile threshold_2.

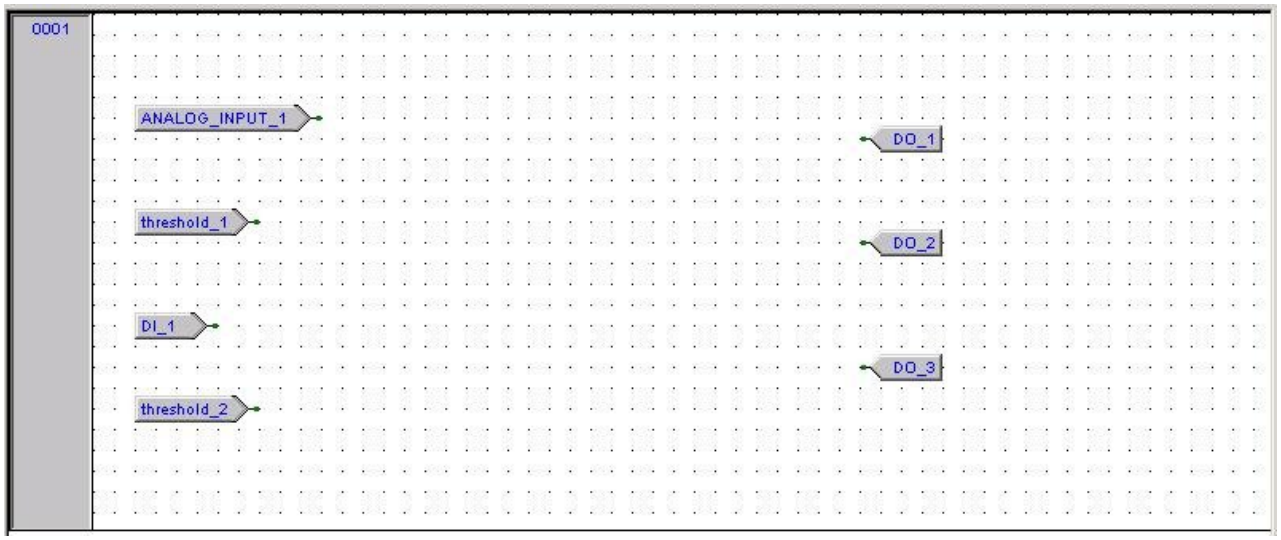


5.5 IL PROGRAMMA FBD

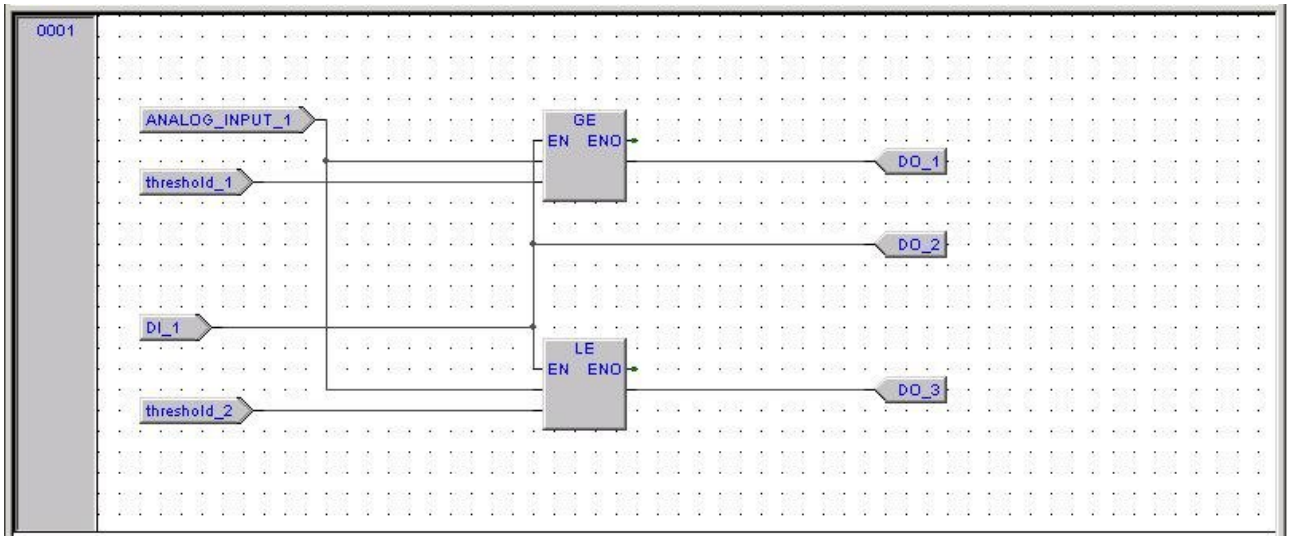
Nella workspace window espandere l'albero I/O variables. In questo albero selezionare quale ingresso o uscita utilizzare, quindi spostarlo nella finestra di programma semplicemente mantenendo premuto il tasto del mouse. Il software chiederà se si tratta di un input o di un output. Selezionare il tipo corretto, quindi premere OK.



Eseguire le stesse operazioni per le variabili, **threshold_1** and **threshold_2**.
Nella program window, si vedranno delle etichette corrispondenti agli ingressi e uscite selezionate.



Dopo aver posizionato gli IO e le variabili nella program window, gli stessi si devono connettere con i blocchi funzionali.
Nella library window selezionare il tab Operator and standard blocks. Se la library window non è attiva, attivarla premendo Ctrl+L, oppure selezionare View->Library nel menu della finestra principale. A questo punto si possono inserire i blocchi funzione, selezionandoli e trasportandoli nella program window ottenendo un layout come in figura. Se il blocco funzione non dovesse essere tra quelli di sistema sono disponibili altri blocchi funzione creati da MECT, che integrano quelli standard. Per accedere ad una lista di questi ulteriori blocchi funzione dalla workspace window selezionare Function blocks. Per maggiori dettagli sul funzionamento di questo blocchi funzione fare riferimento al capitolo di questo manuale: “Blocchi funzione”.



E' evidente che si devono collegare gli ingressi e le uscite con i blocchi inseriti. Questa operazione si esegue premendo nella finestra program window con il tasto destro del mouse e selezionando Connection mode dalla finestra di pop-up che apparirà. Il cursore cambia forma indicando che ora è possibile disegnare le connessioni tra i vari elementi presenti nello schema. Per far ciò, premere sul pin di connessione DI_1, quindi sul pin di connessione EN del blocco funzione GE. Un filo di connessione sarà disegnato tra i due pin. Ripetere questa operazione tra gli altri elementi del diagramma.
Terminate le connessioni si è pronti per compilare ed effettuare il download del programma sul TPAC, così com'è descritto nella sezione Scaricare un programma nel TPAC.

5.6 PROGRAMMA ST

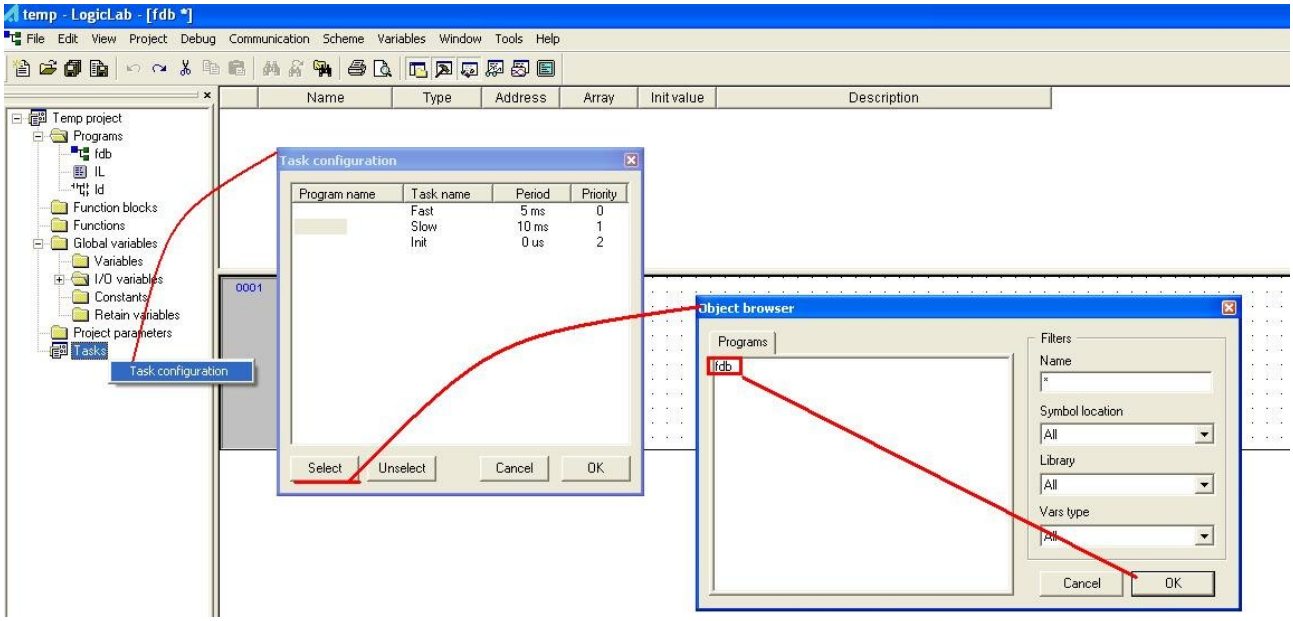
Lo stesso programma può essere implementato usando il seguente codice nel linguaggio di programmazione Structured Text. Ripetere i passi descritti nel capitolo: "Passi comuni per creare un nuovo programma" quindi selezionare ST invece di FBD.

```
IF DI_1 = 1 THEN
        DO_2 := 1;
        IF AnIn0 >= threshold_1 THEN
                DO_1 := 1;
                DO_2 := 0;
        ELSIF AnIn0 <= threshold_2 THEN
                DO_1 := 0;
                DO_2 := 1;
        END_IF;

ELSE
        DO_1 := 0;
        DO_2 := 0;
        DO_3 := 0;
END_IF;
```

5.7 COMPILARE E SCARICARE IL PROGRAMMA

Prima di poter compilare il programma lo si deve assegnare ad un task del PLC.

A questo proposito selezionare Task nella workspace window, cliccando con il tasto destro del mouse, quindi premere sul tag Task configuration che appare a video. Nella colonna Program name della finestra di pop-up Task configuration selezionare il task a cui assegnare il programma, quindi premere Select. Apparirà la finestra Object browser. Selezionare il nome del programma, quindi premere OK per convalidare la scelta e chiudere la finestra.



Una volta assegnato il programma al task, si può procedere alla compilazione e quindi al download nel TPAC.
Per compilare il progetto, selezionare Recompile all nel menu Project. L'avanzamento dello stato della compilazione e gli eventuali errori si possono visualizzare nella Output window nella parte bassa della finestra principale.

Se il programma è stato modificato o, come nel caso esaminato in cui è stato creato nuovo progetto, nella parte in basso a destra della barra di stato della finestra principale si vedrà la dicitura DIFF.CODE. Per aggiornare il codice presente sul TPAC, selezionare Download code nel menu Communication. Nella finestra Output window si visualizzerà il risultato dell'operazione.

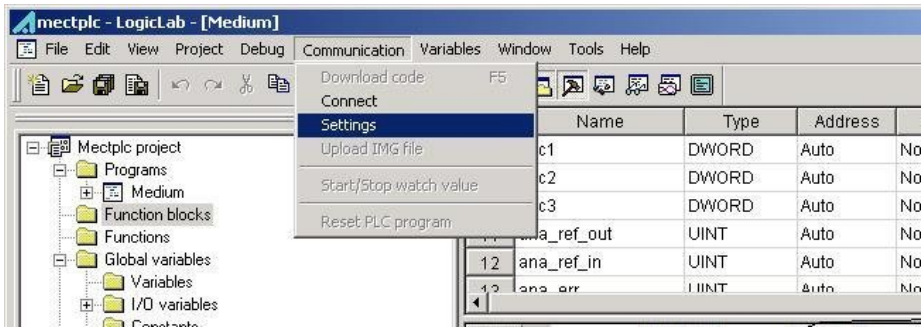
6.0 SCARICARE UN PROGRAMMA NEL TPAC

Prima di scaricare il programma nel TPAC, è necessario stabilire una connessione tra il personal computer ed il TPAC.

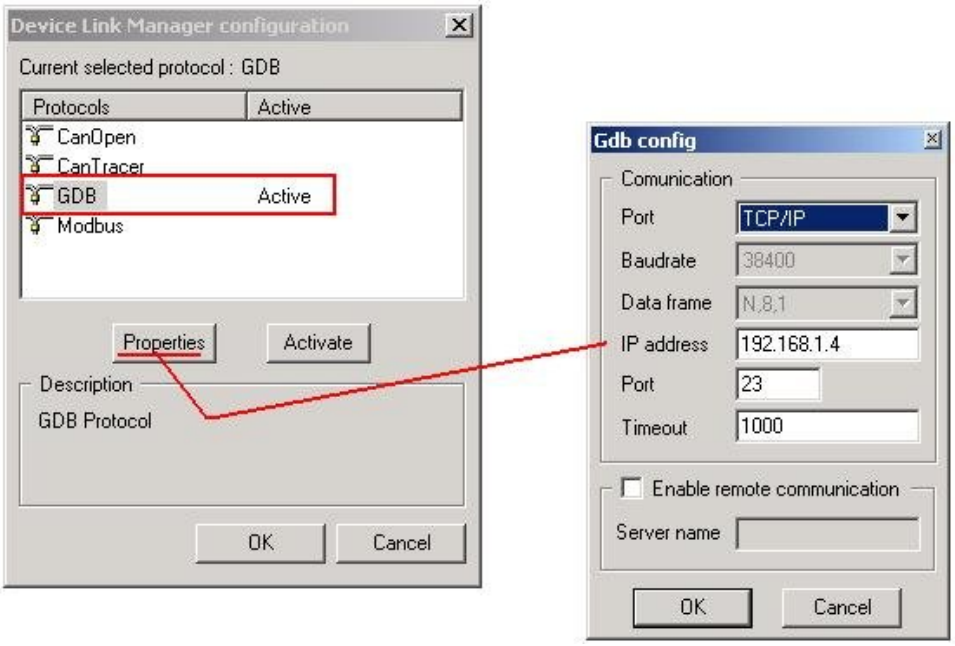
È necessario collegare il TPAC al personal computer usando la rete TCP/IP.

6.1 SCARICARE UN PROGRAMMA PLC

Per scaricare un programma è necessario:
Conoscere l'indirizzo IP che è stato assegnato al TPAC.
Impostare i parametri di comunicazione dell'ambiente di sviluppo.



Scegliere **Settings** dal menu **Communication**. Nella finestra **Device Link Manager Configuration** selezionare il protocollo GDB. Se non attivo premere il pulsate **Activate**, quindi premere il pulsante **Properties**: si visualizzerà finestra **Gdb config**. Selezionare come Port: TCP/IP ed inserire nel campo IP address l'indirizzo IP del TPAC. Impostare 5000 come numero di porta nel campo sottostante IP address e lasciare gli altri parametri invariati. Infine premere **OK**.

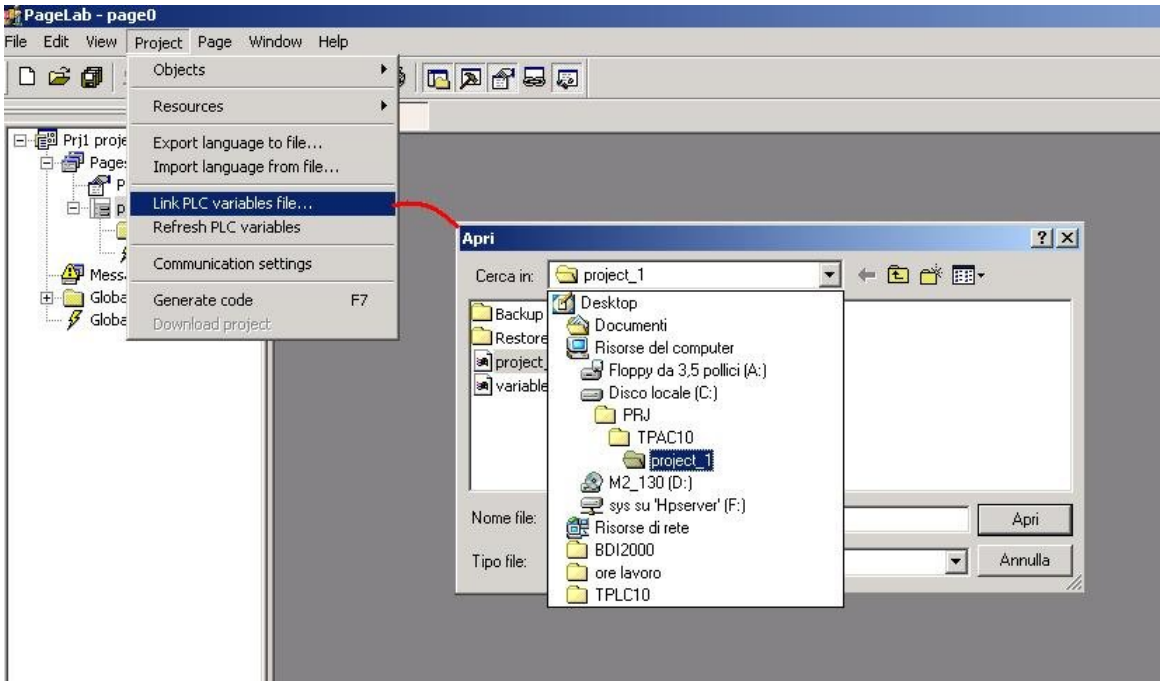


Da questo momento è possibile collegarsi al TPAC. Dal menu Communication della finestra principale, selezionare Connect per stabilire la comunicazione tra il target ed il PC. Quando si è stabilita una connessione, nella parte destra della barra di stato di LogicLab si potrà vedere:
DIFF. CODE, che indica che una connessione si è stabilita ma il codice sul TPAC è differente da quello presente sul PC.
CONNECTED che indica che una connessione si è stabilita ed il codice sul TPAC è lo stesso di quello presente sul PC
ERROR che indica un problema di connessione tra il TPAC ed il PC.

7.0 COME CREARE UN'INTERFACCIA HMI

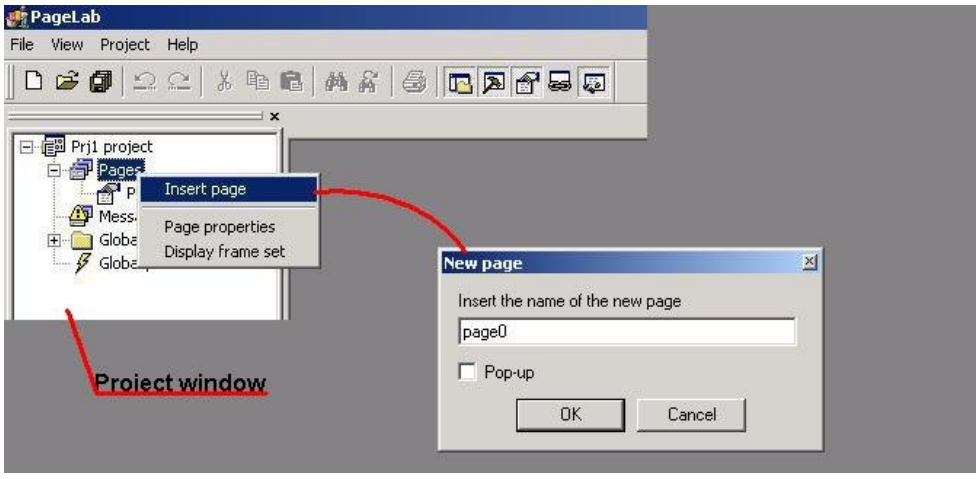
Dopo aver creato il programma PLC, è possibile sviluppare un'interfaccia HMI. Dalla dashboard si crea un nuovo progetto cliccando, nella sezione HMI, il pulsante New project; quindi si procede in modo analogo alla creazione di un progetto PLC. Per lanciare un progetto (o per aprirne uno nuovo) premere il tasto Launch HMI: si aprirà l'ambiente di sviluppo per HMI.

Importante: prima di iniziare, le variabili che abbiamo definito e utilizzato nel progetto PLC devono essere importate nell'ambiente HMI. Per fare ciò, selezionare dal menu Project, Link PLC variables file…, quindi, cercare nella cartella del progetto PLC il file .exp, selezionarlo e premere Apri. Le variabili del progetto PLC possono ora essere viste e modificate anche in ambiente HMI.



Come per il progetto PLC, si devono impostare i parametri di comunicazione con TPAC.
Selezionare il menu Project->Communication settings ed eseguire gli stessi passi effettuati per la configurazione della comunicazione con PLC.

Ora è possibile sviluppare il progetto HMI.
A questo proposito si inserisce una pagina. Nella project window, selezionare Pages e premere con il tasto destro del mouse. Cliccare sulla voce Insert page della finestra di pop-up. Nella finestra New page inserire il nome della pagina e selezionare OK.



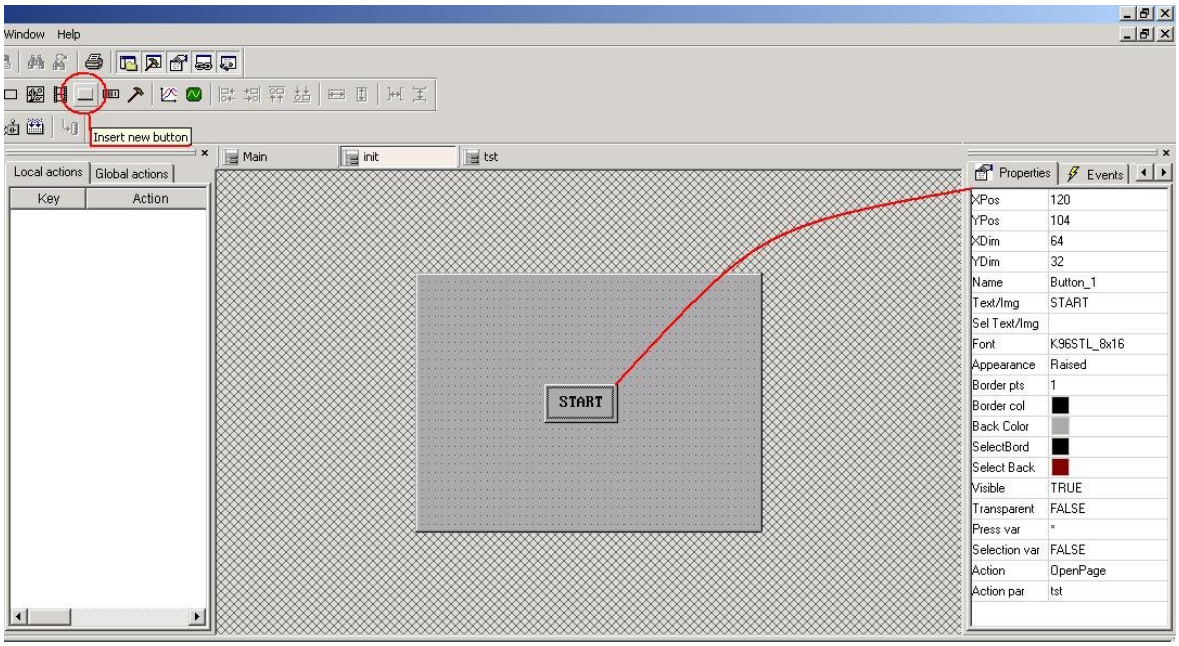
Per vedere la pagina, premere due volte sul nome della pagina nella finestra **Project**. Si visualizzerà una pagina vuota e si potrà iniziare a popolarla con gli elementi disponibili.

Per inserire un oggetto nella pagina, selezionare l'elemento da inserire scegliendolo all'interno del menu Page. Gli oggetti disponibili sono:
Static
Image
Animation
Edit
Button
Custom control

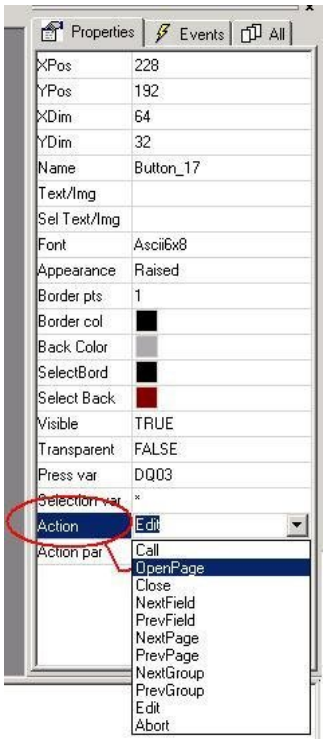
Per il progetto sviluppato in questo tutorial si definiscono due pagine: una pagina iniziale ed una pagina operativa.

La pagina iniziale ha tre elementi: due oggetti statici e un tasto e appare come nella figura seguente.

Per inserire il tasto, selezionare la voce **New Button** nel menu **Page**, quindi introdurre il tasto sulla pagina nella posizione desiderata. Nella parte destra della finestra principale sono elencate le proprietà dell'oggetto tasto. Queste proprietà sono modificabili secondo le necessità del programmatore.

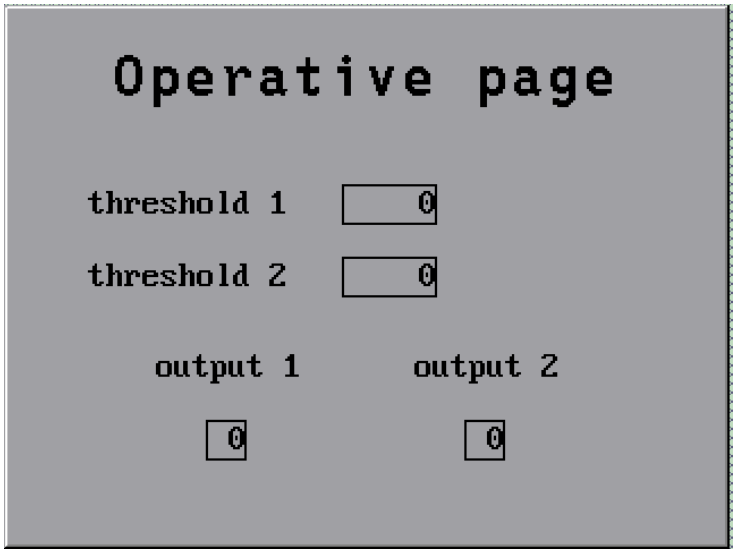


Tra le proprietà più importanti dell'elemento tasto c'è quella di azione a seguito della pressione del medesimo. Per assegnare un'azione al tasto selezionare la riga Action, quindi scegliere dal menu a tendina l'azione che si intende associare tra quelle elencate. Nell'esempio in questione, si vuole che il tasto, quando premuto, apra la pagina operativa, perciò si seleziona la voce OpenPage.

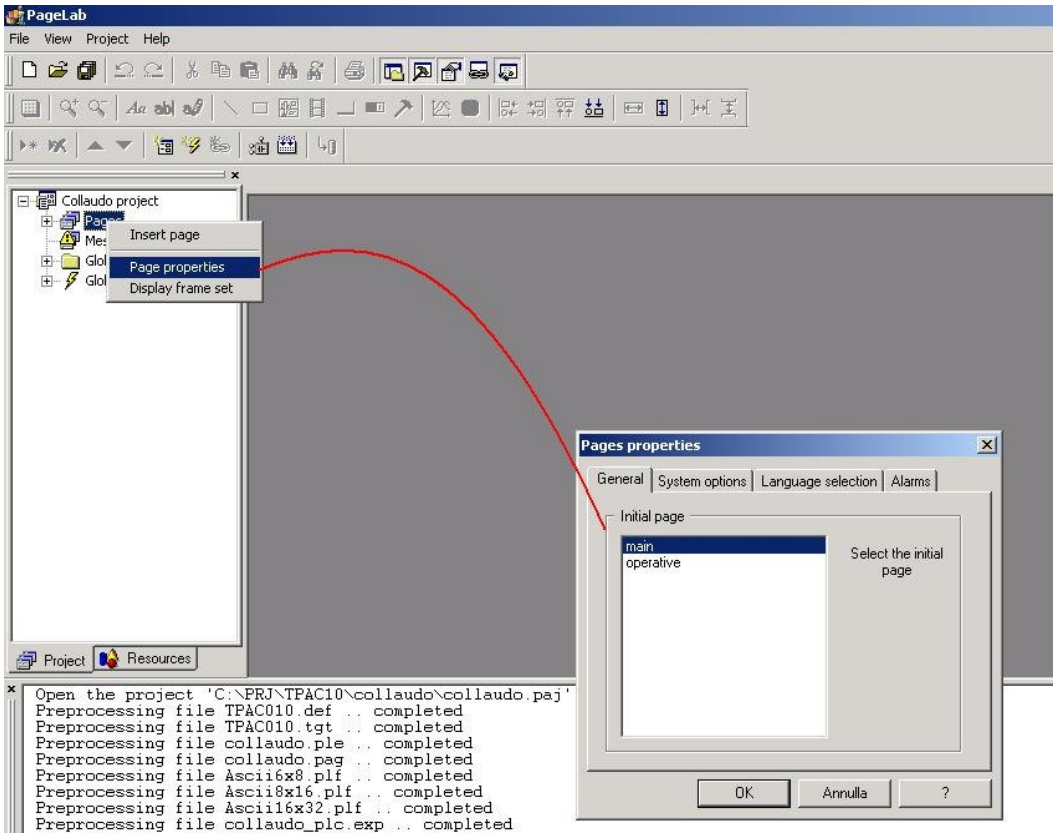


Analogamente a quanto visto con i tasti si possono inserire gli altri elementi e modificarne le proprietà.

La pagina operativa ha più oggetti rispetto alla pagina iniziale. Si possono infatti notare cinque oggetti statici (le scritte) e quattro campi di input-output. Le proprietà di ciascun oggetto si possono variare seguendo le modalità esemplificate per l'oggetto tasto descritto in precedenza.



Prima di effettuare la compilazione del progetto è necessario indicare quale è la prima pagina dalla quale iniziare la visualizzazione. Per far ciò, nella project window, selezionare Pages quindi premere con il tasto destro del mouse. Cliccare sulla voce Page properties della finestra di pop-up. Nella finestra Page properties selezionare General, scegliere dall'elenco delle pagine create quella che si desidera sia la pagina iniziale e premere OK.



Dopo aver definito la pagina iniziale è possibile compilare il progetto. Dal menu Project selezionare Generate code.

Prima di scaricare il progetto sul TPAC è necessario stabilire, come già effettuato in ambiente PLC, la connessione via LAN con il personal computer.

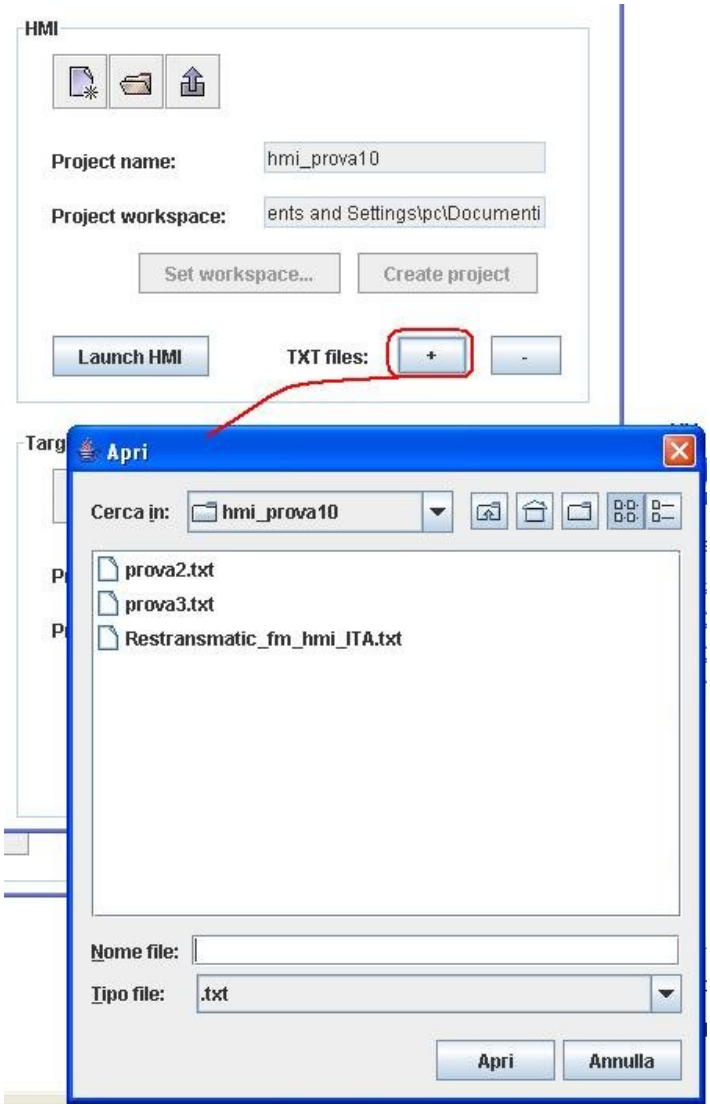
Per scaricare il codice compilato sul TPAC premere Download project dal menu Project.

7.1 VISUALIZZAZIONE DI FILE DI TESTO

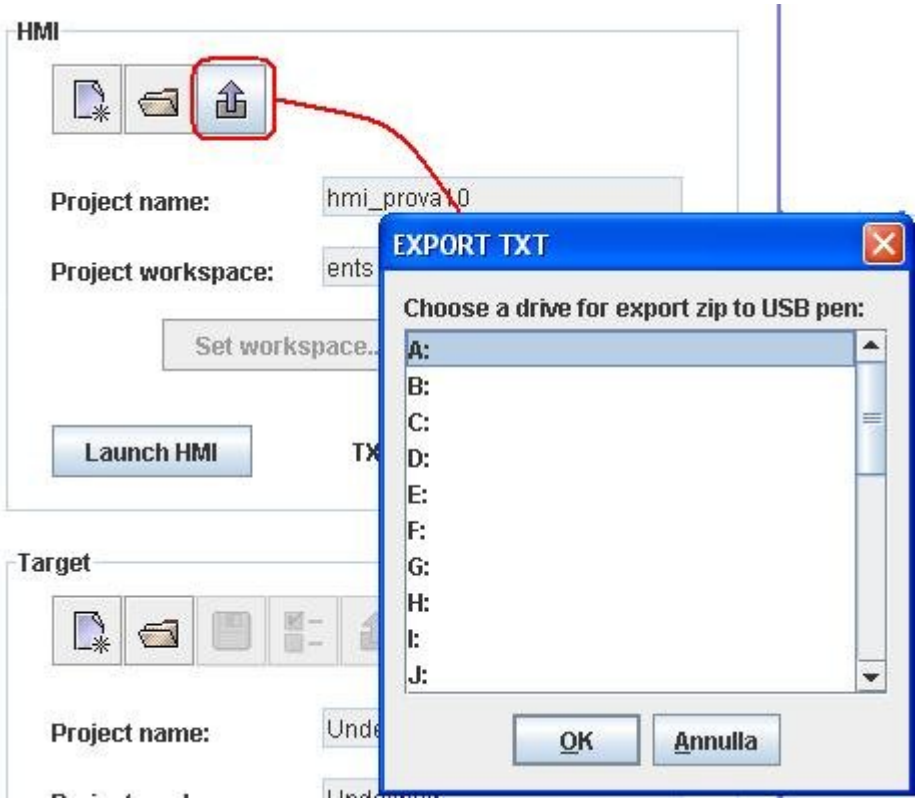
In un progetto HMI può essere necessario visualizzare dei testi statici come ad esempio istruzioni d'uso. Inserire questi testi in variabili ritentive impegnerebbe questa risorsa per uno scopo differente da quello per cui sono state create; il programmatore perciò ha a disposizione la funzione HMItextLoad() per mostrare a video il contenuto di uno o più file presenti sulla flash del TPAC.

7.1.1 inserimento dei file di testo in un progetto HMI

Se il programmatore vuole inserire un file di testo all'interno di un progetto HMI, premendo il tasto + nella sezione HMI della dashboard si attiverà una finestra nella quale si possono selezionare i file di testo da inserire nel progetto. L'eliminazione di un file dall'archivio avviene invece premendo il tasto - e selezionando il file da eliminare.



Dopo aver inserito tutti i file necessari, premendo il tasto  sarà disponibile la seguente finestra di scelta del drive in cui è mappata la penna USB dove salvare l'archivio creato.



Selezionato il drive e premendo OK, il file di archivio sarà salvato sulla penna USB. Inserendola penna USB nel TPAC e riavviando il dispositivo, l'archivio viene salvato sulla flash ed è quindi utilizzabile dal programmatore.

7.1.2 Utilizzo dei file di testo in HMI

Per poter utilizzare i file di testo inseriti nell'archivio, è disponibile la funzione HMItestLoad('nome_file'). Tale funzione cerca se esiste un file con il nome specificato e La funzione restituisce un codice d'errore: se il codice è zero, la funzione è andata a buon fine ed il puntatore al file è disponibile nella variabile HMI_LOAD.
Si dettaglia un esempio di come utilizzare la funzione HMItestLoad:

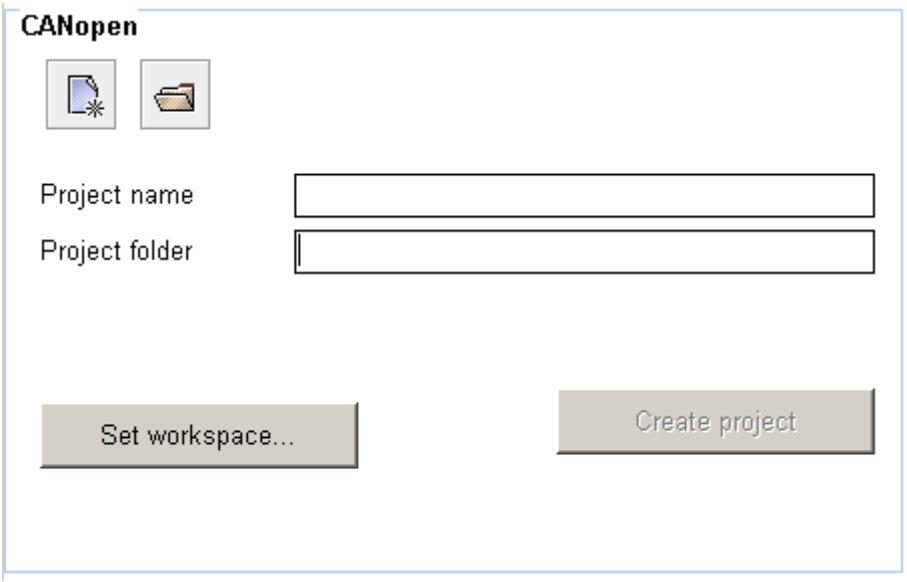
```
res:=HMItextLoad('istruzioni.txt');

IF res = 0 THEN
        output:=FILE_READ;
ELSE
        output:='File non presente';
END_IF;
```

La funzione HMItestLoad ricerca il file istruzioni.txt, se esiste, res è zero, quindi si assegna alla variabile output, dichiarata come stringa di 1000 caratteri, il puntatore al file che la funzione restituisce. Assegnando la variabile output ad una casella di testo HMI il file è mostrato a video.

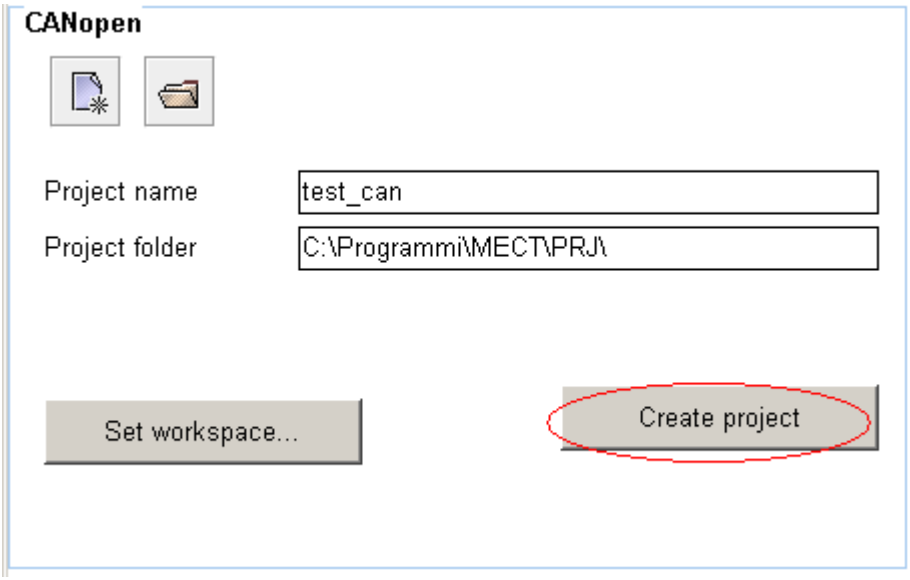
8.0 COME CONFIGURARE UNA RETE CANOPEN

Per utilizzare nodi CANopen in una rete, è necessario effettuare una configurazione di ciascun nodo e impostare i parametri di comunicazione con il master. Per poter effettuare queste configurazioni è necessario utilizzare un software applicativo che aiuti l'utente a definire le impostazioni necessarie.

Il software applicativo è strettamente legato al master della rete CANopen ed è generalmente fornito dal produttore del master.
MECT fornisce il software di configurazione di rete CANopen (**Can Builder**) di cui i componenti TPAC sono i master.
Si accede al configuratore attraverso l'interfaccia della dashboard aprendo o creando un progetto CANopen nella relativa sezione della Dashboard.



Cliccando sull'icona Nuovo Progetto  tramite il tasto "Set workspace" selezionare la directory di lavoro per il progetto e definire il Project Folder. Attribuire un nome al progetto e cliccare il tasto Create Project che avvia l'esecuzione del **Can Builder**.



La schermata iniziale del **Can Builder** è separata in tre sezioni:

- Elenco file EDS
- Nodi CANopen componenti la rete
- Impostazione dei parametri di rete

8.1 Elenco file EDS

Nella parte sinistra dello schermo si trova l'elenco dei file EDS utilizzabili per la costruzione della rete CANopen

8.2 Inserimento di un nodo nella rete

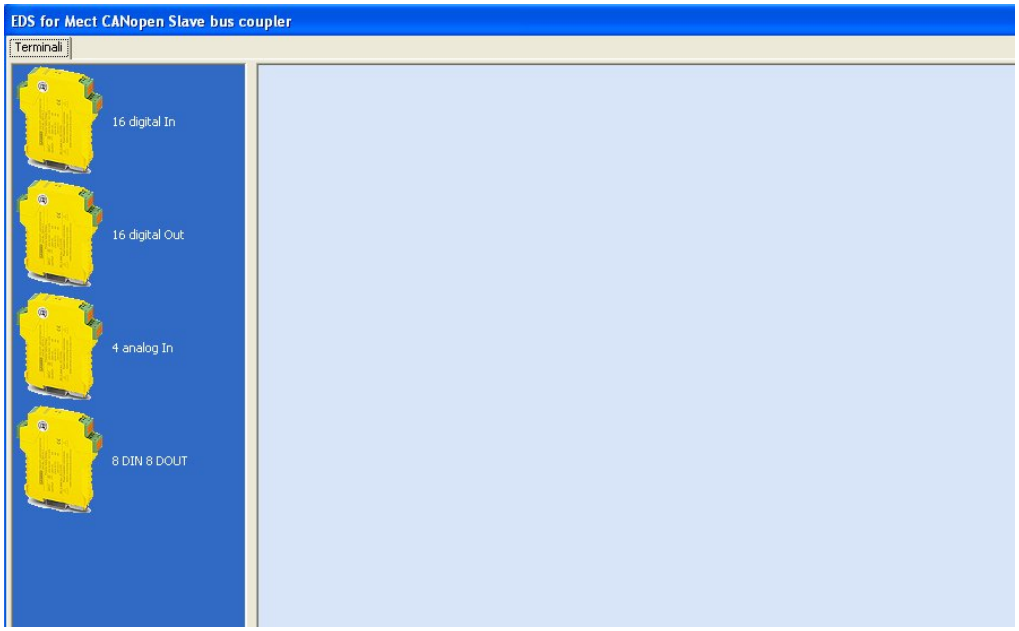
Per inserire un nodo nella rete CANopen che si desidera creare, è sufficiente selezionarlo dall'elenco dei file eds nella parte sinistra della pagina e fare un doppio click. Il nodo selezionato verrà inserito nella parte centrale della pagina e gli verrà assegnato di default un ID di rete: il primo ID disponibile.

8.3 Configurazione nodo

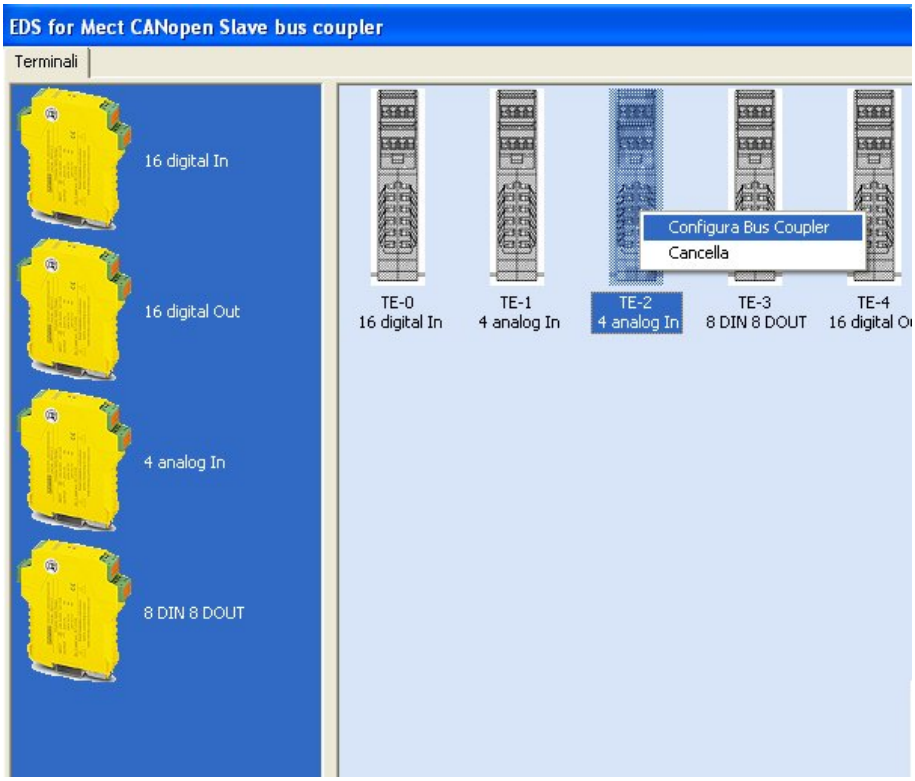
Una volta inserito un nodo, affinché questo possa comunicare con il master è necessario creare i file di configurazione con l'impostazione delle variabili e dei parametri.
Il CanBuilder permette la configurazione di un nodo CanOpen generico, per i nodi CanOpen MECT è disponibile una configurazione grafica che semplifica l'attività di impostazione dei parametri del nodo.Questo paragrafo descrive la configurazione di un nodo CanOpen MECT e di seguito la configurazione manuale generica di un nodo CanOpen.

8.3.1 Nodo MECT

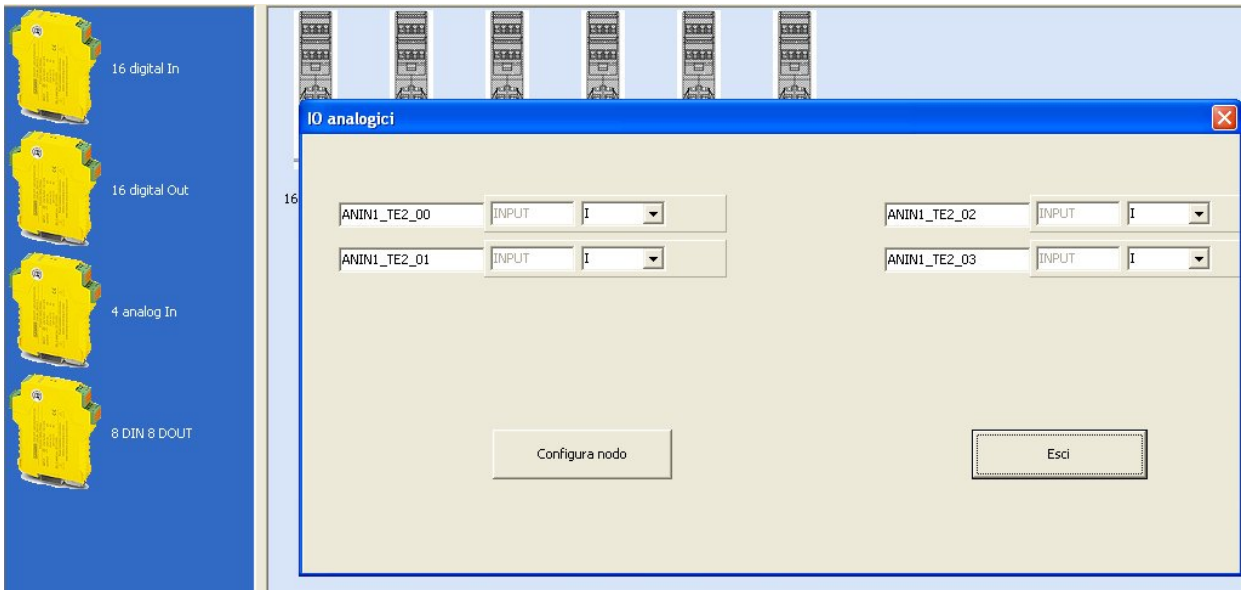
Se si vuole costruire una rete CANopen contenente il modulo MECT MPNC, si procede inserendo nella finestra di rete una o più istanze del modulo MPNC010, quindi facendo un doppio click sull'istanza che si desidera configurare appare la seguente finestra



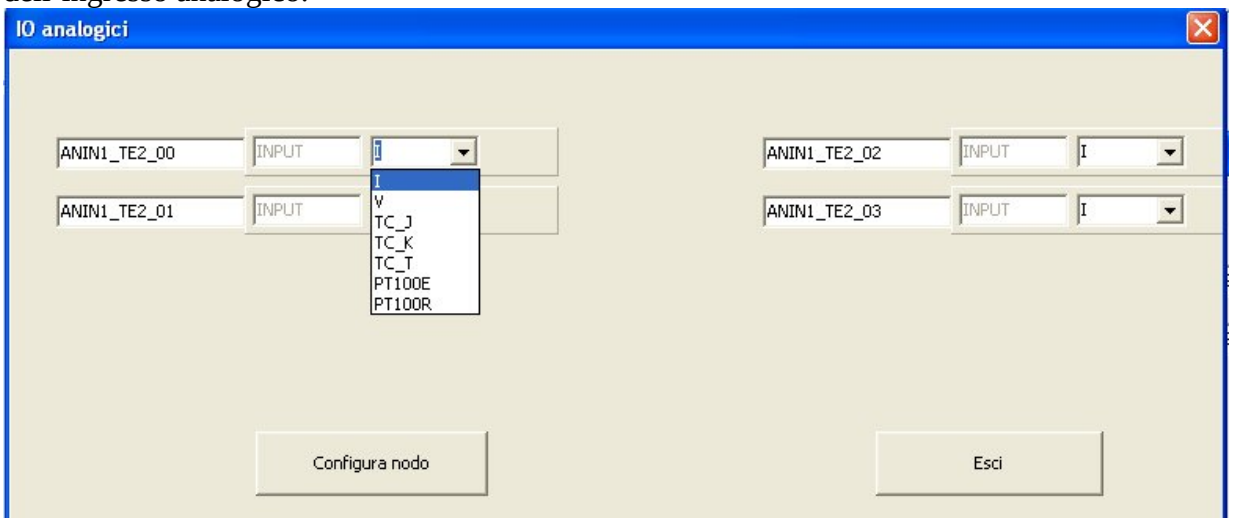
La finestra è divisa in due parti: la parte sinistra contiene l'elenco dei terminali MECT disponibili, nella parte destra invece si inseriranno le istanze dei terminali che comporranno il nodo.
L'utente si costruirà il nodo trascinando nella parte destra della finestra i terminali che vuole inserire. Ad ogni terminale viene assegnato automaticamente un nome che lo identificherà nel nodo. La sequenza di terminali creata dovrà essere rispettata anche nell'assemblaggio fisico del nodo.



I canali dei terminali analogici hanno la possibilità di essere configurati individualmente. Selezionando un terminale analogico e premendo con i tasto destro del mouse appare il menu di popup. Selezionando configura bus coupler appare la seguente finestra nella quale sono elencati i canali componenti il terminale.



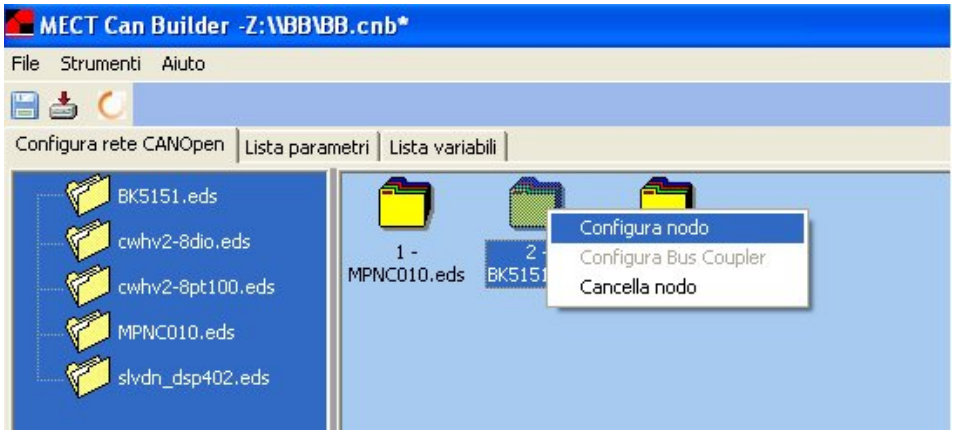
Selezionando la box alla destra del canale viene mostrato l'elenco delle possibili configurazioni dell'ingresso analogico.



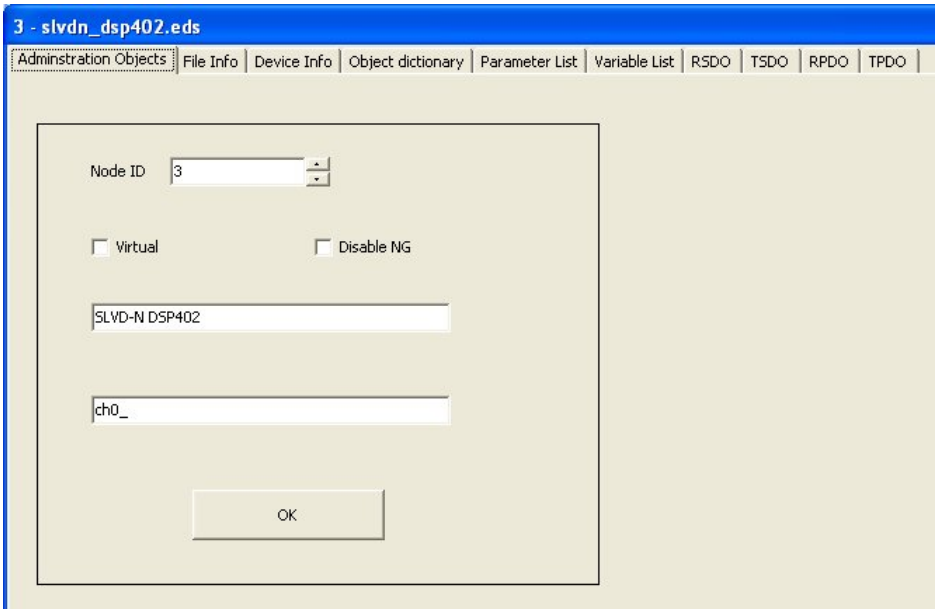
Una volta impostate le configurazioni di ciascun canale è sufficiente premere Configura nodo per salvare le impostazioni. Premendo Esci il terminale manterrà le configurazioni precedenti.
Inserite ed effettuate le configurazioni di tutti i terminali, chiudendo la finestra si torna alla pagina principale del configuratore MECT Builder.
Terminate le configurazioni di tutti i terminali , è necessario inviare i file di configurazione e le variabili al PLC. Per far ciò si procede come descritto nel paragrafo 8.6.1.2.

8.3.2 Configurazione manuale nodi CanOPEN

Selezionando con il mouse il nodo che si intende configurare e premendo il tasto destro appare il menu mostrato in figura.



Selezionando l'opzione **Configura nodo** si apre la finestra seguente, nella quale si possono evidenziare dieci sezioni



8.3.2.1 Administration Objects

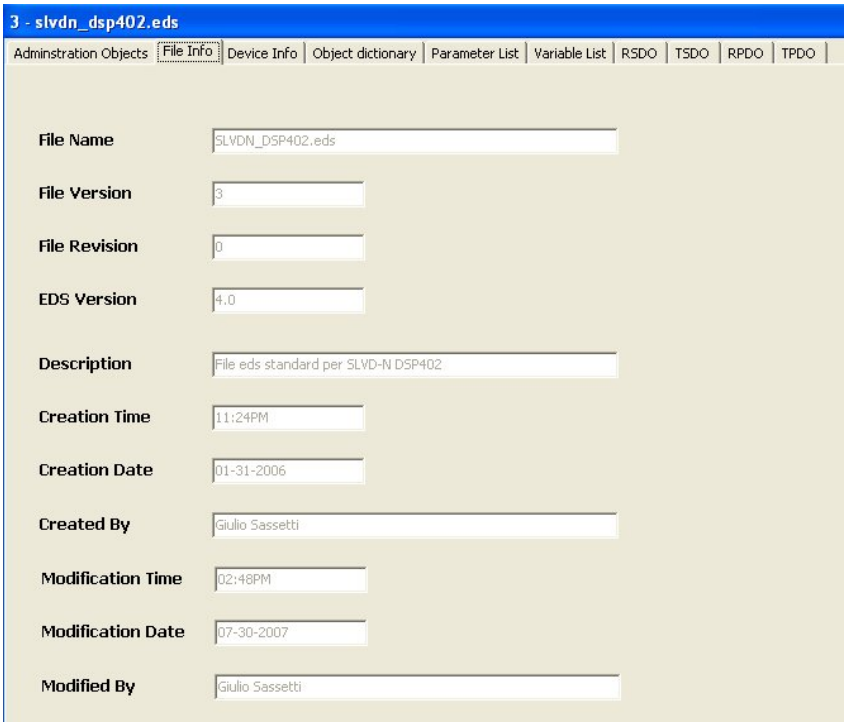
Nella prima sezione si impostano e si visualizzano alcuni parametri

- La modifica dell'ID di rete del nodo selezionato
- la disabilitazione del **Node Guarding**
- la virtualizzazione del nodo
- il nome del nodo come appare nel file EDS
- il prefisso da dare alle variabili di rete che verranno esportate nel PLC

Le modifiche effettuate avranno effetto solo se si preme il tasto **OK**

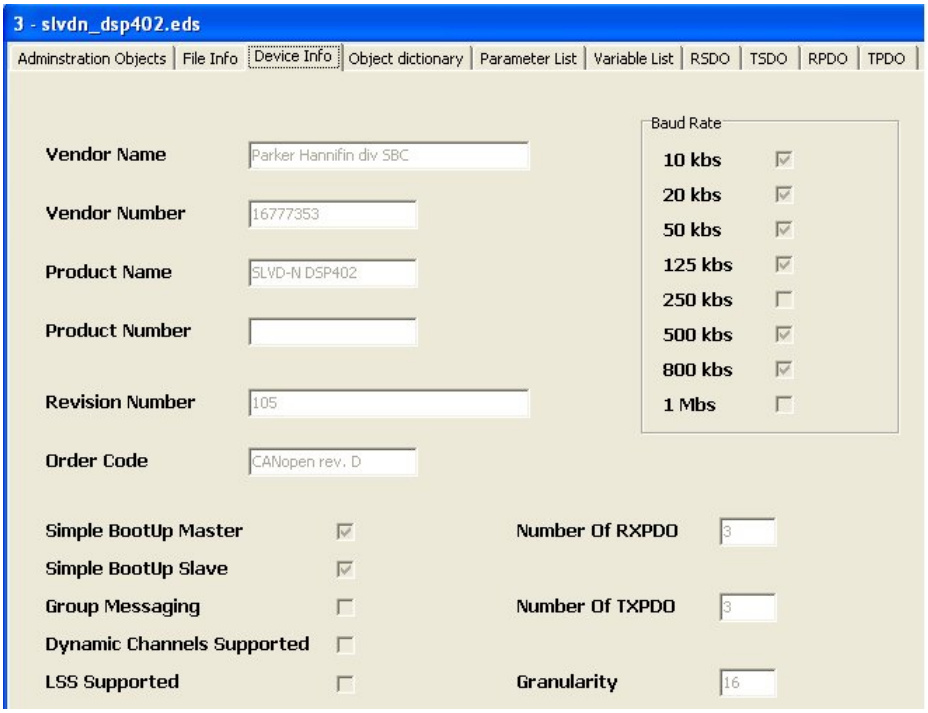
8.3.2.2 File info

La seconda sezione mostra i dati contenuti nel file EDS associato al nodo. I dati mostrati non sono modificabili



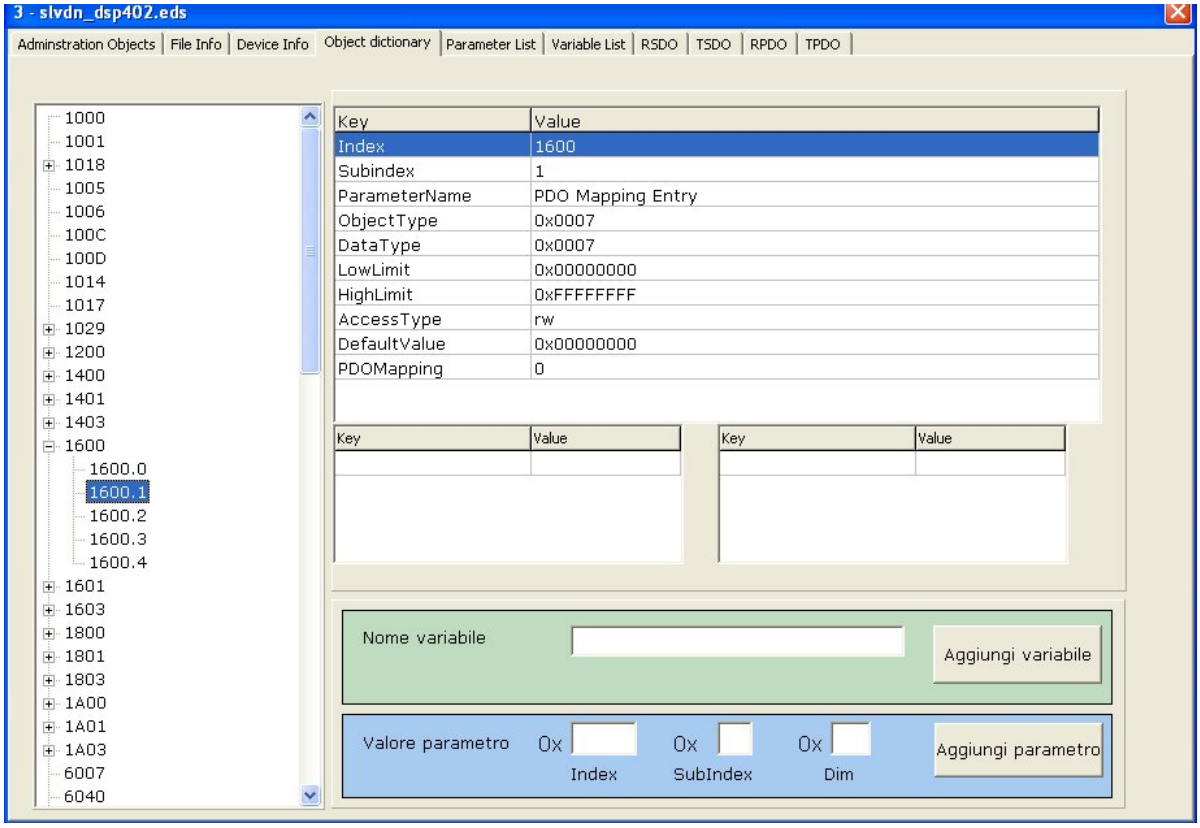
8.3.2.3 Device info

La terza sezione mostra i dati relativi alle caratteristiche del nodo selezionato, come il numero di PDO, i baudrate disponibili ecc.



8.3.2.4 Object dictionary

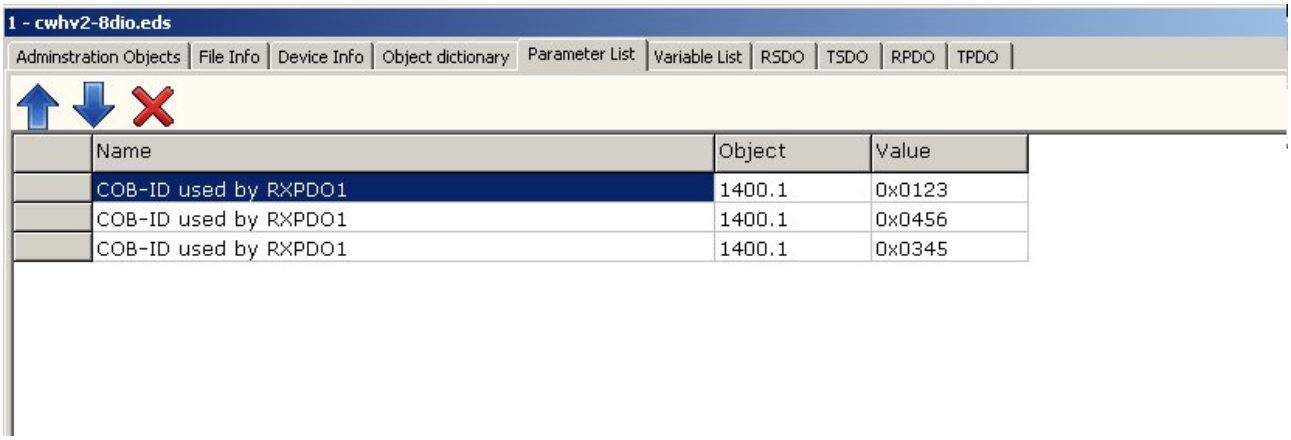
Nella sezione che contiene la lista del dizionario degli oggetti si impostano i parametri e le variabili relative al nodo.



Come si nota dalla figura, la finestra è divisa in più sezioni, la parte sinistra contiene l'elenco degli oggetti presenti nel nodo da configurare. Selezionando un oggetto, vengono mostrate, nella parte superiore della finestra, le sue proprietà.
Ad ogni oggetto con l'accesso in scrittura (w) è possibile inserire un parametro che verrà inviato dal master in fase di configurazione, è inoltre possibile associare ad ogni oggetto una variabile se si vuole accedere all'oggetto tramite PLC.
Per alcuni oggetti, come per esempio quelli per attivare i PDO, è necessario inserire più di un parametro i quali verranno inviati in sequenza verso il nodo. Si vedrà in seguito la procedura per modificare l'ordine di invio dei parametri.
Agli oggetti di tipo read only (ro) è possibile solo associare una variabile e quindi richiedere il valore dell'oggetto attraverso il PLC.
Per cancellare i parametri o le variabili è sufficiente posizionarsi con il mouse sul valore da eliminare e premere il tasto destro, una menu di pop up apparirà con l'opzione di cancellazione del parametro. Gli oggetti da modificare sono strettamente legati al nodo e all'applicazione che si vuole realizzare.
Per poter utilizzare un servizio SDO da PLC, è necessario che nel programma Can Builder si assegni un nome all'oggetto del dizionario al quale si intende accedere.

8.3.2.5 Parameter list

I parametri inseriti negli oggetti selezionati precedentemente sono raccolti in questa sezione come mostrato in figura.

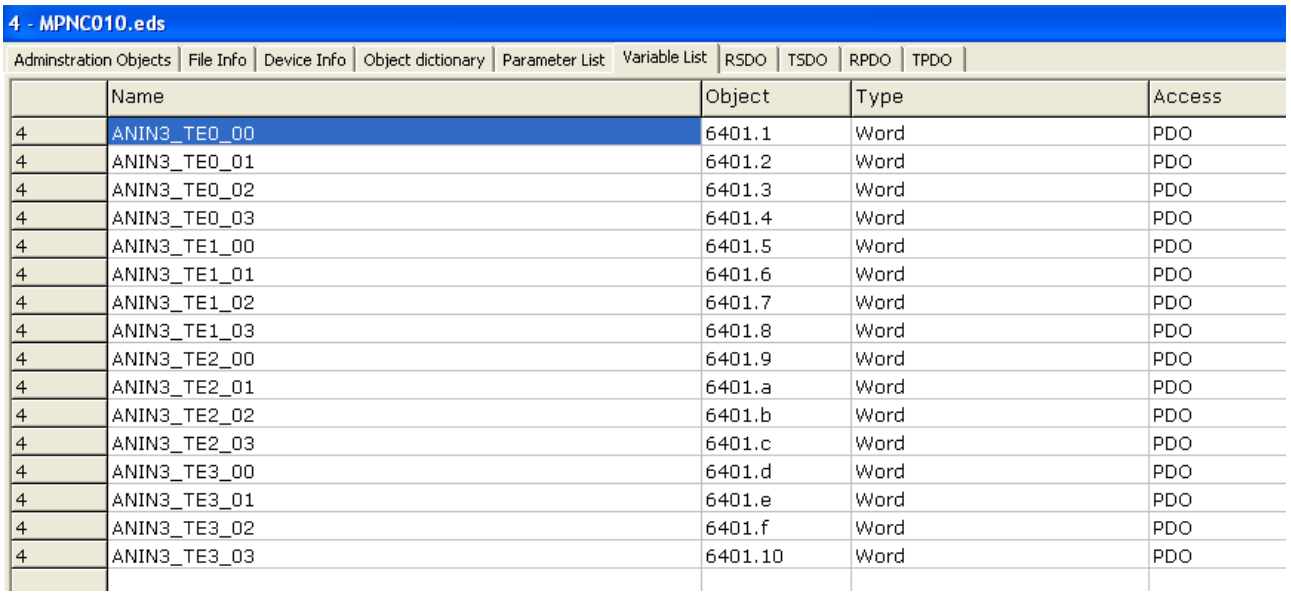


	Name	Object	Value
	COB-ID used by RXPDO1	1400.1	0x0123
	COB-ID used by RXPDO1	1400.1	0x0456
	COB-ID used by RXPDO1	1400.1	0x0345

I parametri sono inviati dal master al nodo in fase di inizializzazione. La sequenza con la quale vengono inviati è indicata dalla lista. In alcuni casi però è necessario modificare l'ordine di invio dei parametri, per far ciò è necessario spostare verso l'alto o verso il basso i parametri nella lista. Per spostare un parametro è sufficiente posizionarsi su di esso, e agendo su una delle frecce disegnante nella parte superiore della finestra ne si modifica la posizione nella lista.

8.3.2.6 Variable list

La lista seguente elenca le variabili create nella sezione dell'object dictionary e che saranno accessibili dall'utente attraverso il PLC. La lista delle variabili non è modificabile.



	Name	Object	Type	Access
4	ANIN3_TE0_00	6401.1	Word	PDO
4	ANIN3_TE0_01	6401.2	Word	PDO
4	ANIN3_TE0_02	6401.3	Word	PDO
4	ANIN3_TE0_03	6401.4	Word	PDO
4	ANIN3_TE1_00	6401.5	Word	PDO
4	ANIN3_TE1_01	6401.6	Word	PDO
4	ANIN3_TE1_02	6401.7	Word	PDO
4	ANIN3_TE1_03	6401.8	Word	PDO
4	ANIN3_TE2_00	6401.9	Word	PDO
4	ANIN3_TE2_01	6401.a	Word	PDO
4	ANIN3_TE2_02	6401.b	Word	PDO
4	ANIN3_TE2_03	6401.c	Word	PDO
4	ANIN3_TE3_00	6401.d	Word	PDO
4	ANIN3_TE3_01	6401.e	Word	PDO
4	ANIN3_TE3_02	6401.f	Word	PDO
4	ANIN3_TE3_03	6401.10	Word	PDO

8.3.2.7 Liste SDO e PDO

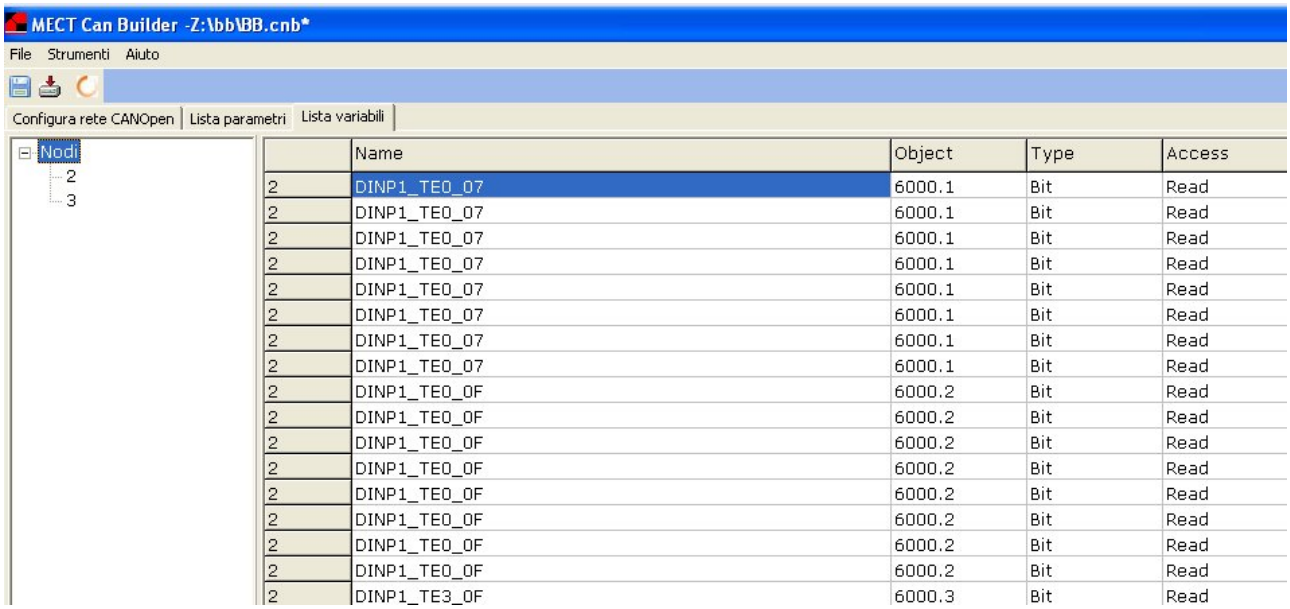
Nelle sezioni RSDO, TSDO, RPDO, TPDO sono presenti delle liste che elencano rispettivamente:

- SDO in lettura
- SDO in scrittura
- PDO in lettura
- PDO in scrittura.

Le liste sono una rappresentazione differente del dizionario degli oggetti. Queste liste sono in sola lettura.

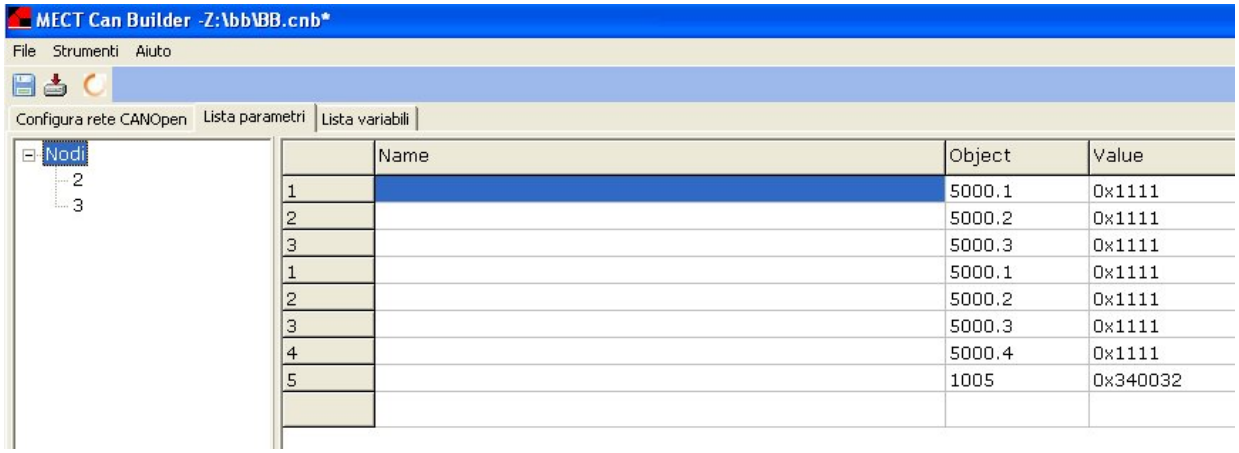
8.4 Liste degli oggetti

Nelle precedenti sezioni si è visto che ad alcuni oggetti sono stati associati dei parametri o delle variabili, e che si sono create le liste corrispondenti. Premendo sul TAG **Lista variabili** della finestra principale, viene mostrato l'elenco delle variabili associate a tutti i nodi della rete.



	Name	Object	Type	Access
2	DINP1_TE0_07	6000.1	Bit	Read
2	DINP1_TE0_07	6000.1	Bit	Read
2	DINP1_TE0_07	6000.1	Bit	Read
2	DINP1_TE0_07	6000.1	Bit	Read
2	DINP1_TE0_07	6000.1	Bit	Read
2	DINP1_TE0_07	6000.1	Bit	Read
2	DINP1_TE0_07	6000.1	Bit	Read
2	DINP1_TE0_07	6000.1	Bit	Read
2	DINP1_TE0_0F	6000.2	Bit	Read
2	DINP1_TE0_0F	6000.2	Bit	Read
2	DINP1_TE0_0F	6000.2	Bit	Read
2	DINP1_TE0_0F	6000.2	Bit	Read
2	DINP1_TE0_0F	6000.2	Bit	Read
2	DINP1_TE0_0F	6000.2	Bit	Read
2	DINP1_TE0_0F	6000.2	Bit	Read
2	DINP1_TE0_0F	6000.2	Bit	Read
2	DINP1_TE3_0F	6000.3	Bit	Read

Allo stesso modo premendo sul TAG **Lista parametri,** si vede l'elenco dei parametri da inviare dal master ai vari nodi presenti nella rete in fase di configurazione.



	Name	Object	Value
1		5000.1	0x1111
2		5000.2	0x1111
3		5000.3	0x1111
1		5000.1	0x1111
2		5000.2	0x1111
3		5000.3	0x1111
4		5000.4	0x1111
5		1005	0x340032

Queste liste, come si può vedere, elencano le variabili ed i parametri di tutti i nodi componenti la rete che si sta costruendo.

8.5 Configurazione parametri di rete

Nella parte destra della finestra sono mostrati i parametri per configurare la rete CANopen:

- Baudrate: velocità di trasmissione dei dati
- Canale CAN: si sceglie quali dei due canali del TPAC utilizzare
- Tempo di ciclo della rete CANopen
- Guard time: tempo che trascorre tra due messaggi di NG (Node Guarding) da parte del master
- Life Time: numero di periodi di Guard time oltre il quale, in caso di mancata ricezione del messaggio di NG si considera il master disconnesso

- Indica al master la richiesta di invio del sync
- Indica che il master dovrà inviare il NG con il bit di dato che cambierà stato ad ogni invio

Questi parametri sono inviati al master che in fase di configurazione si imposterà di conseguenza. Una volta modificati i parametri di rete questi saranno memorizzati salvando il progetto.

8.6 Menu principale

Il menu principale è composto dalle sezioni

- File
- Strumenti
- Aiuto

8.6.1 Menu File

Nel menu file è possibile selezionare:

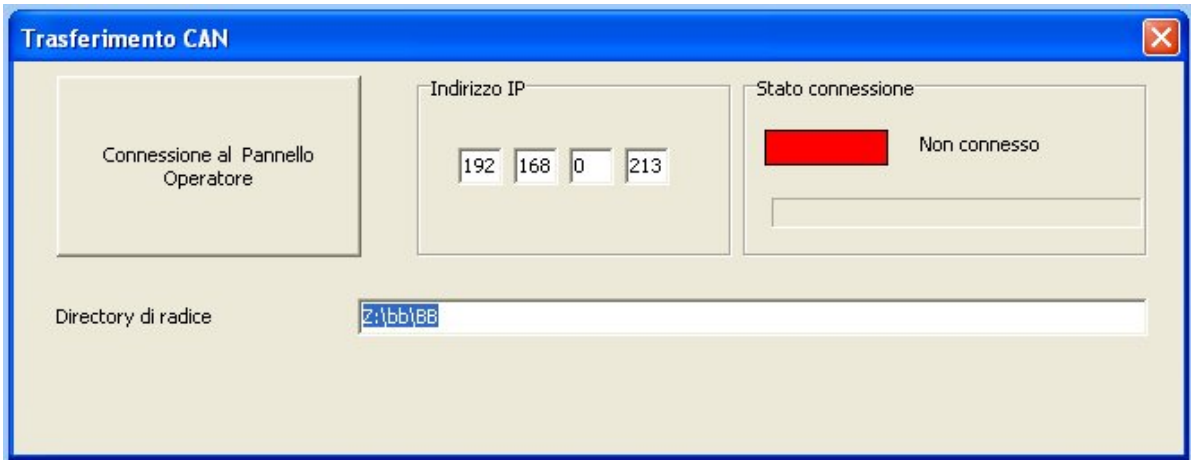
- Costruisci file di configurazione
- Scarica file di configurazione
- Salva progetto
- Esci

8.6.1.1 Costruisci file di configurazione

La scelta Costruisci file di configurazione crea nella directory di progetto i file che dovranno essere inviati al master della rete.

8.6.1.2 Scarica i file di configurazione

I file di configurazione creati, verranno inviati al master della rete CANopen TPAC attraverso la rete LAN. Premendo dal menu File: **Scarica i file di configurazione** si apre la finestra seguente:



Nella sezione Indirizzo IP è possibile impostare l'indirizzo che il master ha nella rete LAN, quindi premendo il pulsate **Connessione al Pannello Operatore,** si trasferiscono i file di configurazione.

8.6.1.3 Salva progetto

Selezionando Salva progetto vengono creati i file di progetto che potranno essere utilizzati in seguito.

8.6.1.4 Esci

Per uscire dal configuratore MECT Can Builder si preme **Esci** dal menu File. Se sono state effettuate delle modifiche al progetto, il programma chiederà se salvare o meno le modifiche effettuate.

8.6.2 Menu Strumenti

Nel menu strumenti sono presenti due opzioni:

- Riordina
- Importa EDS

8.6.2.1 Riordina

In caso di inserimento e cancellazione di nodi nella rete è possibile che i questi non siano ordinati con ID crescente, è perciò possibile premendo **Riordina**, sistemare i nodi della rete in ordine crescente di ID.

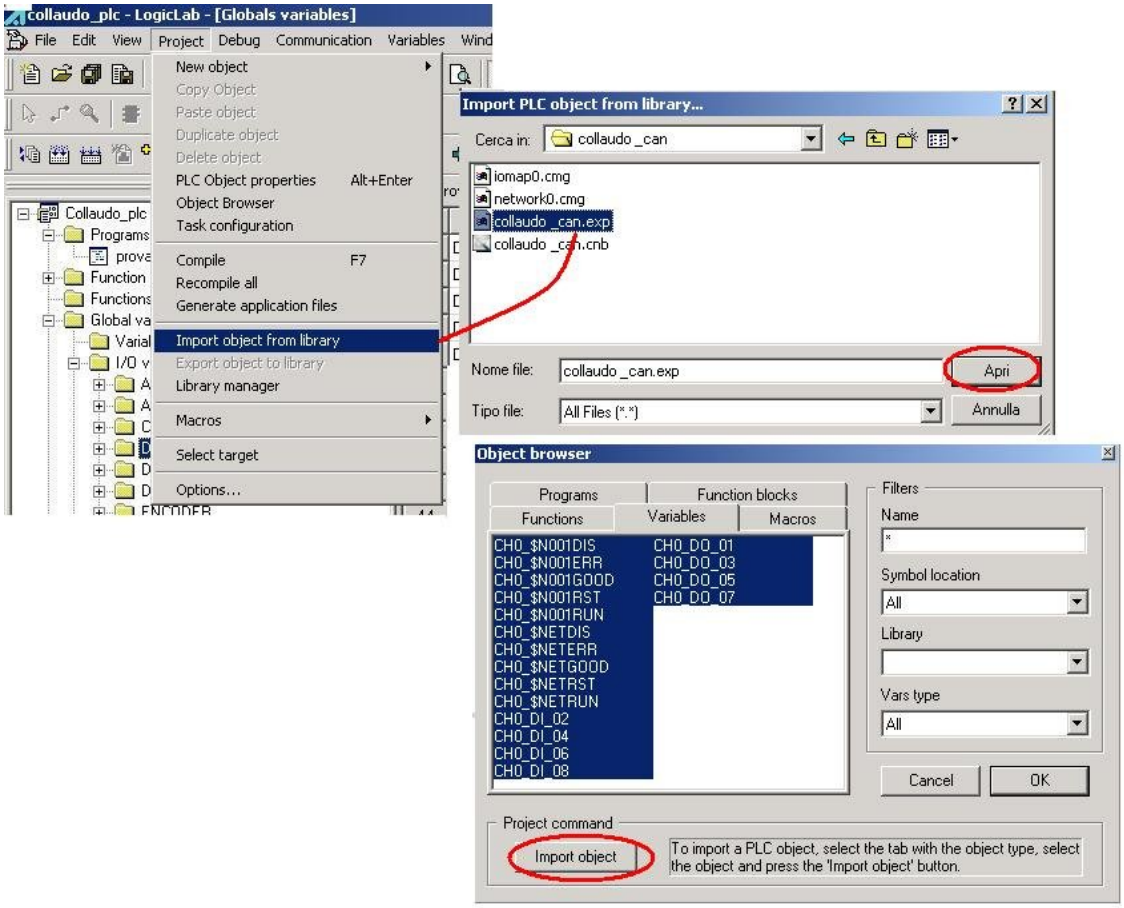
8.6.2.2 Importa EDS

Il software MECT Builder permette di utilizzare in una rete CANopen con master MECT un qualunque tipo di nodo CANopen. Per poter utilizzare un nodo è necessario per prima cosa importare il file EDS fornito dal produttore. Premendo **Importa EDS** dal menu strumenti si apre la finestra di dialogo che permette di inserire e gestire un nuovo tipo di nodo.
La cancellazione di un file EDS si effettua selezionando il file che si intende eliminare nella lista dei file eds, quindi premendo il tasto destro del mouse appare il menu di pop up che permette la cancellazione del file. I file verranno cancellati.

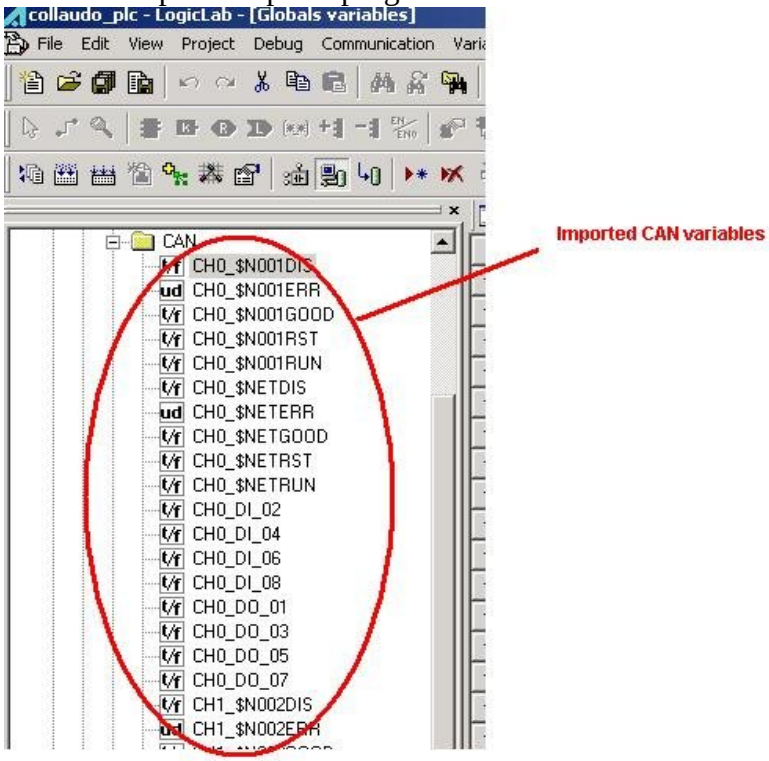
8.7 Utilizzo degli oggetti CAnOPEN nel PLC

8.7.1 Importazione variabili Canopen in LogicLab

Per utilizzare il nodo inserito nella rete CAN, si devono importare le variabili create con **Can Builder** nel progetto PLC per usarle come tutte le altre variabili PLC. Per importare le variabili CANOpen, dal menu **Project** di **LogicLab:** selezionare **Import object from library;** dalla finestra di pop-up che appare selezionare nella directory del progetto CAN il file **var.exp** creato con Can Builder e quindi importarlo.



A questo punto si può aprire il file, selezionare il tab Variables della finestra Object browser. Si visualizza una lista di variabili CAN. Selezionarle tutte, premere il tasto Import object. Le variabili ora sono disponibili per il progetto PLC ed utilizzabili come una qualunque altra variabile locale.



8.7.2 Utilizzo oggetti SDO

La filosofia del protocollo CANOpen associa ai PDO le variabili che si vogliono conoscere in tempo reale o che devono essere modificate costantemente durante il programma PLC. Per contro, i parametri di configurazione del nodo che devono essere modificati generalmente all'inizio di una nuova lavorazione, o comunque poco frequentemente, sono inviati attraverso i servizi SDO (Service Data Object). A tal proposito sono state implementate due funzioni SDORead e SDOWrite che permettono di utilizzare questi servizi CANOpen per configurare i nodi. Gli SDO possono accedere a qualsiasi parametro presente nel dizionario degli oggetti di un nodo, non sono però utilizzabili per la modifica in tempo reale dei valori dei parametri in quanto, essendo servizi a conferma, sono piuttosto lenti.
Dopo aver nominato tutti gli oggetti ai quali si intende accedere si procede all'upload del progetto sul TPAC come descritto in precedenza.
Una volta configurati, gli oggetti sono letti e scritti all'interno del programma PLC richiamando le funzioni SDORead e SDOWrite.
Gli oggetti di tipo SDO sono accessibili da ambiente PLC tramite due funzioni SDORead che effettua la lettura di un SDO ed SDOWrite che effettua una scrittura su un oggetto del dizionario.

8.7.2.1 Utilizzo di SDORead in un programma PLC

La funzione **res:=SDORead(0,'pr80')** contiene due parametri: il primo indica su quale delle due reti CANOpen presenti sul TPAC inviare la richiesta, il secondo invece, indica il nome dell'oggetto che abbiamo assegnato in precedenza nel programma Can Builder.
La funzione restituisce un codice che indica se è andata a buon fine (codice 0) o meno. Il dettaglio sui codici d'errore restituiti sono descritti nel manuale utente.
Nel caso di una SDORead la ricezione del codice d'errore 0 indica solo che il servizio SDO di richiesta di un parametro CANOpen è stato inoltrato sulla rete ed accettato. Il nodo al quale il servizio è inviato risponderà con i suoi tempi quindi, la ricezione del valore 0 a seguito di una SDORead non indica affatto che il dato richiesto sia disponibile. Il dato in lettura è presente in memoria ed inserito nella variabile SDODATA, solo quando, a seguito di una SDORead, la variabile SDOStatus è diversa da 0.

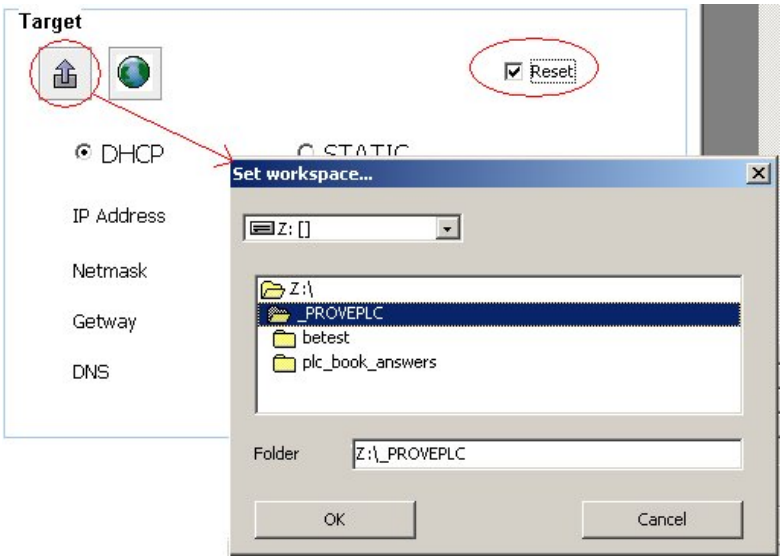
8.7.2.2 Utilizzo di SDOWrite in programma PLC

Nella SDOWrite, oltre alla rete sulla quale inviare la richiesta ed il nome dell'oggetto al quale fare riferimento, si deve indicare come parametro anche il valore da scrivere sull'oggetto CANOpen indirizzato; la funzione restituisce un codice che indica se è andata a buon fine o meno.
Poiché gli SDO sono servizi con conferma (cioè il nodo al quale è inviata la richiesta di scrittura risponde inviando un codice di conferma della ricezione del dato), per effettuare una serie di scritture consecutive su una stessa rete è necessario che il programma PLC a seguito di una SDOWrite, attenda che la variabile SDOStatus vada ad un valore diverso da 0 prima di effettuare la scrittura successiva.
Importante: come si nota nelle funzioni SDORead ed SDOWrite ciascun oggetto della rete è identificato con un nome che deve essere univoco, quindi se abbiamo due nodi identici appartenenti alla stessa rete e vogliamo accedere allo stesso oggetto (es: oggetto 2000.51) dobbiamo assegnare a ciascun oggetto di ciascun nodo un nome univoco.

9.0 RESET DEL SISTEMA

Se per un qualunque motivo non è più possibile mantenere il controllo del TPAC (per programmazione o configurazione) è disponibile una back door che permette di effettuare il reset del TPAC. Dalla sezione **Target** della dashboard, eelezionare la casella **Reset**. Inserire nel personal computer la penna USB utilizzata in precedenza per la configurazione, quindi, selezionato il drive

nel quale è mappata la penna, premere il pulsante: **export current configuration**. La pen-disk sarà caricata con la configurazione di reset.



Successivamente, inserire la pen-disk nella porta USB del TPAC ed accenderlo. I files dei codici del PLC, HMI e CANOpen saranno cancellati dalla memoria del TPAC. Si potrà riavviare il TPAC che non avrà più in memoria i file causa del problema.

10.0 RICETTE

10.1 INTRODUZIONE

TPAC ha la possibilità di inserire all'interno di un programma PLC la gestione delle ricette. Il pacchetto d'interfaccia ricette è caricato automaticamente alla creazione di ogni nuovo progetto HMI. Distinguiamo in questo manuale le operazioni che deve compiere il programmatore rispetto a quelle dell'utilizzatore.

10.1.1 Lato programmatore

Per utilizzare le ricette in un progetto PLC, il programmatore, come primo passo, deve identificare e definire le variabili che sono necessarie alla ricetta dichiarandole nella parte di memoria riservata alle variabili ritentive.
Definite le variabili, il programmatore dovrà associarle alla ricetta. L'associazione delle variabili PLC alla ricetta si effettua in ambiente di programmazione HMI. Alla creazione di un nuovo progetto, nel pacchetto ricette è disponibile la procedura globale: **Rcpt_add_vars**.
All'interno di questa procedura il programmatore utilizzerà le funzioni di aggiunta variabili alla ricetta: **Rcpt_add_tyReal_par** per aggiungere una variabile di tipo reale e **Rcpt_add_tyDInt_par** per aggiungerne una di tipo intero con segno.

Tali funzioni richiedono quattro parametri:

- La posizione della variabile all'interno della ricetta: numero intero da zero a n.
- Il nome della variabile PLC da associare: come definito nel progetto PLC.
- Lo mnemonico associato alla variabile, che può essere il nome della stessa variabile o una qualsiasi descrizione con massimo 18 caratteri.
- L'unità di misura associata alla variabile: stringa di massimo 5 caratteri.

La funzione restituisce il valore zero se è andata a buon fine.

Esempio d'inserimento nella prima posizione della ricetta, della variabile PLC Var1, definita come variabile reale, il cui nome dello mnemonico associato è "velocità" e con unità di misura "m/s". Il risultato dell'operazione è inserito nella variabile Risult.

```
Risult:=Rcpt_add_tyReal_par(0,Var1,'velocita','m/s');
```

Una volta selezionata la ricetta d'interesse, per utilizzarne i parametri in un programma PLC è necessario effettuare una copia dei valori della ricetta selezionata nelle variabili PLC. Il programmatore deve predisporre questa copia inserendo nella procedura **Rcpt_store_vars**, le funzioni per assegnare il contenuto di ogni elemento della ricetta ad una variabile PLC. Le funzioni da richiamare sono: **Rcpt_get_tyDInt_par(i)** per copiare il contenuto dell'i-esimo elemento della ricetta in una variabile intera, e **Rcpt_get_tyReal_par(i)**, per copiare l'i-esimo elemento in una variabile reale.

La procedura **Rcpt_store_vars** è richiamata quando l'utente preme il tasto **Scarica su PLC.**

Importante: le variabili PLC che fanno parte della ricetta devono essere poste nella zona ritentiva della memoria del TPAC, in questo modo i valori delle variabili non saranno persi allo spegnimento del TPAC.

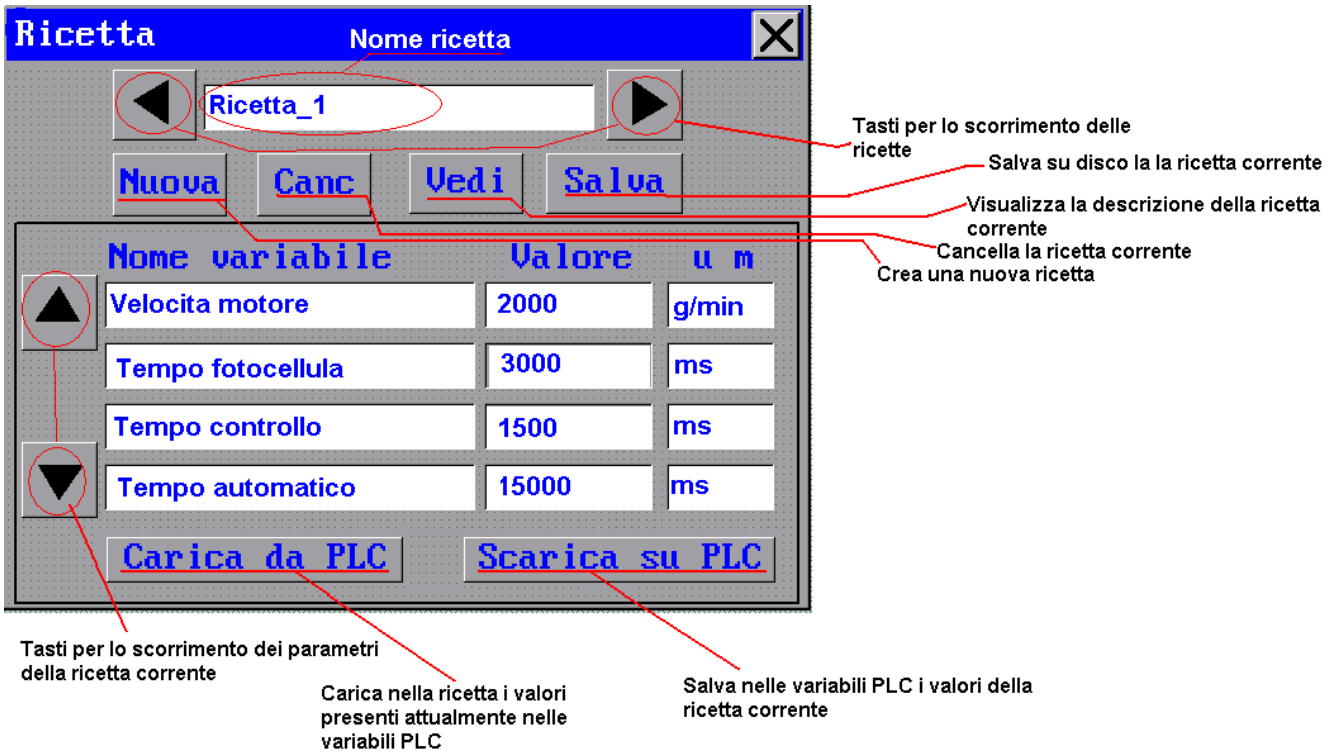
Per accedere al pacchetto ricette è necessario richiamare l'apertura della finestra "**ric**".

10.1.2 Lato utente

L'utente ha a disposizione un'interfaccia grafica che gli permette di:

- Inserire nuove ricette

- Visualizzare i parametri delle ricette
- Cancellare ricette
- Scorrere l'elenco delle ricette
- Visualizzare e modificare le descrizioni associate alle ricette
- Visualizzare e modificare i valori contenuti nelle variabili PLC che fanno parte delle ricette



Creare una nuova ricetta

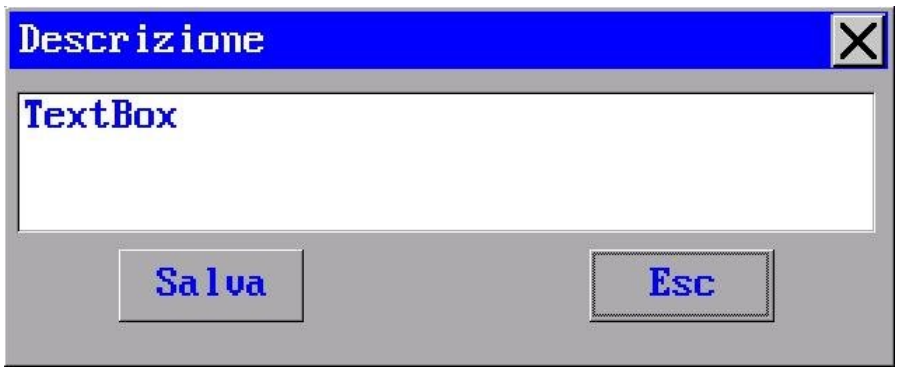
Per creare una nuova ricetta l'utente deve premere il tasto **Nuova**, apparirà la finestra **Nuova ricetta** nella quale si dovrà inserire il nome della ricetta e una descrizione della stessa.

Per inserire il nome della ricetta bisogna toccare il campo **Nome ricetta**, apparirà a video una tastiera virtuale per l'inserimento del nome della ricetta che potrà avere un massimo di 18 caratteri. Terminato l'inserimento premere ↵ sulla tastiera virtuale o **enter** su quella hardware.

Il campo descrizione è un commento associato alla ricetta che può aiutare l'operatore ad identificarla meglio. Toccando sullo schermo il campo descrizione, apparirà a video la tastiera virtuale. La descrizione può contenere fino a 190 caratteri. Per terminare l'inserimento della descrizione premere **Esc** sulla tastiera virtuale.

Premere il tasto **OK** dopo aver inserito il nome della ricetta ed eventualmente la descrizione per salvarla in memoria. I valori dei parametri della nuova ricetta, sono i valori delle variabili in quel momento presenti nel PLC. Premendo invece **ESC** si esce dalla pagina **Nuova ricetta** e si torna alla pagina **Ricetta** senza aver salvato nulla.

Visualizzazione e modifica della descrizione
La descrizione associata ad una ricetta può essere visualizzata premendo il tasto **Vedi** nella pagina **Ricetta**.



Una volta visualizzata la descrizione, per modificarne il contenuto è sufficiente premere sul campo di testo, in questo modo apparirà a video la tastiera virtuale. Per uscire e salvare le modifiche premere il tasto **Salva**, per uscire senza salvare le modifiche premere **Esc**.
Modifica dei parametri di una ricetta
Dopo aver creato una nuova ricetta o dopo averne selezionata una, ne vengono visualizzati a video i primi quattro parametri. Utilizzando le frecce è possibile fare la scansione di tutti i parametri che compongono la ricetta. I parametri visualizzati sono divisi in tre campi: **Nome variabile**, **Valore**, **u m** (unità di misura). Il contenuto dei campi **Nome variabile** ed **u m** non può essere modificato dall'utente in quanto definito dal programmatore; il campo V**alore** è invece modificabile

dall'utente. Inseriti valori delle variabili che compongono la ricetta secondo le proprie esigenze, l'utente, può salvarne il contenuto premendo il tasto **Salva**.

Salvataggio dei parametri di una ricetta nelle variabili del PLC
Le ricette sono memorizzate sulla flash del TPAC; per salvare i parametri di una ricetta nelle variabili PLC, l'operatore deve per prima cosa visualizzare la ricetta d'interesse e quindi premere il tasto **Scarica su PLC**. I valori dei parametri presenti nella ricetta selezionata saranno inseriti nelle variabili PLC ed immediatamente utilizzate dal programma.

Visualizzazione delle variabili PLC correnti
Premendo il tasto **Carica da PLC,** l'utente ha la possibilità di vedere i valori delle variabili attualmente in uso dal PLC.

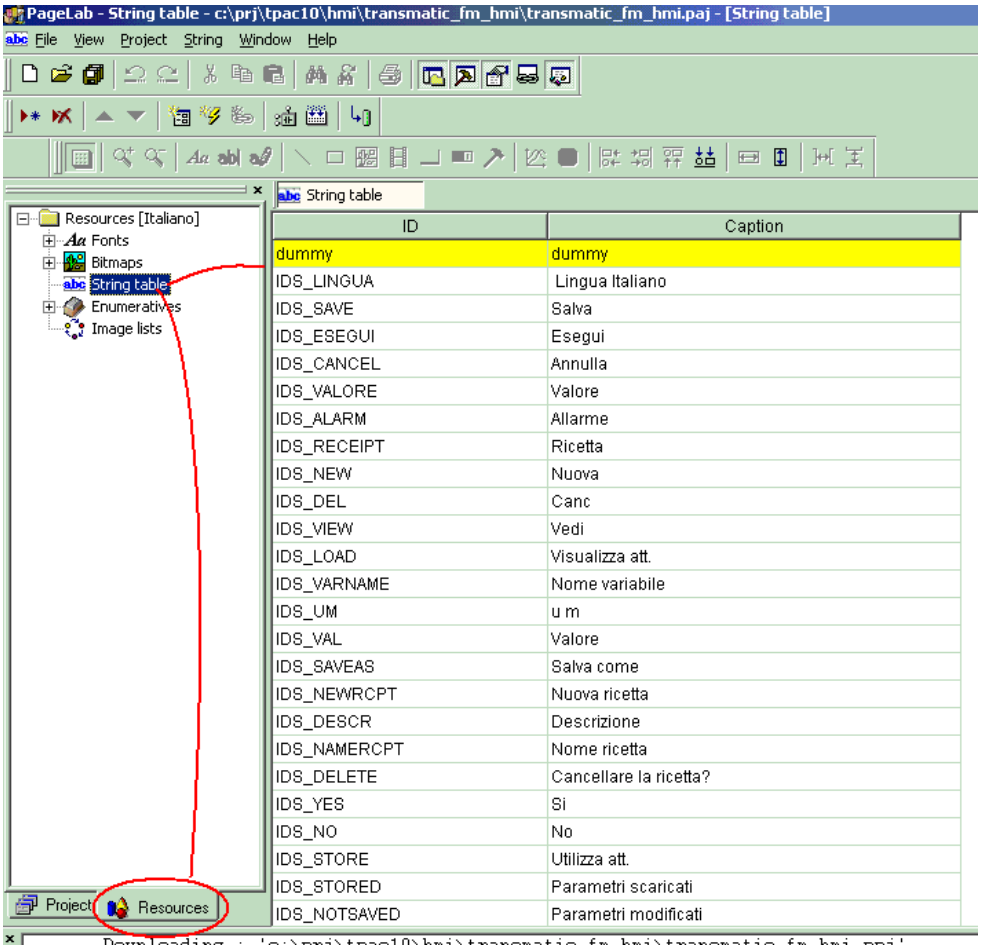
11.0 GESTIONE LINGUE IN HMI

11.1 INTRODUZIONE

Quando si inserisce una stringa in un progetto PageLab, è sempre possibile scrivere esplicitamente il testo da visualizzare, oppure fare riferimento al contenuto di una stringa delle risorse, citando il suo identificativo. Nel primo caso, il testo rimarrà sempre costante, mentre ricorrendo alle stringhe delle risorse sarà visualizzato il testo corrispondente al linguaggio attivo.

11.2 TABELLA DELLE STRINGHE

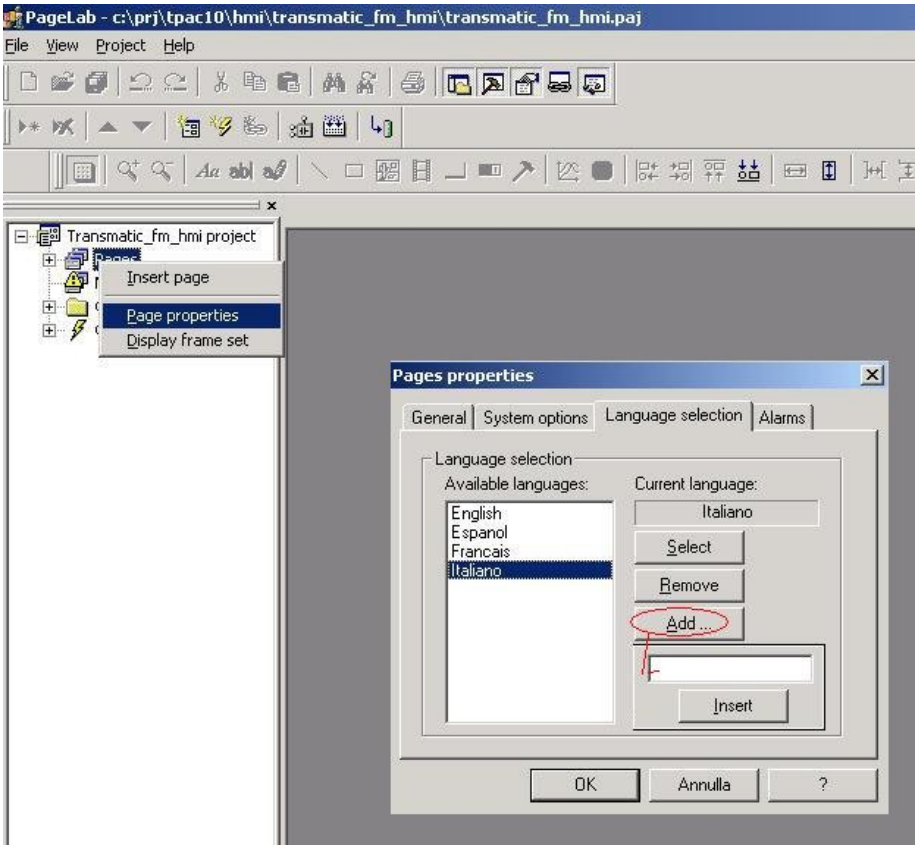
Per visualizzare la tabella delle stringhe premere il Tag **Resources** nella finestra di progetto e selezionare la voce **String table.** La tabella è composta da due colonne, sulla colonna sinistra è indicato un identificativo (**ID**), mentre su quella destra la stringa che apparirà a video (**Caption**).



Per aggiungere una stringa alla tabella premere il tasto nella barra degli strumenti, una nuova riga si aggiungerà alla tabella, selezionare la cella **ID**, ed inserire il nome dell'identificativo, quindi selezionare la cella **Caption** ed inserire la stringa da associare all'identificativo. In questo modo una nuova stringa farà parte del progetto.

11.3 INSERIMENTO DI UNA NUOVA LINGUA

Selezionare **Pages**, quindi il tag **Page properties:** nella finestra di progetto appare la finestra **Pages properties.** Selezionando il tab **Language selection** nella sezione **Available languages** si visualizza una lista delle lingue fino a quel momento definite. Per l'aggiunta di una nuova lingua premere il tasto **Add…,** inserire il nome della lingua che si desidera aggiungere, quindi premere **Insert,** la nuova lingua viene aggiunta alla lista.



Per modificare la lingua corrente nelle stringhe di **PageLab** evidenziare una lingua nella lista **Available languages** e premere il tasto **Select.,**
Premendo **OK** si chiude la finestra **Pages properties** e si salvano le modifiche.

Selezionata la lingua, il programmatore dovrà modificare la tabella delle stringhe inserendo le traduzioni. Per far ciò deve accedere alla tabella delle stringhe come descritto nel paragrafo precedente e modificare i valori nella colonna **Caption**.

Attenzione: la selezione della lingua appena descritta, serve solo al programmatore per compilare le tabelle delle stringhe con le traduzioni, la scelta della lingua visualizzata sul TPAC non dipende in alcun modo dalla lingua correntemente selezionata nell'ambiente **PageLab**.

11.4 MODIFICA DELLA LINGUA SUL TPAC

La variazione della lingua visualizzata sullo schermo del TPAC avviene modificando la variabile: **sysLangID**. I valori assegnabili alla variabile sono definiti da costanti. Per esempio se in PageLab sono state definite 3 lingue: ITA, ENG e FRA, il compilatore creerà tre costanti: **kLangITA**; **kLangENG**; **kLangFRA**. Per modificare la lingua visualizzata è sufficiente assegnare una delle costanti alla variabile **sysLangID** e quindi riavviare il TPAC.

Per effettuare invece un cambiamento della lingua senza dover riavviare il sistema è necessario modificare la funzione globale **chg_lang()** all'interno del programma HMI.

Per ogni lingua creata in PageLab sono generate altre due costanti. Le costanti sono composte da un suffisso che è il nome la lingua creata (p.e. ITA, ENG, FRA, ecc) e un prefisso **kbLangIdx** per le parole definite nella tabella delle stringhe e **kbEnumIdx** per le stringhe enumerative. Con riferimento all'esempio precedente saranno create le costanti: **kLangIdxITA**, **kLangIdxFRA**, **kLangIdxENG**, e **kbEnumIdxITA**, **kbEnumIdxENG** e **kbEnumIdxFRA**.

La funzione chg_lang() sarà quindi modificata come segue:

CASE sysLangID OF

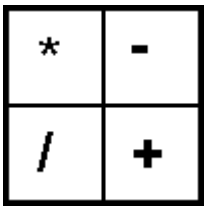
```
   kLangITA: dummy:=Video_LoadLanguage(?kbResStrings[kbLangIdxITA],
?kbResEnums[kbEnumIdxITA] );

   kLangENG: dummy := Video_LoadLanguage( ?kbResStrings[kbLangIdxESP],
?kbResEnums[kbEnumIdxESP] );

   kLangFRA: dummy :=Video_LoadLanguage( ?kbResStrings[kbLangIdxFRA],
?kbResEnums[kbEnumIdxFRA] );

END_CASE;

x1 := 1 ;
y1 := 1 ;
x2 := 318 ;
y2 := 238 ;
bStatRedrw := TRUE ;
bMapsReset := TRUE ;
EXIT ;
```

Le linee da inserire all'interno del costrutto CASE sono tante quante le lingue che si vogliono gestire. Questa procedura deve essere richiamata ogni volta che avviene una variazione della variabile sysLangID.

12.0 TASTIERA SOFTWARE

TPAC è fornito di una tastiera numerica posizionata nella parte bassa del pannello frontale. Se come nel caso delle ricette, si definiscono dei campi editabili associati a variabili non solo numeriche (nome della ricetta o la descrizione), quando questi sono selezionati dall'utente, si visualizza una tastiera virtuale che permette di inserire nei campi anche le lettere.
La tastiera definita di default è la seguente:

Il programmatore può creare una propria tastiera con layout e significato dei tasti differenti.
Per definire una nuova tastiera, il programmatore deve creare un'immagine in formato "gif" della tastiera che intende realizzare.
Per esempio, supponiamo di voler creare una tastiera con quattro tasti: + - * /.
Con un programma di editor grafico generiamo l'immagine seguente:



L'immagine tastiera deve avere dimensioni inferiori a quelle dello schermo (320 x 240) altrimenti non potrà essere rappresentata correttamente. Il file deve essere chiamato **skbd.gif**

Oltre all'immagine della tastiera il programmatore deve crearsi un file che descrive le funzionalità della stessa.

Il formato del file deve essere il seguente: nella riga 1 si definiscono le dimensioni, espresse in pixel della tastiera (dimensione x e dimensione y). Dalla seconda riga in avanti si definiscono le funzioni di ciascun tasto così codificate:

- Codice ASCII del tasto
- Coordinata x dell'angolo in alto a sinistra
- Coordinata y dell'angolo in alto a sinistra
- Coordinata x dell'angolo in basso a destra
- Coordinata y dell'angolo in basso a destra

Il file deve avere nome **skbd.txt**.

Nel caso dell'esempio il file skbd.txt sarà:

```
100 100
42    3    3    48    48
45    51   3    96    48
47    3    51   48    96
43    51   51   96    96
```

I file creati si devono salvare sul TPAC come descritto nel paragrafo: aggiornamento software.

13.0 DEFINIZIONE DEI COLORI IN HMI

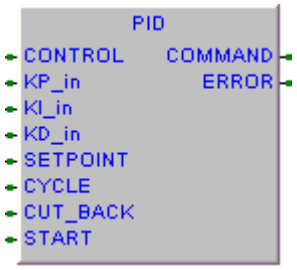
Il TPAC utilizza una tavolozza di 256 colori, codificati nel formato RGB con 8 bit per colore. Di default in **PageLab** sono definiti 16 colori il cui codice è riportato nella tabella sottostante:

ID	CODICE esadecimale RGB
0.	000000
1.	000080
2.	008000
3.	800000
4.	008080
5.	800080
6.	808000
7.	808080
8.	AAAAAA
9.	0000FF
10.	00FF00
11.	FF0000
12.	00FFFF
13.	FF00FF
14.	FFFF00
15.	FFFFFF

Se si vogliono definire altri colori, il programmatore deve inserire nella sezione: **ENUM 103: color codes** del file **C:\Programmi\MECT\TPAC_01\dashboard\MECTjar\TemplateHMI\Targets\TPAC.def**, i nuovi colori che desidera aggiungere.
Importante: il numero di colori definiti deve essere sempre una potenza di 2.

14.0 BLOCCHI FUNZIONE

PID



Parametro	Tipo dati	Dimensione	Descrizione
CONTROL	REAL	32	Grandezza in misurazione
KP_in	REAL	32	Costante proporzionale
KI_in	REAL	32	Costante integrativa
KD_in	REAL	32	Costante derivativa
SETPOINT	REAL	32	Setpoint
CYCLE	UDINT	32	Tempo di ciclo del PID espresso in ms
CUT_BACK	REAL	32	Costante di Cutback
START	BOOL	8	Comando d'inizio regolazione
COMMAND	REAL	32	Valore di regolazione (-100% a +100%)
ERROR	BOOL	8	Segnala che alcuni parametri non sono stati correttamente impostati

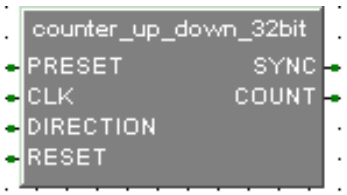
Descrizione dell'operazione
Regolazione PID
Quando START va a 1, il blocco funzione effettua una regolazione PID in funzione dei parametri KP_in, KI_IN, KD_in E cut_back in ingresso. L'uscita del regolatore assume valori che vanno da –100% a +100% afinchè CONTROL raggiunga e mantenga il SETPOINT. Se si imposta il tempo di ciclo o la costante proporzionale KP_in a 0, il blocco alza l'uscita ERROR e non effettua alcuna regolazione.

Self tuning PID

Parametro	Tipo dati	Dimensione	Descrizione
CONTROL	REAL	32	Grandezza in misurazione
START	BOOL	8	Comando d'inizio del selftuning
SETPOINT	REAL	32	Setpoint
KP_out	REAL	32	Costante proporzionale calcolata
KI_out	REAL	32	Costante integrativa calcolata
KD_out	REAL	32	Costante derivativa calcolata
COMMAND	REAL	32	Valore di regolazione (-100% a +100%)
TUNED	BOOL	8	Indica il termine della procedura di selftuning
CUT_BACK	REAL	32	Costante di Cutback calcolata

Descrizione dell'operazione
Selftuning PID
Quando START va a 1, il blocco funzione inizia la procedura di selftuning utilizzando i parametri impostati. L'uscita COMMAND varia in funzione dell'algoritmo di selftuning Al termine della procedura l'uscita TUNED è posta a 1 e sulle uscite KP_out, KI_out, KD_out e CUT_BACK sono disponibili i valori calcolati.

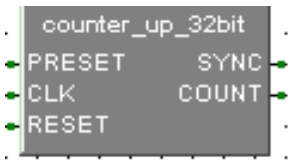
Contatore 32 bit Up – Down



Parametro	Tipo dati	Dimensione	Descrizione
PRESET	UDINT	32	Valore di conteggio impostato da 0 a 2^32
CLK	BOOL	1	Ingresso di clock
DIRECTION	BOOL	1	Ingresso che indica se conta avanti o indietro DIRECTION = 1 conta avanti DIRECTION = 0 conta indietro
RESET	BOOL	1	Se RESET = 0 il conteggio va a zero se DIRECTION = 1 (count up) Il conteggio va a PRESET se DIRECTION = 0 (count down) SYNC va ad 1
SYNC	BOOL	1	Uscita che indica il termine del conteggio e rimane ad 1 se in count up raggiunge PRESET, o in count down raggiunge 0
COUNT	UDINT	32	Valore corrente del contatore

Descrizione dell'operazione
Contatore a 32 bit in salita (up) e discesa (down).
Il contatore varia il suo valore di conteggio se l'ingresso CLK cambia da 0 ad 1 (fronte di salita).
Se DIRECTION = 1 il valore di conteggio è incrementato di 1, se DIRECTION = 0 il valore di conteggio è decrementato di 1.
Se DIRECTION = 1, raggiunto il valore di PRESET, l'uscita SYNC va ad 1.
Se DIRECTION = 0, raggiunto il valore di 0, l'uscita SYNC va ad 1.
Al successivo fronte di salita su CLK, se DIRECTION = 1 il contatore riprende da 0, se DIRECTION = 0 il contatore riprende da PRESET.
Se RESET = 0 il contatore si posiziona a 0 se DIRECTION = 1, si posiziona a PRESET, se DIRECTON = 0.
Se RESET = 0 il contatore è insensibile all'ingresso CLK

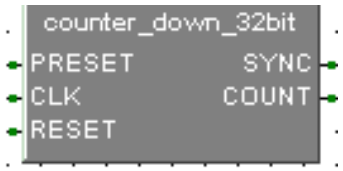
Contatore 32 bit Up



Parametro	Tipo dati	Dimensione	Descrizione
PRESET	UDINT	32	Valore di conteggio impostato da 0 a 2^32
CLK	BOOL	1	Ingresso di clock
RESET	BOOL	1	Se RESET = 0 il conteggio va a 0 SYNC va ad 1
SYNC	BOOL	1	Uscita che indica il termine del conteggio e rimane ad 1 se il conteggio raggiunge PRESET.
COUNT	UDINT	32	Valore corrente del contatore

Descrizione dell'operazione
Contatore a 32 bit in salita (up)
Il contatore incrementa di 1 il suo valore di conteggio se l'ingresso CLK cambia da 0 ad 1 (fronte di salita)
Raggiunto il valore di PRESET, l'uscita SYNC va ad 1.
Al successivo fronte di salita su CLK, il contatore riprende da 0.
Se RESET = 0 il contatore si posiziona a 0
Se RESET = 0 il contatore è insensibile all'ingresso CLK

Contatore 32 bit Down

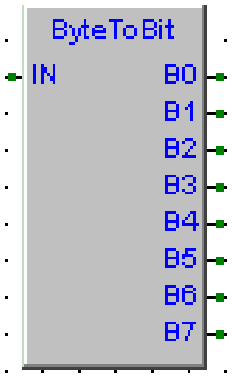


Parametro	Tipo dati	Dimensione	Descrizione
PRESET	UDINT	32	Valore di conteggio impostato da 0 a 2^32
CLK	BOOL	1	Ingresso di clock
RESET	BOOL	1	Se RESET = 0 il conteggio va a PRESET SYNC va a 1

SYNC	BOOL	1	Uscita che indica il termine del conteggio e rimane ad 1 se il conteggio raggiunge 0.
COUNT	UDINT	32	Valore corrente del contatore

Descrizione dell'operazione Contatore a 32 bit in discesa (down) Il contatore decrementa di 1 il suo valore di conteggio se l'ingresso CLK cambia da 0 ad 1 (fronte di salita) Raggiunto il valore di 0, l'uscita SYNC va ad 1. Al successivo fronte di salita su CLK, il contatore riprende da PRESET. Se RESET = 0 il contatore si posiziona a PRESET Se RESET = 0 il contatore è insensibile all'ingresso CLK

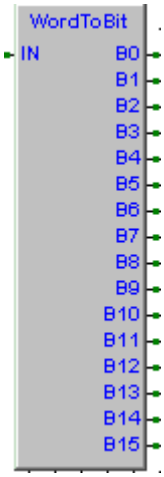
Byte to bit



Parametro	Tipo dati	Dimensione	Descrizione
IN	BYTE	8	Byte in ingresso
B0..B7	BOOL	1	Bit di uscita

Descrizione dell'operazione Divide il byte in ingresso in 8 bit

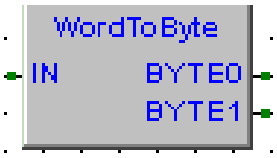
Word to bit



Parametro	Tipo dati	Dimensione	Descrizione
IN	WORD	16	Word in ingresso
B0..B15	BOOL	1	Bit di uscita

Descrizione dell'operazione Divide la word in ingresso in 16 bit

Word to byte



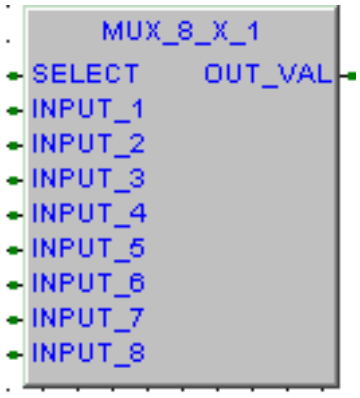
Parametro	Tipo dati	Dimensione	Descrizione
IN	WORD	16	Byte in ingresso
BYTE0	BYTE	8	Byte di uscita basso
BYTE1	BYTE	8	Byte di uscita alto

Descrizione dell'operazione Divide la word in ingresso in due byte

Double Word to byte

Parametro	Tipo dati	Dimensione	Descrizione
IN	UDINT	32	Double word in ingresso
BYTE0	BYTE	8	Byte di uscita 0
BYTE1	BYTE	8	Byte di uscita 1
BYTE2	BYTE	8	Byte di uscita 2
BYTE3	BYTE	8	Byte di uscita 3
Descrizione dell'operazione Divide la double word in ingresso in quattro byte			

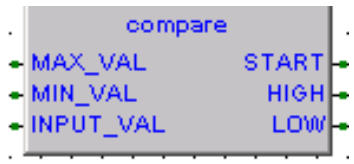
MUX_8_X_1 multiplexer 8 per 1



Parametro	Tipo dati	Dimensione	Descrizione
INPUT_1.. INPUT_8	BOOL	1	Bit in ingresso
SELECT	USINT	8	Seleziona il bit di ingresso da mandare in uscita
OUT_VAL	BOOL	1	Bit di ingresso selezionato

Descrizione dell'operazione L'uscita OUT_VAL è l'ingresso selezionato da SELECT

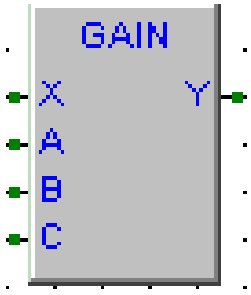
COMPARE



Parametro	Tipo dati	Dimensione	Descrizione
MAX_VAL	UDINT	32	Valore massimo di comparazione
MIN_VAL	UDINT	32	Valore minimo di comparazione
INPUT_VAL	UDINT	32	Valore in ingresso
START	BOOL	1	Uscita: 1 se valore in ingresso in range
HIGH	BOOL	1	Uscita: 1 se valore in ingresso fuori soglia alta
LOW	BOOL	1	Uscita: 1 se valore in ingresso fuori soglia bassa

Descrizione dell'operazione
Regolazione ON-OFF
Confronta il valore in ingresso su 32 bit con i valori impostati sugli ingressi MAX_VAL e MIN_VAL; se l'ingresso è all'interno del range, l'uscita START è ad 1; se l'ingresso è superiore a MAX_VAL, l'uscita START è a 0, l'uscita HIGH ad 1; se l'ingresso è inferiore a MIN_VAL l'uscita START è a 0, l'uscita LOW ad 1.

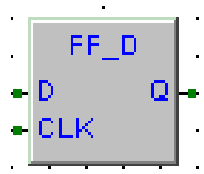
GAIN



Parametro	Tipo dati	Dimensione	Descrizione
X	UDINT	32	Ingresso X
A	UDINT	32	Parametro numeratore
B	UDINT	32	Parametro denominatore
C	UDINT	32	Parametro offset
Y	UDINT	32	Uscita Y

Descrizione dell'operazione
Effettua il calcolo della funzione Y=X*(A/B)+C

FF_D: flip flop di tipo D



Parametro	Tipo dati	Dimensione	Descrizione
D	BOOL	1	Bit in ingresso
CLK	BOOL	1	Ingresso di clock
Q	BOOL	1	Bit di uscita

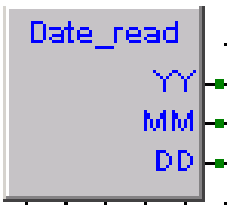
Descrizione dell'operazione
Se sull'ingresso CLK viene rilevato un fronte di salita (transizione 0-1) l'uscita Q assume il valore presente sull'ingresso D

FF_T: flip flop di tipo T

Parametro	Tipo dati	Dimensione	Descrizione
CLK	BOOL	1	Ingresso di clock
T	BOOL	1	Bit di uscita

Descrizione dell'operazione
Se sull'ingresso CLK viene rilevato un fronte di salita (transizione 0-1) l'uscita Q assume il valore negato rispetto al valore attuale.

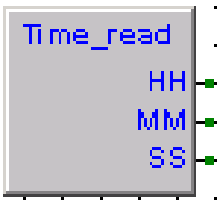
Date_ read: lettura data corrente



Parametro	Tipo dati	Dimensione	Descrizione
YY	UINT	8	Anno
MM	UINT	8	Mese
DD	UINT	8	giorno

Descrizione dell'operazione
L'invocazione di questa funzione restituisce il valore presente nelle variabili di sistema: DATA_YY DATA_MM DATA_DD

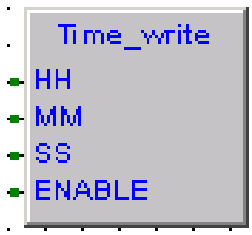
Time_ read: lettura ora corrente



Parametro	Tipo dati	Dimensione	Descrizione
HH	UINT	8	ore
MM	UINT	8	minuti
SS	UINT	8	secondi

Descrizione dell'operazione L'invocazione di questa funzione restituisce il valore presente nelle variabili di sistema: TIME_HH TIME_MM TIME_SS

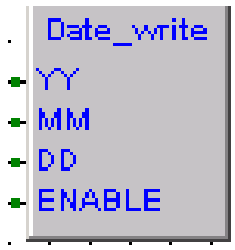
Time_write: scrittura ora



Parametro	Tipo dati	Dimensione	Descrizione
HH	UINT	8	ore
MM	UINT	8	minuti
SS	UINT	8	secondi
ENABLE	BOOL	1	Bit di abilitazione

Descrizione dell'operazione Se ENABLE = 1, l'invocazione di questa funzione scrive sull'orologio di sistema i valori presenti in ingresso

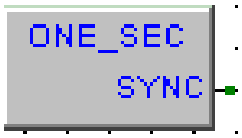
Date_write: scrittura data



Parametro	Tipo dati	Dimensione	Descrizione
YY	UINT	8	Anno
MM	UINT	8	Mese
DD	UINT	8	giorno
ENABLE	BOOL	1	Bit di abilitazione

Descrizione dell'operazione Se ENABLE = 1, l'invocazione di questa funzione scrive sull'orologio di sistema i valori presenti in ingresso

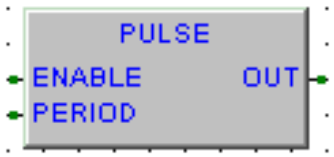
ONE_SEC: impulso con frequenza 1 secondo



Parametro	Tipo dati	Dimensione	Descrizione
SYNC	BOOL	1	Impulso di sincronismo a 1secondo

Descrizione dell'operazione
Genera un impulso della durata di un ciclo PLC ogni secondo.

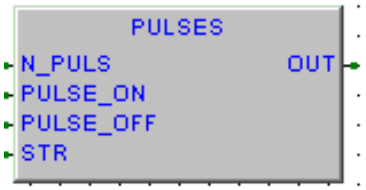
pulse



Parametro	Tipo dati	Dimensione	Descrizione
ENABLE	BOOL	1	Bit di abilitazione
PERIOD	UINT	16	Durata impulso
OUT	BOOL	1	Impulso d'uscita

Descrizione dell'operazione
Se l'ingresso ENABLE è 1, l'uscita OUT è a 1 per una durata pari a PERIOD

pulses

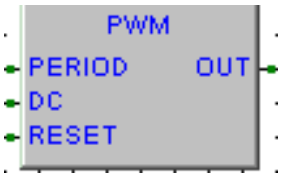


Parametro	Tipo dati	Dimensione	Descrizione
N_PULS	UINT	16	Numero di impulsi da generare
PULSE_ON	UINT	16	Durata ON di ciascun impulso
PULSE_OFF	UINT	16	Durata OFF di ciascun impulso
STR	BOOL	1	Ingresso di abilitazione

OUT	BOOL	1	Uscita

Descrizione dell'operazione Se sull'ingresso STR c'è un fronte di salita l'uscita genera un treno di impulsi impostato dall'ingresso N_PULS, della durata di PULSE_ON, con intervallo di PULSE_OFF tra un impulso e l'altro.

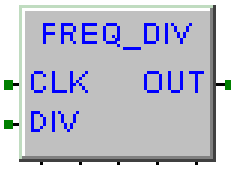
pwm



Parametro	Tipo dati	Dimensione	Descrizione
PERIOD	UINT	16	Periodo PWM
DC	UINT	16	Duty cycle
RESET	UINT	16	Ingresso di reset
OUT	BOOL	1	Uscita

Descrizione dell'operazione Se l'ingresso RESET è ad 1, l'uscita genera una forma d'onda rettangolare con periodo pari a PERIOD e duty cycle pari a DC.

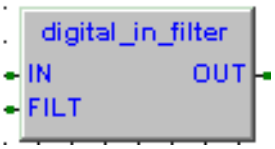
freq_div



Parametro	Tipo dati	Dimensione	Descrizione
CLK	BOOL	1	Frequenza d'ingresso
DIV	UINT	16	Fattore di divisione
OUT	BOOL	1	Uscita

Descrizione dell'operazione Divide la frequenza del segnale in ingresso del fattore presente sull'ingresso DIV.

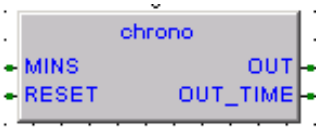
digital_in_filter



Parametro	Tipo dati	Dimensione	Descrizione
IN	BOOL	1	Segnale di ingresso
FILT	UINT	16	Fattore di filtraggio
OUT	BOOL	1	Uscita

Descrizione dell'operazione
L'uscita OUT viene aggiornata al valore presente in ingresso se questo risulta stabile per almeno il numero impostato sull'ingresso FILT di cicli di PLC.

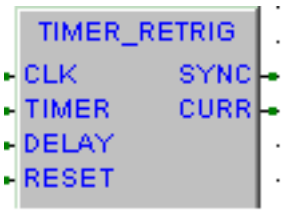
chrono



Parametro	Tipo dati	Dimensione	Descrizione
MINS	UDINT	16	Minuti impostati sul cronometro
START	BOOL	16	Ingresso di inizio conteggio
OUT	BOOL	1	Uscita di fine conteggio
OUT_TIME	UDINT	1	Tempo trascorso dallo start

Descrizione dell'operazione
Effettua un conteggio di minuti, quando il conteggio raggiunge il valore impostato su MINS l'uscita OUT va ad 1 e vi rimane fino a che RESET non va a 0. Il conteggio riparte se RESET torna ad 1

timer_retrig

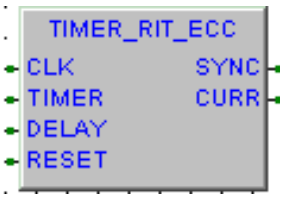


Parametro	Tipo dati	Dimensione	Descrizione
CLK	UDINT	32	Ingresso di clock
TIMER	BOOL	1	Selezione di uno dei 128 timer disponibili
DELAY	BOOL	1	Programmazione ritardo
RESET	UDINT	32	Reset timer
SYNC	BOOL	1	Uscita di fine conteggio
CURR	UDINT	32	Valore corrente conteggio

Descrizione dell'operazione
TIMER CON RITARDO ALL'ECCITAZIONE RETRIGGERABILE.

Avvia un temporizzatore se all'ingresso CLK c'è un
fronte di salita. Per abilitare il temporizzatore è necessario che il segnale RESET sia
ad 1. Il temporizzatore continua ad operare per la durata del valore dell'ingresso DELAY
anche se il segnale su CLK va a 0 prima che il tempo sia trascorso.
L'uscita SYNC va ad 1 quando il tempo è trascorso.
Il timer viene riavviato con il tempo specificato se sul CLK si rileva un nuovo fronte
di salita.

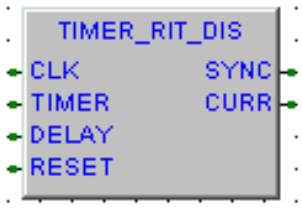
timer_rit_ecc



Parametro	Tipo dati	Dimensione	Descrizione
CLK	UDINT	32	Ingresso di clock
TIMER	BOOL	1	Selezione di uno dei 128 timer disponibili
DELAY	BOOL	1	Programmazione ritardo
RESET	UDINT	32	Reset timer
SYNC	BOOL	1	Uscita di fine conteggio
CURR	UDINT	32	Valore corrente conteggio

Descrizione dell'operazione
TIMER CON RITARDO ALL'INSERZIONE.
Avvia un temporizzatore se all'ingresso CLK c'è un
fronte di salita. Per abilitare il temporizzatore è necessario che il segnale RESET sia
ad 1. Il temporizzatore continua ad operare per la durata del valore dell'ingresso DELAY;
se il segnale su CLK va a 0 prima che il tempo sia trascorso, il timer si azzera .
L'uscita SYNC va ad 1 quando il tempo è trascorso.
Il timer viene riavviato con il tempo specificato se sul CLK si rileva un nuovo fronte
di salita.

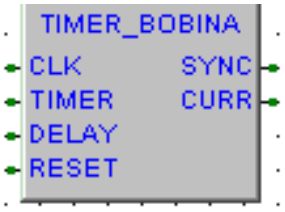
timer_rit_dis



Parametro	Tipo dati	Dimensione	Descrizione
CLK	UDINT	32	Ingresso di clock
TIMER	BOOL	1	Selezione di uno dei 128 timer disponibili
DELAY	BOOL	1	Programmazione ritardo
RESET	UDINT	32	Reset timer
SYNC	BOOL	1	Uscita di fine conteggio
CURR	UDINT	32	Valore corrente conteggio

Descrizione dell'operazione
TIMER CON RITARDO ALLA DISECCITAZIONE.
Avvia un temporizzatore se all'ingresso CLK c'è un fronte di discesa.
Per abilitare il temporizzatore è necessario che il segnale RESET sia ad 1. Il temporizzatore continua ad operare per la durata del valore dell'ingresso DELAY; se il segnale su CLK va a 1 prima che il tempo sia trascorso, il timer si azzera .
L'uscita SYNC va ad 1 quando il tempo è trascorso.
Il timer viene riavviato con il tempo specificato se sul CLK si rileva un nuovo fronte di discesa.

timer_coil

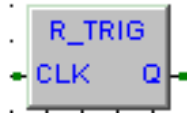


Parametro	Tipo dati	Dimensione	Descrizione
CLK	UDINT	32	Ingresso di clock
TIMER	BOOL	1	Selezione di uno dei 128 timer disponibili
DELAY	BOOL	1	Programmazione ritardo
RESET	UDINT	32	Reset timer
SYNC	BOOL	1	Uscita di fine conteggio
CURR	UDINT	32	Valore corrente conteggio

Descrizione dell'operazione
BOBINA CON USCITA PROLUNGATA RETRIGGERABILE.

Avvia un temporizzatore se all'ingresso CLK c'è un fronte di salita.
Per abilitare il temporizzatore è necessario che il segnale RESET sia ad 1.
Il temporizzatore continua ad operare per la durata del valore dell'ingresso DELAY, anche se il segnale su CLK va a 0 prima che il tempo sia trascorso.
L'uscita SYNC rimane ad 1 fino a quando il tempo è trascorso.
Il timer viene riavviato con il tempo specificato se sul CLK si rileva un nuovo fronte di salita.

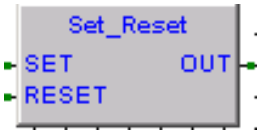
r_trig



Parametro	Tipo dati	Dimensione	Descrizione
CLK	BOOL	1	Ingresso di clock
Q	BOOL	1	uscita

Descrizione dell'operazione
Rileva un fronte di salita dell'ingresso CLK. Q è 0 quando l'ingresso CLK = 0.

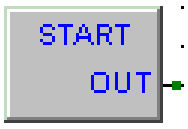
Set_Reset



Parametro	Tipo dati	Dimensione	Descrizione
SET	BOOL	1	Ingresso di set
RESET	BOOL	1	Ingresso di reset
OUT	BOOL	1	Uscita

Descrizione dell'operazione
L'uscita è a livello logico 1 se SET = 1, rimane ad 1 fino a quando RESET non va ad 1

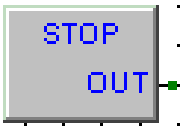
start



Parametro	Tipo dati	Dimensione	Descrizione
OUT	BOOL	1	Uscita a livello logico 1

Descrizione dell'operazione
L'uscita è a livello logico 1

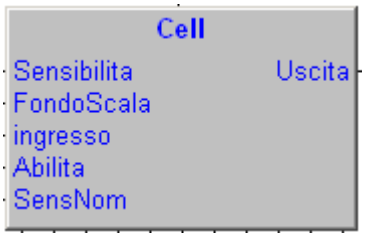
stop



Parametro	Tipo dati	Dimensione	Descrizione
OUT	BOOL	1	Uscita a livello logico 0

Descrizione dell'operazione
L'uscita è a livello logico 0

Cell (Solo TPAC 02 versione cella)



Parametro	Tipo dati	Dimensione	Descrizione
Sensibilita	REAL	4	Sensibilità di targa della cella di carico
FondoScala	REAL	4	Fondo scala della cella di carico
Ingresso	REAL	4	Ingresso Cella di carico (variabili: da CELL_1 a CELL_4)
Abilita	BOOL	1	Se 1, abilita il blocco funzione
SensNom	REAL	4	Sensibilità nominale della cella collegata
Uscita	REAL	4	Peso misurato

Descrizione dell'operazione
Se Abilita = TRUE: il peso misurato è espresso in funzione dell'ingresso, del FondoScala impostato e della sensibilità. La sensibilità di targa (Sensibilita) viene rapportata alla sensibilità nominale della cella (SensNom).

14.1 Funzioni Embedded

Nome	Descrizione
ABS	Valore assoluto
ACOS	Arccoseno
ADD	Addizione
AND	AND logico
ASIN	Arcoseno
ATAN	Arctangente
ATAN2	Arctangente di Y/X
CEIL	Intero superiore di un valore
COS	Coseno
COSH	Coseno iperbolico
DIV	Divisione
EQ	Confronta uguaglianza
EXP	Esponenziale base e
FLOOR	Intero inferiore di un valore
GE	Maggiore uguale
GT	Maggiore
JMP	Salta
LE	Minore uguale
LIMIT	Limite
LN	Logaritmo naturale
LOG	Logaritmo Base –10
LT	Minore
MAX	Valore massimo
MIN	Valore minimo
MOD	Operatore modulo
MOVE	Move
MUL	Moltiplicazione
MUX	Multiplexer
NE	Diverso
NOT	Not
OR	OR logico
POW	Potenza
R	Reset
RET	Return
ROL	Rotazione binaria a sinistra
ROR	Rotazione binaria a destra
S	Set
SEL	Selettore
SHL	Shift binario a sinistra
SHR	Shift binario a destra
SINH	Seno iperbolico
SQRT	Radice quadrata
SUB	Sottrazione
TAN	Tangente
TANH	Tangente iperbolica
TO_BOOL	Conversione BOOL
TO_DINT	Conversione a DINT
TO_INT	Conversione INT
TO_REAL	Conversione REAL
TO_SINT	Conversione SINT

Nome	Descrizione
TO_UDINT	Conversione UDINT
TO_UINT	Conversione UINT
TO_USINT	Conversione USINT
XOR	XOR logico

15.0 CONFIGURARE IL TPAC ATTRAVERSO L'INTERFACCIA WEB

15.1 Premessa

Il TPAC dispone di una serie di parametri e periferiche che devono essere configurate prima di poter essere utilizzate. Le configurazioni sono effettuate attraverso un'interfaccia web utilizzando un qualunque browser internet.

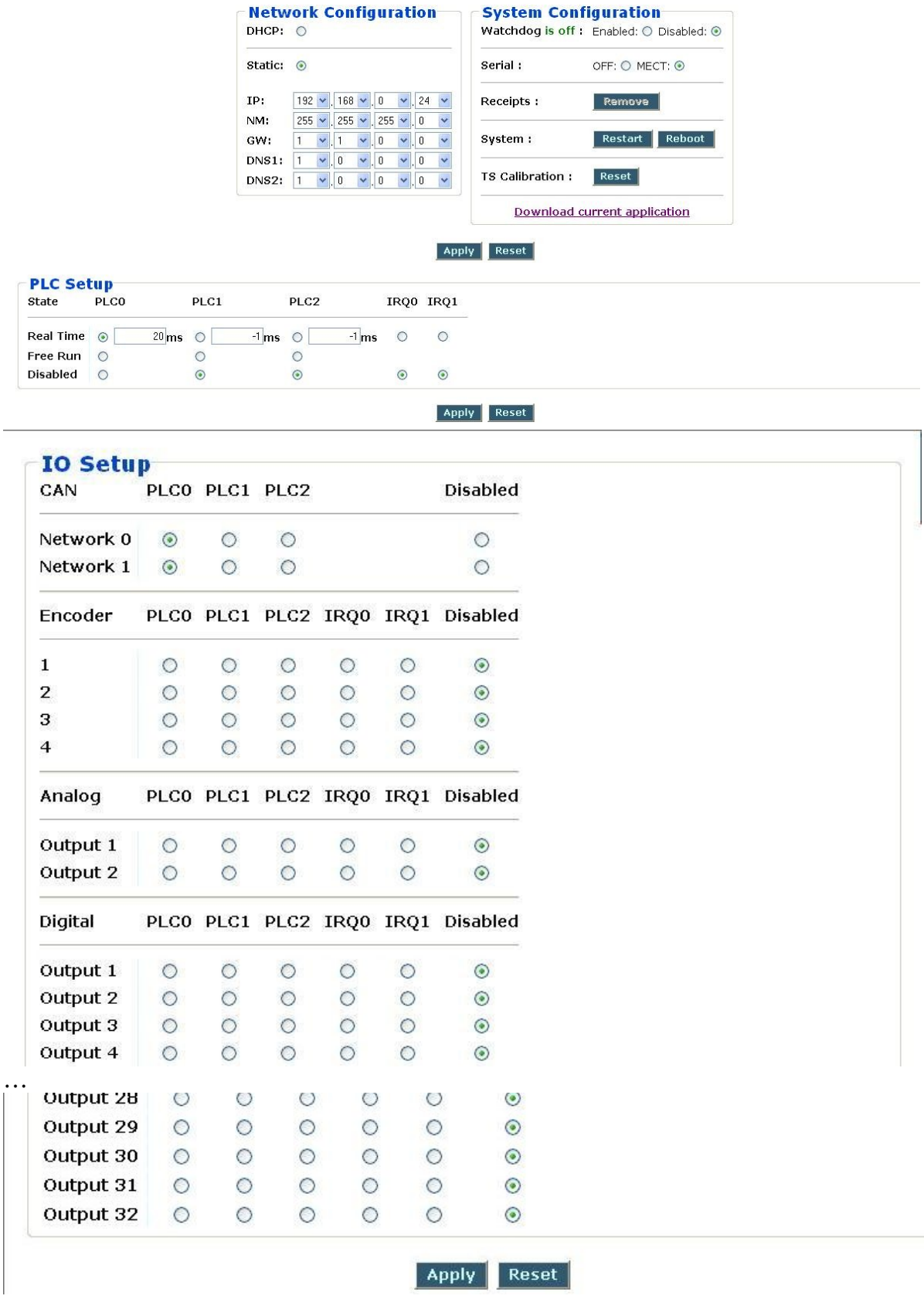
15.2 Aprire la pagina web del TPAC

Nell'esempio utilizzeremo il browser Internet Explorer nella versione 6.0 (IE6). Colleghiamo il TPAC alla rete LAN ed alimentiamolo, quindi lanciamo IE6 ed inseriamo nel campo Indirizzo: *http:// < indirizzo IP del TPAC>*.
L'assegnazione dell'indirizzo IP al TPAC avviene come descritto nel paragrafo: Configurazione del TPAC.

Dopo aver inserito l'indirizzo IP premere Invio, il TPAC risponde inviando al browser la pagina web mostrata nella figura seguente. All'interno della pagina web possiamo notare le sezioni:

Network Configuration
System Configuration
PLC setup
IO Setup
IO Configuration
IDs

TPAC010 Configuration



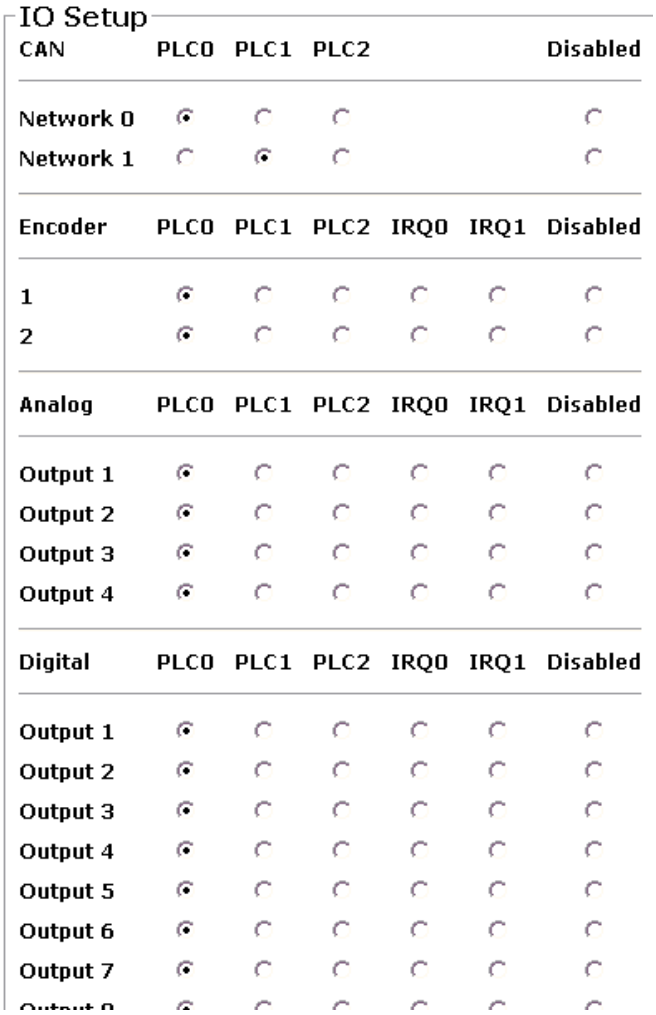
15. 3 Configurazione

La sezione Network Configuration permette di modificare l'indirizzo IP al quale, il TPAC risponderà al successivo riavvio.
La sezione System Configuration permette di attivare una serie di parametri:
Il Watchdog
Il protocollo seriale MECT
La cancellazione delle ricette salvate sul TPAC da un precedente progetto
Il restart del sistema
La calibrazione del touch screen
Il download del dell'applicazione PLCe HMI in esecuzione sul TPAC

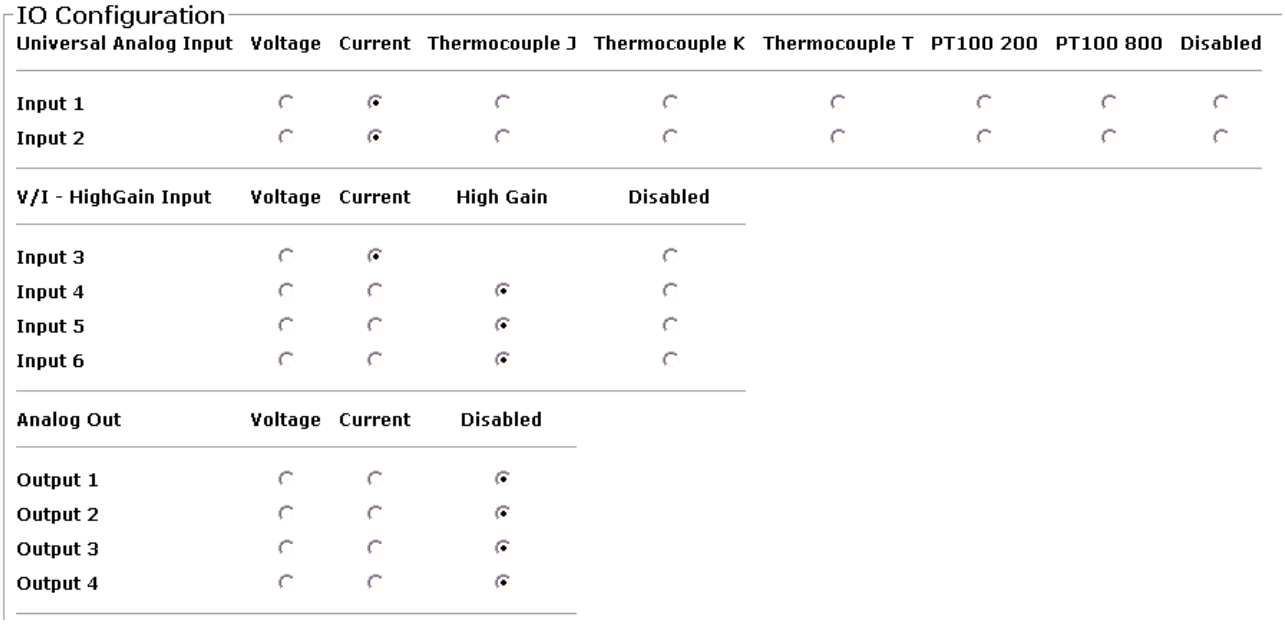
La sezione PLC setup permette di impostare il tempo di ciclo di ciascuno dei PLC utilizzabili; l'impostazione del tempo di ciclo di ciascun PLC si effettua inserendo, nella casella di testo corrispondente, il valore (espresso in millisecondi) del tempo di ciclo richiesto.
E' necessario impostare solo i tempi ciclo dei PLC che si intendono impiegare nel progetto, perciò se si prevede l'utilizzo di un unico PLC, si imposta il tempo di ciclo del PLC0 al valore richiesto, mentre si disabilitano gli altri due. Se si seleziona la casella Free Run, il PLC corrispondente verrà eseguito senza un tempo di ciclo prestabilito.
Importante: un PLC con tempo di ciclo diverso da zero, ma non utilizzato nel progetto consuma tempo di CPU riducendo le prestazioni del sistema.

Il comando di una periferica di output può essere effettuato in modo esclusivo da un solo PLC, la sezione IO Setup assegna ad ogni PLC, quali delle uscite analogiche e digitali, dei canali CanOpen, e encoder può utilizzare.

Nell'esempio in figura si è assegnata la rete CAN 0 al PLC 0 e la rete CAN 1 al PLC 1. Impostando invece Idle, si indica al TPAC che la corrispondete rete CANOpen non deve essere gestita. Come accennato in precedenza, un PLC può gestire entrambe le reti CANOpen, ma non è possibile assegnare la stessa rete CANOpen a più di un PLC. Un discorso analogo vale per uscite analogiche e digitali.

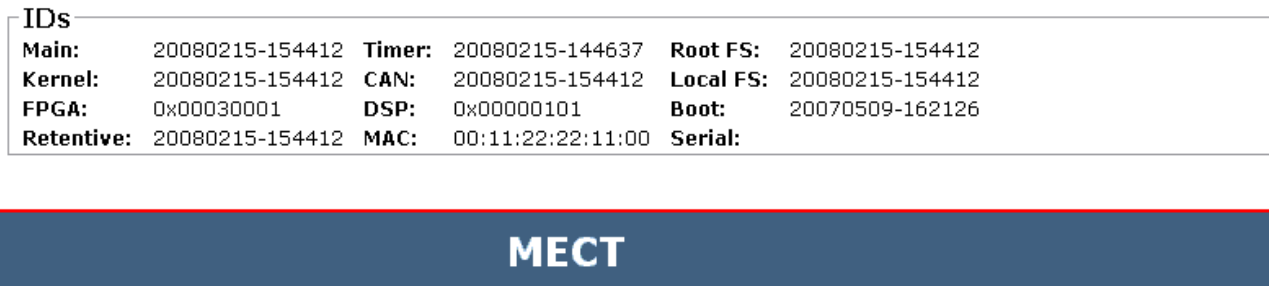


La sezione IO Configuration invece imposta, per i canali analogici, sia ingressi che uscite, il tipo di grandezza fisica che rileva (in caso di ingressi analogici) o genera (nel caso di uscite).



Scelta la configurazione, premere Set values per salvare sul TPAC i parametri impostati. Al successivo riavvio del TPAC, i parametri di configurazione saranno automaticamente impostati.

L'ultima sezione IDs mostra le revisioni dei software e firmware componenti il sistema



16.0 FUNZIONI

Sono disponibili una serie di funzioni per il controllo del TPAC:

1. Funzioni immediate
2. Funzioni di delay input
3. Timers
4. Controllo funzioni video
5. Funzioni ricette
6. Data e Ora
7. Seriale MECT
8. Seriale MODBUS RTU
9. CanOpen
10. Importazione di un file di testo in hmi
11. Datalogger
12. Funzioni stringhe
13. Funzioni somma e differenze tra date

16.1 FUNZIONI IMMEDIATE

Nei PLC, la variazione delle uscite o l'acquisizione degli ingressi avviene ad ogni ciclo, è però a volte necessario modificare le uscite il più velocemente possibile senza attendere il termine del ciclo. In questi casi sono necessarie delle funzioni dette immediate che effettuano l'aggiornamento delle variabili istantaneamente. Le funzioni in questione sono elencate di seguito:

Nome	Descrizione/ esempio	Parametri di ingresso / Valore restituito
Imm_read_encoder	Legge il valore presente sull'encoder	Parametri: numero encoder –1
	Esempio di lettura del valore sull'encoder 2: valore_letto:= Imm_read_encoder(1);	Restituisce il valore letto, restituisce 0xffffffff in caso di errore
Imm_set_encoder	Imposta il valore di preset dell'encoder	Parametri: numero encoder –1; valore di preset da impostare sull'encoder.
	Esempio di scrittura del valore 14322 sull'encoder 1: risultato:= Imm_set_encoder(0,14322);	Restituisce 0 se la funzione è andata buon fine, restituisce 1 in caso di errore
Imm_set_encoder_mode	Imposta il modo di conteggio dell'encoder	Parametri: numero encoder –1; valore del modo da impostare sull'encoder.
	Esempio di scrittura del modo 3 sull'encoder 1: risultato:= Imm_set_encoder(0,3);	Restituisce 0 se la funzione è andata buon fine, restituisce 1 in caso di errore

Imm_read_digital_input	Legge il valore di un ingresso digitale	Parametri: numero input–1.
	Esempio di lettura del valore sull'ingresso 45: valore_letto:= Imm_read_digital_input (44);	Restituisce il valore letto, restituisce 0xffff in caso di errore
Imm_read_digital_feedback	Legge il valore dello stato delle uscite	Parametri: (uscita da rilevare-1) / 2.
	Esempio di lettura del valore dello stato delle uscite 9 e 10: valore_letto:= Imm_read_digital_feedback (4);	Restituisce il valore letto, restituisce 0xffff in caso di errore
Imm_write_digital_output	Scrive gli out digitali	Parametri: numero uscita – 1 ; valore uscita (0 1).
	Esempio di scrittura a 1 dell'uscita 3: risultato:= Imm_write_digital_output (2,1);	Restituisce 0 se la funzione è andata buon fine, restituisce 1 in caso di errore
Imm_read_analog_i	Legge il valore di un ingresso analogico	Parametri: numero ingresso analogico - 1.
	Esempio di lettura del valore sull'ingresso analogico 2: valore_letto:= Imm_read_analog_i (1);	Restituisce il valore letto, restituisce 0xffff in caso di errore
Imm_write_analog_o	Scrive il valore di uscita analogica	Parametri: numero uscita analogica – 1; valore (0 – 4095) da scrivere sull'uscita.
	Esempio di scrittura del valore 100 sull'uscita 1: risultato:= Imm_write_analog_o (100,0);	Restituisce 0 se la funzione è andata buon fine, restituisce 1 in caso di errore

16.2 FUNZIONI DELAY

Per evitare che gli ingressi digitali siano sensibili a segnali spuri, è possibile effettuare un filtraggio digitale che permette di considerare valido l'ingresso solo quando il segnale è stabile per un certo tempo; questo tempo è impostabile dal programmatore.

Il programmatore, per ogni ingresso che vuole sia ritardato deve effettuare due operazioni: impostare nella variabile **DELAY_DI[input],** il tempo di ritardo richiesto (espresso in millisecondi) e quindi chiamare la funzione **di_delays_set(input).**

La variabile input deve contenere il numero di ingresso che si vuole sia ritardato. Gli ingressi possibili vanno da 0 a 47.

Esempio di impostazione di un tempo di ritardo di 100ms sull'ingresso DI_5:

DELAY_DI[4]:=100;
risult:=di_delays_set(4);

Nell'esempio si vede che la funzione **di_delays_set()**, restituisce un valore che è zero se la funzione è andata a buon fine, altrimenti restituisce un valore diverso da zero.

16.3 TIMER

Il programmatore ha 128 timer a disposizione.
Quando il programmatore imposta la variabile TIMERS_FLAG[i] ad uno, il timer i-esimo (dove i può essere un qualsiasi valore compreso tra 0 e 127), inizia a contare per un tempo pari a quello impostato nella variabile TIMERS[i], arrivato al termine del conteggio, la variabile TIMERS[i] si azzera indicando così che il tempo è scaduto. Se si vuole ricominciare il conteggio del tempo prima che questo sia trascorso, è sufficiente reimpostare ad uno la variabile TIMERS_FLAG[i].

16.4 FUNZIONI DI CONTROLLO VIDEO

Al programmatore sono rese accessibili delle funzioni per il controllo diretto delle funzioni video.

Nome	Descrizione/ esempio	Parametri di ingresso / Valore restituito
LCDSS_set_timeout	Imposta il timeout, in secondi dello screensaver	Parametri: Tempo di timeout dello screensaver
	Esempio d'impostazione del timeout dello screensaver a 5 minuti: risultato:= LCDSS_set_timeout (300);	Restituisce 0 se la funzione è andata buon fine, restituisce 1 in caso di errore
LCDSS_get_timeout	Legge il timeout dello screensaver impostato	Parametri: costante;
	Esempio di lettura dello screensaver: risultato:= LCDSS_set_timeout (0);	Restituisce il valore di timeout impostato
LCD_set_backlight	Imposta il valore tra 0 e 100% della luminosità dello schermo	Parametri: valore luminosità schermo;
	Esempio d'impostazione della luminosità al 54%: risultato:= LCD_set_backlight (54);	Restituisce 0 se la funzione è andata buon fine, restituisce 1 in caso di errore
LCD_get_backlight	Legge il valore impostato della luminosità dello schermo	Parametri: costante; .
	Esempio lettura della luminosità : luminosita:= LCD_get_backlight (0);	Restituisce il valore di luminosità impostato

LCDSS_on	Accensione dello schermo.	Parametri: costante;
	Esempio d'accensione dello schermo: risultato:= LCDSS_on (0);	Restituisce 0 se la funzione è andata buon fine, restituisce 1 in caso di errore
LCDSS_off	Spegnimento dello schermo.	Parametri: costante;
	Esempio di spegnimento dello schermo: risultato:= LCDSS_off (0);	Restituisce 0 se la funzione è andata buon fine, restituisce 1 in caso di errore
LCDSS_enable	Abilita lo screensaver.	Parametri: costante;
	Esempio di abilitazione screensaver: risultato:= LCDSS_enable (0);	Restituisce 0 se la funzione è andata buon fine, restituisce 1 in caso di errore
LCDSS_disable	Disabilita lo screensaver.	Parametri: costante;
	Esempio di disabilitazione screensaver: risultato:= LCDSS_disable (0);	Restituisce 0 se la funzione è andata buon fine, restituisce 1 in caso di errore

16.5 FUNZIONI RICETTE

Il pacchetto ricette è già creato ed immediatamente utilizzabile, ma al programmatore sono anche rese disponibili le funzioni utilizzabili per crearsi un proprio pacchetto ricette.

Nome	**Descrizione/ esempio**	Parametri di ingresso / **Valore restituito**
Rcpt_set_curr	Imposta i parametri della ricetta corrente	Parametro: nome della ricetta corrente Parametro: descrizione
	Esempio: risultato:=Rcpt_set_curr (ric_name, description); Nella parametro ric_name è inserito il nome della ricetta, in description è inserita la descrizione.	Restituisce 0 se la funzione è andata buon fine, restituisce 1 in caso di errore
Rcpt_save_curr	Salva su Flash la ricetta corrente	Parametro: costante
	Esempio: risultato:=Rcpt_save_curr (0);	Restituisce 0 se la funzione è andata buon fine, restituisce 1 in caso di errore
Rcpt_clean_all	Cancella tutte le ricette caricate in memoria	Parametro: costante
	Esempio: risultato:=Rcpt_clean_all (0);	Restituisce 0 se la funzione è andata buon fine, restituisce 1 in caso di errore

Rcpt_clean_list	Cancella i dati della ricetta corrente	Parametro: costante
	Esempio: risultato:=Rcpt_clean_list (0);	Restituisce 0 se la funzione è andata buon fine, restituisce 1 in caso di errore
Rcpt_clean_curr	Cancella i dati della ricetta corrente	Parametro: costante
	Esempio: risultato:=Rcpt_clean_curr(0);	Restituisce 0 se la funzione è andata buon fine, restituisce 1 in caso di errore
Rcpt_remove	Cancella da flash la ricetta corrente	Parametro: nome della ricetta da cancellare
	Esempio: risultato:=Rcpt_remove (ric_name); Nel parametro ric_name è inserito il nome della ricetta.	Restituisce 0 se la funzione è andata buon fine, restituisce 1 in caso di errore
Rcpt_remove_all	Cancella da flash tutte le ricette	Parametro: costante
	Esempio: risultato:=Rcpt_remove (0);	Restituisce 0 se la funzione è andata buon fine, restituisce 1 in caso di errore
Rcpt_exists	Verifica se la ricetta esiste su flash	Parametro: nome della ricetta da cercare
	Esempio: risultato:=Rcpt_exists (ric_name); Nel parametro ric_name è inserito il nome della ricetta.	Restituisce 0 se la funzione è andata buon fine, restituisce 1 in caso di errore
Rcpt_get_first_set_curr	Restituisce il nome della prima ricetta salvata su flash e la imposta come ricetta corrente	Parametro: costante
	Esempio: ric_name:=Rcpt_get_first_set_curr (0); Nella variabile ric_name è inserito il nome della ricetta.	Restituisce il nome della ricetta o 0 se non esistono ricette.
Rcpt_get_next_set_curr	Restituisce il nome della successiva ricetta salvata su flash e la imposta come ricetta corrente	Parametro: costante

	Esempio:
	address:=Rcpt_get_next_set_curr (0);
	ric_name:=address;
	Nella variabile ric_name è inserito l'indirizzo della ricetta.
		Restituisce l'indirizzo della ricetta o 0 se si è arrivati al termine della lista delle ricette.

Rcpt_get_prev_set_curr	Restituisce il nome della ricetta precedente, salvata su flash e la imposta come ricetta corrente	Parametro: costante

	Esempio:
	address:=Rcpt_get_prev_set_curr (0);
	ric_name:=address;
	Nella variabile ric_name è inserito l'indirizzo della ricetta.
		Restituisce l'indirizzo della ricetta o 0 se si è arrivati al termine della lista delle ricette.

Rcpt_load_curr	Carica i valori salvati su flash nella ricetta visualizzata.	Parametro: nome della ricetta
	Esempio: risultato:=Rcpt_load_curr (ric_name); Nella variabile ric_name è inserito il nome della ricetta.	Restituisce 0 se la funzione è andata buon fine, restituisce 1 in caso di errore
Rcpt_set_curr_descr	Imposta la descrizione della ricetta corrente.	Parametro: descrizione
	Esempio: risultato:=Rcpt_set_curr_descr (description); Nella variabile description è inserita la descrizione della ricetta.	Restituisce 0 se la funzione è andata buon fine, restituisce 1 in caso di errore
Rcpt_get_curr_descr	Carica la descrizione della ricetta corrente.	Parametro: costante
	Esempio: description:=Rcpt_set_curr_descr (0); Nella variabile description è inserita la descrizione della ricetta.	Restituisce la stringa di descrizione
Rcpt_add_ty_USInt_par	Aggiunge un parametro su 8 bit intero senza segno alla lista.	Parametro: indice del parametro all'interno della ricetta Parametro: Nome della variabile PLC da inserire Parametro: Mnemonico associato alla variabile Parametro: Unità di misura associata alla variabile
	Esempio: Risultato:=Rcpt_add_ty_USInt_par (0,PLC_var1,'lunghezza','mm'); 	Restituisce 0 se la funzione è andata buon fine, restituisce 1 in caso di errore
Rcpt_add_ty_SInt_par	Aggiunge un parametro su 8 bit intero con segno alla lista.	Parametro: indice del parametro all'interno della ricetta Parametro: Nome della variabile PLC da inserire Parametro: Mnemonico associato alla variabile Parametro: Unità di misura associata alla variabile

	Esempio: Risultato:=Rcpt_add_ty_SInt_par (0,PLC_var2,'larghezza','mm');	Restituisce 0 se la funzione è andata buon fine, restituisce 1 in caso di errore
Rcpt_add_ty_UInt_par	Aggiunge un parametro su 16 bit intero senza segno alla lista.	Parametro: indice del parametro all'interno della ricetta Parametro: Nome della variabile PLC da inserire Parametro: Mnemonico associato alla variabile Parametro: Unità di misura associata alla variabile
	Esempio: Risultato:=Rcpt_add_ty_UInt_par (0,PLC_var1,'lunghezza','mm');	Restituisce 0 se la funzione è andata buon fine, restituisce 1 in caso di errore
Rcpt_add_ty_Int_par	Aggiunge un parametro su 16 bit intero con segno alla lista.	Parametro: indice del parametro all'interno della ricetta Parametro: Nome della variabile PLC da inserire Parametro: Mnemonico associato alla variabile Parametro: Unità di misura associata alla variabile
	Esempio: Risultato:=Rcpt_add_ty_Int_par (0,PLC_var2,'larghezza','mm');	Restituisce 0 se la funzione è andata buon fine, restituisce 1 in caso di errore
Rcpt_add_ty_UDInt_par	Aggiunge un parametro su 32 bit intero senza segno alla lista.	Parametro: indice del parametro all'interno della ricetta Parametro: Nome della variabile PLC da inserire Parametro: Mnemonico associato alla variabile Parametro: Unità di misura associata alla variabile
	Esempio: Risultato:=Rcpt_add_ty_UDInt_par (0,PLC_var1,'lunghezza','mm');	Restituisce 0 se la funzione è andata buon fine, restituisce 1 in caso di errore
Rcpt_add_ty_DInt_par	Aggiunge un parametro su 32 bit intero con segno alla lista.	Parametro: indice del parametro all'interno della ricetta Parametro: Nome della variabile PLC da inserire Parametro: Mnemonico associato alla variabile Parametro: Unità di misura associata alla variabile

	Esempio: Risultato:=Rcpt_add_ty_Dint_par (0,PLC_var2,'larghezza','mm');	Restituisce 0 se la funzione è andata buon fine, restituisce 1 in caso di errore
Rcpt_add_ty_Real_par	Aggiunge un parametro Reale alla lista.	Parametro: indice del parametro all'interno della ricetta Parametro: Nome della variabile PLC da inserire Parametro: Mnemonico associato alla variabile Parametro: Unità di misura associata alla variabile
	Esempio: Risultato:=Rcpt_add_ty_Real_par (0,PLC_var,'velocità','m/s');	Restituisce 0 se la funzione è andata buon fine, restituisce 1 in caso di errore
Rcpt_get_ty_USInt_par	Legge un parametro su 8 bit intero senza segno dalla lista.	Parametro: indice del parametro all'interno della ricetta
	Esempio: PLC_var1:=Rcpt_get_ty_USInt_par (0);	Restituisce il valore letto
Rcpt_get_ty_SInt_par	Legge un parametro su 8 bit intero con segno dalla lista.	Parametro: indice del parametro all'interno della ricetta
	Esempio: PLC_var2:=Rcpt_get_ty_SInt_par (1);	Restituisce il valore letto
Rcpt_get_ty_UInt_par	Legge un parametro su 16 bit intero senza segno dalla lista.	Parametro: indice del parametro all'interno della ricetta
	Esempio: PLC_var3:=Rcpt_get_ty_UInt_par (2);	Restituisce il valore letto
Rcpt_get_ty_Int_par	Legge un parametro su 16 bit intero con segno dalla lista.	Parametro: indice del parametro all'interno della ricetta
	Esempio: PLC_var4:=Rcpt_get_ty_Int_par (3);	Restituisce il valore letto
Rcpt_get_ty_UDInt_par	Legge un parametro su 32 bit intero senza segno dalla lista.	Parametro: indice del parametro all'interno della ricetta
	Esempio: PLC_var5:=Rcpt_get_ty_UDInt_par (4);	Restituisce il valore letto
Rcpt_get_ty_DInt_par	Legge un parametro su 32 bit intero con segno dalla lista.	Parametro: indice del parametro all'interno della ricetta
	Esempio: PLC_var6:=Rcpt_get_ty_DInt_par (5);	Restituisce il valore letto
Rcpt_get_ty_Real_par	Legge un parametro Reale dalla lista.	Parametro: indice del parametro all'interno della ricetta
	Esempio: PLC_var7:=Rcpt_get_ty_Real_par (6);	Restituisce il valore letto

Rcpt_get_param_comment	Legge il Nome Variabile dalla lista.	Parametro: indice del nome variabile all'interno della ricetta
	Esempio: ric_name:=Rcpt_get_param_comment (6);	Restituisce il valore letto
Rcpt_get_param_type	Legge il tipo della variabile dalla lista.	Parametro: indice del tipo della variabile all'interno della ricetta
	Esempio: type:=Rcpt_get_param_type (6); Elenco tipi: 1. 0 unsigned short (8 bit senza segno) 2. 1 signed short (8 bit con segno) 3. 2 unsigned int (16 bit senza segno) 4. 3 signed int (16 bit con segno) 5. 4 unsigned short (8 bit senza segno) 6. 5 signed double int (32 bit con segno) 7. 6 unsigned double int (32 bit senza segno) 7 float (reale)	Restituisce il tipo della variabile
Rcpt_get_param_um	Legge l'unità di misura della variabile i dalla lista.	Parametro: indice della variabile da leggere all'interno della ricetta
	Esempio: um:=Rcpt_get_param_um (6);	Restituisce il valore letto
Rcpt_get_curr_descr	Legge la descrizione della ricetta corrente.	Parametro: costante
	Esempio: description:=Rcpt_get_curr_descr(0);	Restituisce la descrizione associata alla ricetta
Rcpt_get_curr_param_count	Legge il numero di variabili che compongono la ricetta corrente.	Parametro: costante
	Esempio: Risultato:=Rcpt_get_curr_param_count(0);	Restituisce il numero di parametri letto
Rcpt_import	Importa tutte le ricette presenti su una chiave usb connessa al pannello	Parametro: indice che identifica il dispositivo usb da cui leggere le ricette, Parametro: percorso relativo della cartella sul dispositivo usb in cui si trovano le ricette da importare.
	Risulato:=Rcpt_import(1,temp/ricette);	Restituisce : 0 se l'operazione è andata a buon fine >0 altrimenti

Rcpt_export	Esporta tutte le ricette presenti sul pannello su un dispositivo usb ad esso connesso.	Parametro: indice che identifica il dispositivo usb da cui leggere le ricette, Parametro: percorso relativo della cartella sul dispositivo usb in cui si desiderano salvare le ricette
	Risulato:=Rcpt_export(1,temp/ricette);	Restituisce : 0 se l'operazione è andata a buon fine >0 altrimenti

16.6 FUNZIONI DATA E ORA

Sono disponibili due funzioni per la lettura e la scrittura della data e dell'ora corrente.

Nome	Descrizione/ esempio	Parametri di ingresso / Valore restituito
DateTimeRead	Esempio: Risultato:= DateTimeRead(0);	Aggiorna le variabili: TIME_HH, TIME_MM, TIME_SS, DATE_YY, DATE_MM, DATE_DD
DateTimeWrite	Esempio: Risultato:= DateTimeWrite(0);	Aggiorna il contenuto del RTC con i valori contenuti nelle variabili: TIME_HH, TIME_MM, TIME_SS, DATE_YY, DATE_MM, DATE_DD

16.7 FUNZIONI SERIALE MECT

Il TPAC ha la possibilità di interfacciarsi via seriale (RS232 o RS485) con strumenti MECT, sono perciò disponibili quattro funzioni che permettono d'interrogare o impostare i parametri degli strumenti.

Le funzioni restituiscono un codice che indica se l'operazione è andata a buon fine o meno. La tabella dei codici d'errore è riportata di seguito

0 - ACK -- successo
1 - NACK
9 - errore del formato dati
11 - errore TX ID
12 - errore codice di comando trasmesso
13 - errore del valore intero trasmesso
14 - errore del valore esadecimanle trasmesso
15 - errore di TX di un valore reale
16 - errore di TX del tipo di dato
21 - errore di TX
31 - errore RX BCC
32 - timeout
33 - errore RX
41 - stringa ricevuta non corretta
100 – errore sconosciuto

Nome	Descrizione/ esempio	Parametri di ingresso / Valore restituito
MECT_H_sread	Legge una variabile esadecimale da uno strumento MECT	Input: id:INT: indirizzo dello strumento da interrogare; command: STRING: Codice del comando dd:UDINT: Numero di digit del protocollo (6 o 8)
		Output: INT: Codice d'errore restituito dalla funzione
MECT_H_swrite	Scrive su una variabile esadecimale di uno strumento MECT	Input: **id**:UINT: Indirizzo dello strumento da interrogare; command: Codice del comando value:UINT: Valore della variabile dd:UDINT numero di digit del protocollo (6 o 8)
		Output: Codice d'errore restituito dalla funzione
MECT_sread	Legge una variabile ASCII da uno strumento MECT	Input: id:INT: indirizzo dello strumento da interrogare; command: STRING: Codice del comando dd:UDINT: Numero di digit del protocollo (6 o 8)
		Output: Codice d'errore restituito dalla funzione
MECT_swrite	Scrive su una variabile ASCII di uno strumento MECT	Input: **id**:UINT: Indirizzo dello strumento da interrogare; command: Codice del comando value:UINT: Valore della variabile dd:UDINT numero di digit del protocollo (6 o 8)8)
		Output: Codice d'errore restituito dalla funzione

Esempio MECT_H_sread()
Lettura di un valore esadecimale da seriale: error:=MECT_H_sread(1,'SW',6);
Il dato letto è inserito nella variabile PLC_MECT_enq.
Esempio MECT_sread()
Lettura di un valore ASCII da seriale: error:=MECT_sread(1,'RO',6);
Il dato letto è inserito nella variabile PLC_MECT_enq.
Esempio MECT_H_swrite()
Scrittura di un valore esadecimale su seriale: error:=MECT_H_swrite(1,'SW'16#0001,6);
Esempio MECT_swrite()

Scrittura di un valore ASCII su seriale: error:=MECT_swrite(1,'SW'16#0001,6);

16.8 FUNZIONI MODBUS

Il pannello operatore realizza la funzionalità di MODBUS master. La comunicazione è implementata su RS485 o RS232 secondo il protocollo MODBUS RTU. L'attivazione della funzionalità MODBUS e l'impostazione dei parametri per la configurazione della comunicazione seriale si effettuano dall'interfaccia web di configurazione del TPAC raggiungibile tramite browser all'indirizzo *http:// < indirizzo IP del TPAC>*.
Il TPAC rende disponibili le seguenti funzioni MODBUS:
01 Read Coil Status
02 Read Input Status
03 Read Holding Registers
04 Read Input Registers
05 Force Single Coil
06 Preset Single Register
15 Force Multiple Coil
16 Preset Multiple Registers
22 Mask Write 4X Register
23 Read/Write 4X Registers

Nome	Descrizione/Esempio	Parametri in ingresso/ Valore restituito
MB_f01	Legge lo stato ON/OFF delle uscite discrete con riferimento 0x (coils) nello slave. La funzione non supporta la modalità broadcast	INPUT: slave_addr: UINT indirizzo dello slave start_addr: UINT indirizzo iniziale della coil da leggere (la coil 1 in realtà inidirizza la coil 0) count UINT: numero di coil da leggere. E' possibile leggere al massimo 256 coil con un'unica operazione di lettura.
	Esempio:	OUTPUT: Valore ritornato: UINT 0 la richiesta è processata 1 se il modbus è occupato; Errori: Se presenti sono riportati in MODBUSstatus[1]; Valori letti: sono riportati nel vettore MODBUSdata MODBUSdata[0] = numero degli elementi letti MODBUSdata[i] = stato della coil 1 → ON , 0 → OFF ordinati dal più basso al più alto degli indirizzi richiesti.

MB_f02	Legge lo stato ON/OFF degli ingressi discreti con riferimento 1X nello slave. La funzione non supporta la modalità broadcast	INPUT: slave_addr: UINT indirizzo dello slave start_addr: UINT indirizzo iniziale dell'ingresso da leggere (l'ingresso 1 in realtà corrisponde all'ingresso 0) count UINT: numero di ingressi da leggere. E' possibile leggere al massimo 256 ingressi con un'unica operazione di lettura.
	Esempio:	OUTPUT: Valore ritornato: UINT 0 la richiesta è processata 1 se il modbus è occupato; Errori: Se presenti sono riportati in MODBUSstatus[1]; Valori letti: sono riportati nel vettore MODBUSdata MODBUSdata[0] = numero degli elementi letti MODBUSdata[i] = stato degli ingressi 1 → ON , 0 → OFF ordinati dal più basso al più alto degli indirizzi richiesti.
MB_f03	Legge i registri con riferimento 4XXXX e ne rende disponibile il contenuto nel vettore MODBUSdata. La funzione non supporta il broadcast.	INPUT: slave_addr: UINT indirizzo dello slave start_addr: UINT indirizzo iniziale del registro da leggere (l'indirizzo 40001 in realtà corrisponde al registro 40000) count UINT: numero di registri da leggere al massimo 256 byteorder: USINT ordinamento dei byte nella risposta dello slave (il protocollo RTU prevede prima il byte ALTO e poi quello BASSO ma alcuni controller adottano la convenzione inversa) 0 byte ALTO – byte BASSO 1 byte BASSO – byte ALTO

	Esempio:	OUTPUT: Valore ritornato: UINT 0 la richiesta è processata 1 se il modbus è occupato; Errori: Se presenti sono riportati in MODBUSstatus[1]; Valori letti: sono riportati nel vettore MODBUSdata MODBUSdata[0] = numero degli elementi letti MODBUSdata[i] = contenuto del registro
MB_f04	Legge i registri con riferimento 3XXXX e ne rende disponibile il contenuto nel vettore MODBUSdata. La funzione non supporta il broadcast.	INPUT: slave_addr: UINT indirizzo dello slave start_addr: UINT indirizzo iniziale del registro da leggere (l'indirizzo 30001 in realtà corrisponde al registro 30000) count: UINT numero di registri da leggere al massimo 256 byteorder: USINT ordinamento dei byte nella risposta dello slave (il protocollo RTU prevede prima il byte ALTO e poi quello BASSO ma alcuni controller adottano la convenzione inversa) 0 byte ALTO – byte BASSO 1 byte BASSO – byte ALTO
	Esempio	OUTPUT: Valore ritornato: UINT 0 la richiesta è processata 1 se il modbus è occupato; Errori: Se presenti sono riportati in MODBUSstatus[1]; Valori letti: sono riportati nel vettore MODBUSdata MODBUSdata[0] = numero degli elementi letti MODBUSdata[i] = contenuto del registro

MB_f05	Forza lo stato ON /OFF nella singola coil indirizzata (riferimento 0X). La funzione supporta il broadcast, in tal caso lo stesso stato verrà forzato sulla coil indirizzata per tutti gli slave presenti nella rete.	INPUT: slave_addr: UINT indirizzo dello slave coil_addr: UINT indirizzo della coil da scrivere (la coil 1 in realtà inidirizza la coil 0) coil_state UINT: 0 per forzare la coil a OFF, 1 per forzare la coil a ON.
	Esempio:	OUTPUT: Valore ritornato: UINT 0 la richiesta è processata 1 se il modbus è occupato; Errori: Se presenti sono riportati in MODBUSstatus[1];
MB_f06	Imposta un valore all'interno di un registro (riferimento 4X). La funzione supporta il broadcast, in tal caso lo stesso valore viene impostato nel registro indirizzato per tutti gli slave presenti sulla rete.	INPUT: slave_addr: UINT indirizzo dello slave reg_addr: UINT indirizzo del registro da impostare. Il valore da scrivere nel registro viene letto in MODBUSdata[0]
	Esempio:	OUTPUT: Valore ritornato: UINT 0 la richiesta è processata 1 se il modbus è occupato; Errori: Se presenti sono riportati in MODBUSstatus[1];

MB_f0F	Forza lo stato ON/OFF in un insieme di coil i cui indirizzi siano contigui. La funzione supporta il broadcast, in tal caso lo stesso stato viene forzato per le coil indirizzate in tutti gli slave presenti nella rete.	INPUT: slave_addr: UINT indirizzo dello slave coil_addr: UINT indirizzo iniziale delle coil da impostare. coil_count: UINT numero delle coil da impostare Il valori da impostare per le coils vengono letti da MODBUSdata: stato (0→ OFF, 1→ON) MODBUSdata[0] → stato per la coil all'indirizzo coil_addr MODBUSdata[1] → stato per la coil all'indirizzo coil_addr+1 MODBUSdata[N] → stato per la coil all'indirizzo coil_addr+N
	Esempio:	OUTPUT: Valore ritornato: UINT 0 la richiesta è processata 1 se il modbus è occupato; Errori: Se presenti sono riportati in MODBUSstatus[1];
MB_f10	Imposta il valore di un insieme di registri (riferimento 4X) i cui indirizzi siano contigui. La funzione supporta il broadcast, in tal caso lo stesso valore viene impostato per i registri indirizzati in tutti gli slave presenti nella rete.	INPUT: slave_addr: UINT indirizzo dello slave start_addr: UINT indirizzo del primo registro da impostare count: UINT numero di registri da impostare I valori da impostare per registri vengono letti MODBUSdata[0] → valore da impostare per il registro all'indirizzo start_addr MODBUSdata[1] → valore da impostare per il registro all'indirizzo start_addr+1 MODBUSdata[N] → valore da impostare per il registro all'indirizzo start_addr +N

	Esempio:	OUTPUT: Valore ritornato: UINT 0 la richiesta è processata 1 se il modbus è occupato; Errori: Se presenti sono riportati in MODBUSstatus[1];
MB_f16	Modifica il contenuto di uno specifico registro (riferimento 4X) utilizzando una combinazione di una maschera di AND e una maschera di OR. La funzione può essere utilizzata per impostare un bit in un particolare registro. Il broadcast non è supportato.	INPUT: slave_addr: UINT indirizzo dello slave start_addr: UINT indirizzo del registro da impostare I valori delle maschere vengono letti dal vettore MODBUSdata MODBUSdata[0] → machera AND MODBUSdata[1] → maschera OR
	Esempio	OUTPUT: Valore ritornato: UINT 0 la richiesta è processata 1 se il modbus è occupato; Errori: Se presenti sono riportati in MODBUSstatus[1];

MB_f17	Effettua una operazione combinata di lettura e scrittura . La funzione legge/scrive da/in un insieme contiguo di registri (riferimento 4X) . Il broadcast non è supportato.	INPUT: slave_addr: UINT indirizzo dello slave start_addr: UINT indirizzo iniziale dei registri da leggere count: UINT numero dei registri da leggere reg_addr: UINT indirizzo iniziale dei registri da scrivere reg_count: UINT numero dei registri da scrivere byteorder: USINT ordinamento dei byte nella risposta dello slave (il protocollo RTU prevede prima il byte ALTO e poi quello BASSO ma alcuni controller adottano la convenzione inversa) 0 byte ALTO – byte BASSO 1 byte BASSO – byte ALTO I valori da impostare per registri vengono letti MODBUSdata[0] → valore da impostare per il registro all'indirizzo reg_addr MODBUSdata[1] → valore da impostare per il registro all'indirizzo reg_addr+1 MODBUSdata[N] → valore da impostare per il registro all'indirizzo reg_addr +N

	Esempio	Valore ritornato: UINT 0 la richiesta è processata 1 se il modbus è occupato; Errori: Se presenti sono riportati in MODBUSstatus[1]; Valori letti: sono riportati nel vettore MODBUSdata MODBUSdata[0] = numero degli elementi letti MODBUSdata[1] = contenuto del registro all'indirizzo start_addr MODBUSdata[2] = contenuto del registro all'indirizzo start_addr+1 MODBUSdata[N] = contenuto del registro all'indirizzo start_addr+N-1

Esempio funzione MODBUS:

```
SLAVE_ADD:=1;
START_ADD:=40689;
COUNT:=12;
ORDER:=0;

IF MODBUSstatus[0] = 0 THEN
     MB_READ1:=MB_f03(SLAVE_ADDRESS, START_ADD, COUNT, ORDER);
END_IF;

IF MB_READ1 = 0  THEN
(* FREQUENZA DI RETE*)
     Hz:=MODBUSdata[5]/10;
(* FATTORE DI POTENZA*)
     PF_Tot:=TO_USINT(MODBUSdata[8]/10);
END_IF;
```

16.9 FUNZIONI CanOpen

Sono disponibili due funzioni per la lettura e la scrittura di SDO sulle reti CanOpen. Le funzioni restituiscono un codice che indica se l'operazione è andata a buon fine o meno. La tabella dei codici d'errore è riportata di seguito

0 - OK
1 – nome della variabile non valido
2 – la variabile non esiste
3 – numero canale CAN non valido
126 – il nodo non è in running
127 – operazione non ancora disponibile

Nome	Descrizione/ esempio	Parametri di ingresso / Valore restituito
SDORead	Legge una variabile da un nodo CanOpen via SDO	Input: ch: UINT:Rete CanOpen (0 o 1) da interrogare var_name: STRING: Stringa contenente il nome della variabile come definita nel programma di configurazione della rete CanOpen Netbuilder
		Output: Codice d'errore
SDOWrite	Scrive su una variabile di un nodo CanOpen via SDO	Input: ch: UINT:Rete CanOpen (0 o 1) da interrogare var_name: STRING: Stringa contenente il nome della variabile come definita nel programma di configurazione della rete CanOpen Netbuilder data: UDINT: Valore da scrivere
		Output: Codice d'errore

Esempio SDORead()
Si vuole leggere la variabile pr80 da un nodo posto sulla rete CanOpen numero 1

```
IF can_var= 0 THEN
     ris:=SDORead(1,'pr80');
     can_var:=1;
END_IF;
IF ris =0 THEN
     IF SDOStatus > 0 AND can_var = 1 THEN
          count_can:=count_can+1;
          can_var:=0;
          PR80_2:=SDODATA;
     END_IF;
END_IF;
```

Il valore letto dal nodo, a seguito di una funzione SDORead, è salvato nella variabile SDOData, questo però non è immediatamente disponibile, in quanto il nodo impiega un certo tempo per rispondere alla richiesta. Il programmatore perciò dovrà controllare la variabile SDOStatus, che è posta a 0 a seguito dell'invio del comando, e va a 1 quando nel registro SDOData è disponibile il valore restituito dal nodo interrogato. Il valore di SDOStatus torna a zero a seguito di una lettura del registro SDOData.

Esempio SDOWrite()

Volendo scrivere il valore 100 nella variabile pr81 di un nodo CanOpen posto sulla rete 0 si potrebbe procedere come segue:

```
IF SDOStatus = 0 THEN
     ris:=SDOWrite(1,'pr81',100);
END_IF;
```

Prima di poter effettuare una scrittura è necessario verificare che la precedente scrittura sia andata a buon fine e che quindi il registro SDOStatus sia a zero;

16.10 FUNZIONI DI IMPORTAZIONE DI UN FILE DI TESTO IN HMI

Se in un progetto HMI si vuole visualizzare uno o più file di testo è necessario richiamare la funzione HMItextLoad La funzione restituisce un codice che indica se l'operazione è andata a buon fine o meno. La tabella dei codici d'errore è riportata di seguito:
0 - success
1 - error reading file contents
2 - invalid file name
3 - file does not exist
4 - file cannot be open

Nome	Descrizione/ esempio	Parametri di ingresso / Valore restituito
HMItextLoad		Input: Nome del file da leggere Output: codice d'errore

Se la funzione restituisce zero, la variabile **FILE_READ** contiene l'indirizzo del file letto. Per mostrare a video il contenuto del file è necessario crearsi una variabile stringa sufficientemente grande d contenere il file ed assegnargli il puntatore.

L'esempio seguente mostra l'utilizzo della funzione HMItextLoad: definiamo la variabile contenuto_file come una variabile STRING di 500 elementi nella quale salveremo il contenuto letto dal file.
Se chiamando la funzione questa restituisce zero assegnamo il valore presente in FILE_READ alla variabile contenuto_file, altrimenti scriviamo un messaggi d'errore.

```
Risultato:= HMItextLoad('istruzioni_italiano ');

IF Risutato =  0 THEN
     Contenuto_file:=FILE_READ;
ELSE
     Contenuto_file:=' file non presente';
END_IF;
```

A seguito di queste operazioni la variabile contenuto_file contiene: o il valore del file letto o il messaggio d'errore, per mostrarlo a video è sufficiente associare la variabile ad una TextBox in HMI, inserendo nella proprietà "Assoc string" la variabile contenuto_file.

16.11 DATALOGGER

Sono disponibili alcune funzioni per il salvataggio di dati su un disco esterno montato sulla porta USB. In questo modo è possibile registrare i valori delle variabili o eventi che si sono verificati durante il funzionamento del programma. I dati sono salvati in modalità di testo e quindi facilmente importabili in un programma di foglio di calcolo per una successiva elaborazione. Ogni funzione restituisce un codice che indica se l'operazione è andata a buon fine o meno. La tabella delle funzioni e dei codici d'errore è riportata di seguito:

Nome	Descrizione	Parametri di ingresso / Valore restituito
Datalog_start	Effettua il caricamento del disco USB sul TPAC e verifica se lo spazio richiesto è sufficiente	Input: BUFF_LENGHT : UDINT indica lo spazio richiesto per salvare i dati sul disco CHANNELS: UDINT indica il numero di variabili da registrare INDEX: UINT indica il device USB sui cui salvare i dati tra quelli eventualmente disponibili (da 1 a 4)
		Output: Codice d'errore DINT -1 - OK -2 – il disco USB non può essere collegato -3 – Non è possibile rilevare lo spazio disponibile su disco -4 – Non è possibile creare la directory per il salvataggio dei dati >=0 - MAX numero massimo di byte massimi disponibili su disco
Datalog_get	Salva il valore di una variabile su disco	Input: LABEL : STRING: Stringa che indica l'etichetta da associare ai dati in pratica l'intestazione della colonna dei dati VALUE: REAL: Il nome della variabile da salvare su disco.
		Output: Codice d'errore: DINT * 0 - OK * 1 errore
Datalog_stop	Termina il processo di logging e crea il file contenete i dati acquisiti	Input: FILE: STRING: Stringa che indica il nome del file contenete i dati aggregati
		Output: Codice d'errore: DINT: * 0 OK * >0 errore
		Output: SINT * 0 – OK * > 0 – Errore

Operativamente il programmatore deve conoscere la quantità di spazio necessario per salvare i dati per il tempo richiesto, sapendo che ogni campione occupa 100byte.

Dovendo per esempio effettuare un campionamento di 5 variabili alla frequenza di un campione ogni quattro secondi per tre ore, sarà necessario che sul disco sia disponibile almeno uno spazio di:

100byte*5variabili*3*3600secondi/4secondi=1350000 byte.

Se dopo le tre ore il processo di registrazione non viene interrotto, i dati acquisiti in precedenza saranno sovrascritti secondo la logica di un buffer circolare.
Se si esegue una funzione Dalaog_start() e lo spazio su disco non è sufficiente la funzione restituisce la dimensione massima disponibile su disco.

Effettuato il caricamento del disco, e verificato lo spazio disponibile, con la frequenza richiesta (nell'esempio una volta ogni quattro secondi), si richiama per ciascuna variabile la funzione Datalog_get(), indicando come parametri l'etichetta associata e il nome della variabile da registrare.

Terminato il processo di registrazione si chiama la funzione Datalog_stop() indicando il nome del file nel quale aggregare i dati acquisiti.

I dati acquisiti sono disponibili sia nel file indicato nella funzione Datalog_stop() come dati aggregati, sia nella cartella tempdata dove vi un file per ciascuna variabile registrata. Il nome del file è quello dell'etichetta definita nella funzione Datalog_get();
Si riporta di seguito un esempio di programma in ST, per a acquisire e salvare i dati di due variabili:

```
(************        variables                                    ************)
VAL1:=(AIN_CH1-4.0)*600.0/16.0;
VAL2:=(AIN_CH2-4.0)*600.0/16.0;

(************        mounting disk                                ************)

IF stato = 0 THEN
num_ch:=2;                          (*canali da acquisire*)
acq_rate:=4;                    (* acquisizione ogni 4 secondi*)
acq_dur:=2;                         (* durata acquisizione due ore*)
(* lunghezza_buffer = 1800*)
        lunghezza_buffer:= acq_dur*3600/ acq_rate;
        ris:=Datalog_start(lunghezza_buffer, num_ch); (* richiesta spazio su disco*)
        stato:=1;
END_IF;

(************        setup samples number                         ************)

IF  stato = 1 THEN
IF ris < 0 THEN       (* errore di caricamento disco*)
stato:=3;
ELSIF ris >0 THEN                   (* spazio su disco non sufficiente *)
        max_camp:=ris;
        stato:=2 ;
ELSE
        stato:=2;
        max_camp:= lunghezza_buffer;
```

```
END_IF;
TIMERS[1]:=4000;
campioni:=0;
END_IF;

(************          data recording          ************)

IF stato = 2 THEN
IF  TIMERS[1]=0 AND campioni < max_camp THEN
        TIMERS_FLAG[1]:=1;
        campioni:=campioni+1;
        dummy:=Datalog_get('Temp_ch1',VAL1);
dummy:=Datalog_get('Temp_ch2',VAL2);
ELSIF campioni >=max_camp THEN
        stato:=4;
END_IF;
END_IF;

(************          stop recording          ************)

IF stato=4 THEN
        ris:=Datalog_stop('dati.txt');
        stato:=5;
END_IF;

IF stato=5 THEN
IF  ris=0 THEN
        stato:=6;
ELSIF ris<0 THEN
        Stato:=7;
END_IF;
END_IF;
```

16.11.1 CRYPTO-DATALOGGER – CFR21/11

La funzionalità di salvataggio dati è disponibile anche secondo lo standard CFR21/11. La funzionalità di cripto –datalogger opera come descritto per il datalogger standard ma, diversamente dal primo i dati, prima di essere memorizzati sul disco esterno, sono cifrati in memoria.
La decifratura dei dati stessi è possibile solo tramite la conoscenza della chiave utilizzata per cifrare i dati e della password di autorizzazione garantendo così l'inalterabilità dei dati. I dati, una volta decifrati sono disponibili in formato testo per l'importazione in un programma di foglio di calcolo per la successiva eventuale elaborazione.

Nome	Descrizione	Parametri di ingresso / Valore restituito

CryptoDatalog_start	Effettua il caricamento del disco USB sul TPAC e verifica se lo spazio richiesto è sufficiente, inizializza l'ambiente di cifratura.	Input: BUFF_LENGHT : UDINT indica lo spazio richiesto per salvare i dati sul disco CHANNELS: UINT indica il numero di variabili da registrare INDEX: UINT indica il dispositivo USB sui cui salvare i dati tra quelli eventualmente disponibili (da 1 a 4) KEY: STRING stringa di 16 caratteri utilizzata come chiave della fase di cifratura dei dati PWD: STRING stringa di 8 caratteri utilizzata come password per la protezione dei dati.
		Output: Codice d'errore DINT -1 - OK -2 – il disco USB non può essere collegato -3 – Non è possibile rilevare lo spazio disponibile su disco -4 – Non è possibile creare/cancellare le directory per il salvataggio dei dati >=0 - MAX numero massimo di byte massimi disponibili su disco
CryptoDatalog_get	Salva il valore cifrato di una variabile su disco	Input: LABEL : STRING: Stringa che indica l'etichetta da associare ai dati in pratica l'intestazione della colonna dei dati VALUE: REAL: Il nome della variabile da salvare su disco.
		Output: Codice d'errore: INT * 0 - OK * 1 errore
CryptoDatalog_stop	Termina il processo di registrazione. Sul dispositivo usb sono disponibili, nella cartella temprata/ i log delle variabili cifrati.	Input: costante
		Output: Codice d'errore: UINT: * 0 OK * >0 errore

CryptoDatalog_decrypt	Decifra i log delle variabili e crea il log complessivo in formato testo.	Input: **FILE**: STRING nome del file di log complessivo **SRC_INDEX**: USINT indica il dispositivo USB su cui si trovano i dati da copiare (da 1 a 4). **DST_INDEX**: USINT indica il dispositivo USB su cui si trovano i dati da copiare (da 1 a 4). **KEY**: STRING stringa di 16 caratteri utilizzata in fase di cifratura **PWD:** STRING stringa di 8 caratteri utilizzata come password per la protezione dei dati in fase di cifratura.
		Output: Codice d'errore: INT * 0 OK * >0 Errore
CryptoDatalog_eject	Smonta la chiave USB	Input: **INDEX**: UINT indica il dispositivo USB sui cui salvare i dati tra quelli eventualmente disponibili (da 1 a 4)
		Output: INT * 0 – OK * > 0 – impossibile smontare il dispositivo USB

16.12 FUNZIONI STRINGA

Sono disponibili due funzioni per la manipolazione delle stringhe

Nome	Descrizione/ esempio	Parametri di ingresso / Valore restituito
Str_concat()	Effettua il concatenamento di due stringhe	Input: IN1:STRING: stringa alla quale accodare Stringa2 IN2:STRING: stringa da accodare a Stringa1
		Output: BOOL: risultato funzione FALSE OK TRUE FAIL
Str_inttostr()	Converte un intero in stringa	Input: X:UDINT: intero da convertire DEST:STRING: stringa risultato WIDTH:USINT: numero di cifre da copiare nella stringa ZERI:BOOL: indica se deve effettuare il riempimento delle cifre mancanti con zeri

	Ris:=Str_inttostr(34,string1,3,TRUE); Risultato: la stringa string1 conterrà ‘034’	Output: BOOL: risultato funzione FALSE OK TRUE FAIL

Esempio: Str_concat():
Devono essere definite due stringhe: String1 e String2 da concatenare, ed un intero per contenere il risultato della funzione.

```
String1:=’abc’;
String2:=’defg’;
Ris:=Str_concat(String1,String2);
```

Se Ris =0, la stringa String1 conterrà il valore ‘abcdefg’

Esempio: Str_inttostr():
Devono essere definite una stringa, ed un intero per contenere il risultato della funzione.

Si vuole convertire in stringa il numero 34 ed il risultato deve essere espresso su tre cifre inserendo degli zeri per il completamento della stringa

Ris:=Str_inttostr(34,string1,3,TRUE);

Se Ris=0 la stringa String1 conterrà il valore :’034’

16.13 FUNZIONI DATA

Somma e differenza tra due date.

Nome	Descrizione/ esempio	Parametri di ingresso / Valore restituito
Date_add()	Somma un offset di giorni, settimane o mesi ad una data e restituisce una stringa con la data calcolata	TYPE: STRING: indica se si devono aggiungere ‘D’, settimane ‘W’, ‘M’ mesi alla data indicata nella stringa START START:STRING stringa che contiene la data alla quale sommare l’offset. La data deve essere espressa in giorno mese anno: gg-mm-aaaa OFFS:UDINT: valore numerico che indica quanto bisogna aggiungere
		Output: STRING: stringa nella quale inserire la data ottenuta
Date_diff()	Effettua la differenza tra due date e restituisce un valore espresso in giorni, settimane o mesi.	START: STRING: prima data END: STRING: seconda data TYPE: STRING: tipo di risultato

		Output: DINT: contiene la differenza tra le due date espressa con l'unità di tempo indicata tra i parametri.

Esempio Date_add()
Deve essere definita la stringa che conterrà il risultato della funzione.
Si aggiungono 2 settimane alla data indicata con gg-mm-aaaa.

```
String1:=Date_add('W','06-03-2008',2);
```

La funzione inserisce nella variabile String1 il risultato: '20-03-2008'.

Se il formato della data in ingresso è errato (p.e. la notazione invece che essere gg-mm-aaaa, è mm-gg-aaaa) nella stringa di risultato sarà inserito il valore 'Error'

Esempio Date_diff()
Si definisce una variabile DINT per contenere il risultato.

Si calcola il numero di giorni tra il '25-03-2008' e il '18-03-2008'.

```
Ris:=Date_diff('25-03-2008', '18-03-2008','D') ;
```

Nella variabile Ris si troverà il valore –8.

17.0 TPAC Remote Options

Il pannello operatore TPAC a partire dalla release software 3.0 include le seguenti nuove funzionalità:
Remote Desktop: accesso remoto all'interfaccia HMI
Monitor: visualizzazione dello stato delle variabili PLC
Remote PLC: creazione di un plc remoto o interazione remota con il PLC a bordo pannello per la gestione degli I/O di sistema.
L'accesso a tali funzioni avviene tramite connessione di rete pertanto richiede la configurazione di un indirizzo IP per il pannello operatore TPAC. È possibile utilizzare tali funzioni oltre che dalla LAN in cui è inserito il TPAC anche da una rete diversa previa opportuna configurazione.

NOTA: Il manuale utilizza, nell'esemplificazione delle funzionalità remote del TPAC, l'indirizzo IP di defualt assegnato al pannello: 192.168.0.210. Nella normale operatività è necessario sostituire tale IP con l'indirizzo assegnato al TPAC nella propria LAN o l'eventuale indirizzo assegnato per accessi da altre reti.

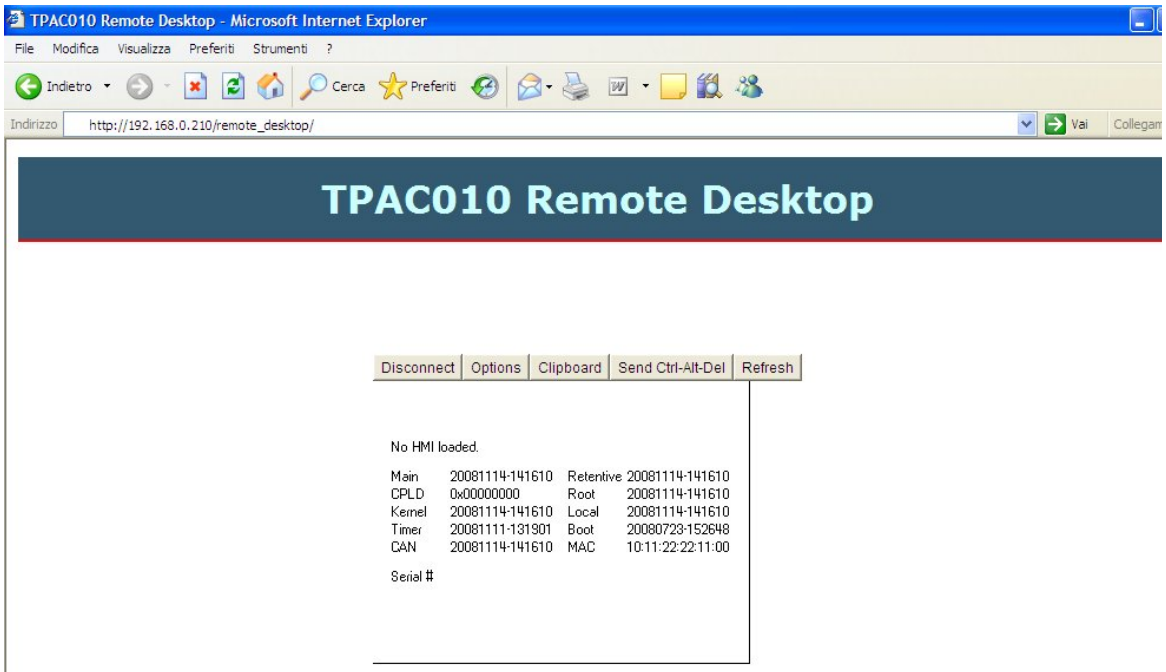
17.1 Remote Desktop

Il pannello operatore TPAC offre la funzionalità di accesso remoto WEB all'interfaccia HMI realizzata sul pannello. Tale accesso consente all'operatore di agire sull'interfaccia HMI come se si stesse operando fisicamente sul touchscreen del pannello utilizzando il proprio mouse.

17.2 Accesso al Remote Desktop

Per utilizzare la funzionalità di Remote Desktop del TPAC si apra il proprio browser (Internet Explorer 6.0 o superiore, o Mozilla firefox 2.0 o superiore) e digitare nella barra dell'indirizzo la seguente URL:
http://192.168.0.210/remote_desktop/

Nella pagina che appare nel browser inserire la password di accesso al pannello e premere OK. La pagina HMI visualizzata sul pannello sarà visibile anche nel browser . Utilizzando il mouse si portá interagire sulla interfaccia HMI visualizzata nel browser come se si stesse operando direttamente sul pannello. Per interrompere la connessione con il pannello premere “Disconnect”.



17.3 Impostazione della password di accesso al Remote Desktop

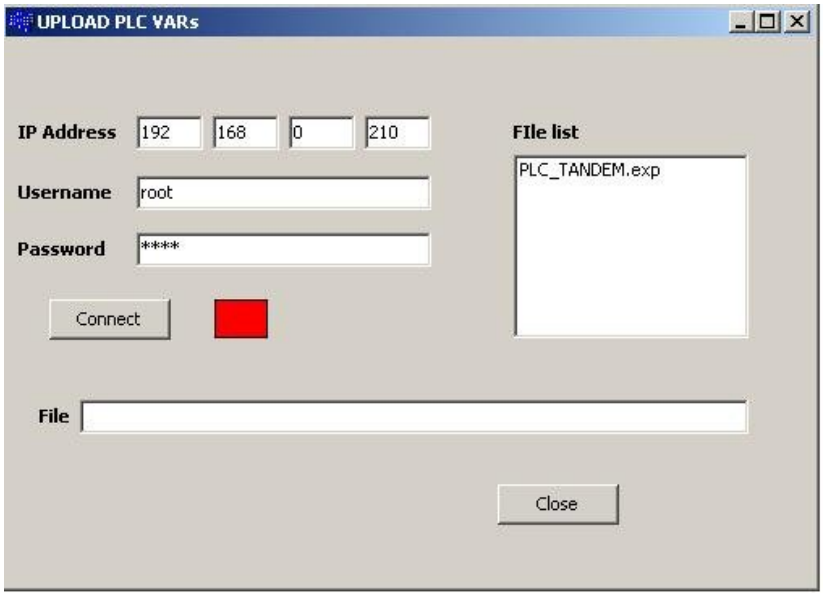
La funzionalità di Remote Desktop è accessibile per default con password “root”. Per impostare una password diversa digitare nel proprio browser l’indirizzo http://192.168.0.210/remote_desktop/ e seguire il link per accedere alla pagina di modifica password. Nella sezione Monitor Password inserire la vecchia password , la nuova password da impostare e premere il pulsante SET. La nuova password è immediatamente attiva e sarà necessaria per ogni accesso al Remote Desktop successivo alla sua impostazione. La password è modificabile un numero indefinito di volte.

17.4 Monitor

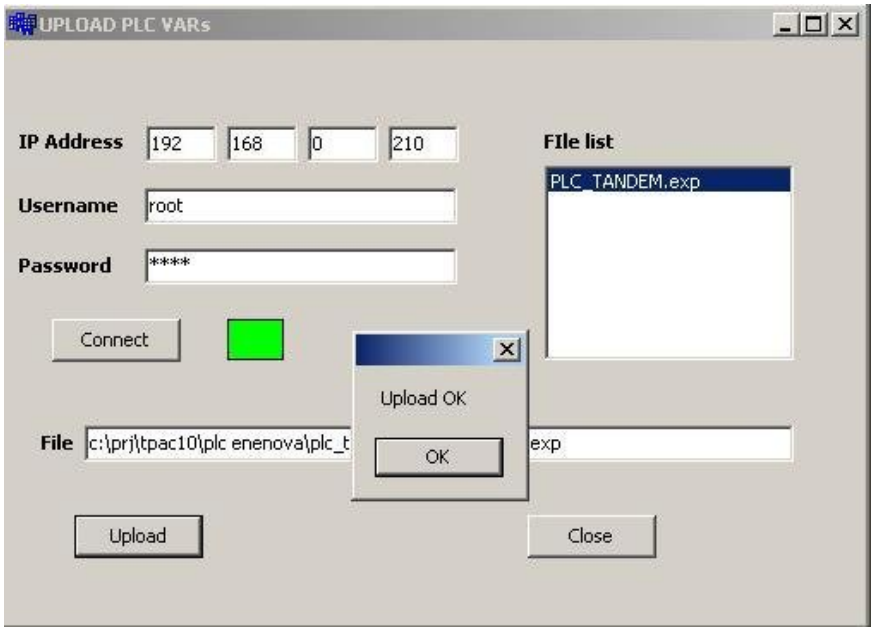
Il pannello operatore TPAC permette di visualizzare il valore delle variabili utilizzate dal PLC presente sul pannello. Tale funzionalità permette la lettura dello stato degli I/O a bordo pannello e delle variabili PLC create dal programmatore.

17.5 Upload delle variabili

Prima di poter leggere le variabili presenti sul TPAC è necessario sincronizzare la mappa delle variabili presente sul TPAC con quelle create all’interno del progetto PLC, perciò prima di poter essere lette da remoto le variabili devono essere esportate sul TPAC.
A seguito di una compilazione, il software LogicLab crea un file con estensione .exp, questo file deve essere inviato al TPAC per sincronizzare la mappa delle variabili. Per effettuare l’upload, dal menu Tools di LogicLab selezionare upload_var, che apre la seguente finestra

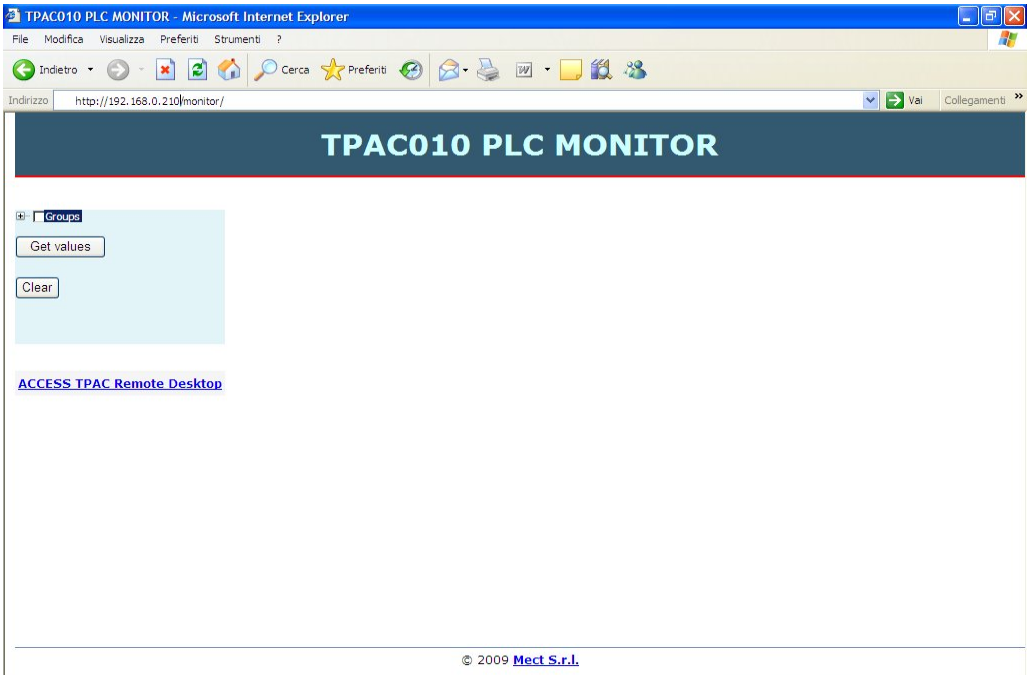


L’interfaccia propone: l’indirizzo IP ricavato dal progetto (se nel frattempo lo si è modificato inserire l’indirizzo corretto); la username e la password dell’ultimo accesso; la lista dei file .exp trovati nella directory del progetto corrente, (generalmente solo uno). Impostati i valori dell’indirizzo IP, della username e della password, premere il tasto Connect, quando si stabilisce la connessione, il led si colora di verde. Selezionato quindi il file da inviare, appare il Pulsante Upload. Premere Upload e in caso di corretta trasmissione apparirà la message box Upload OK.



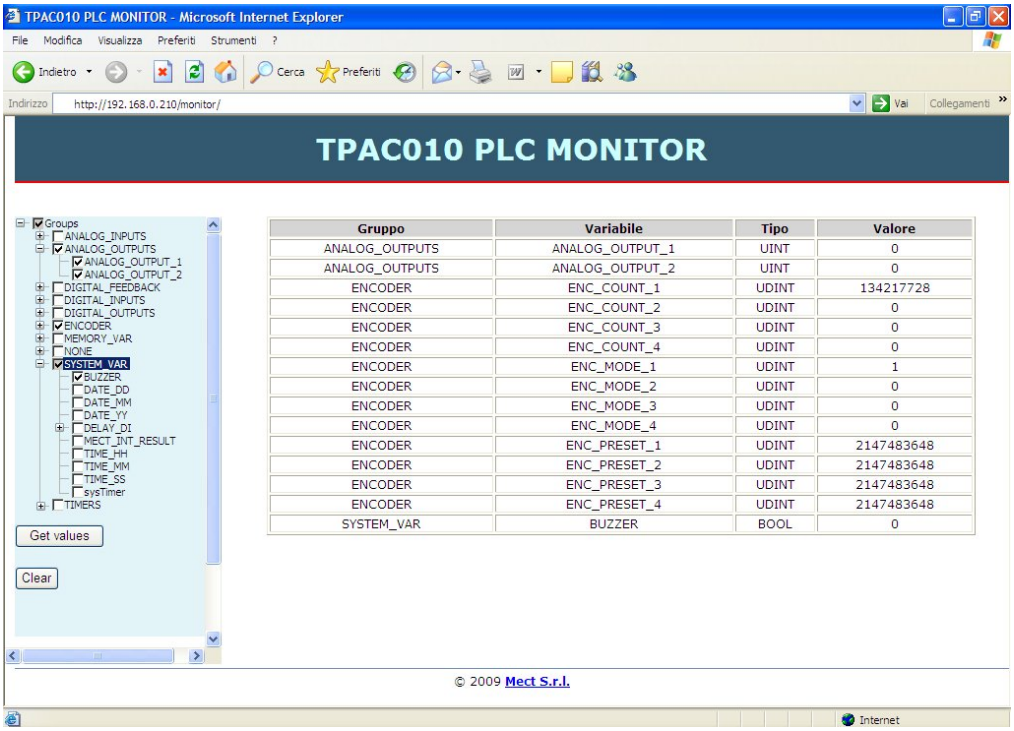
17.6 Accesso al Monitor

Per accedere alla funzionalità di monitor digitare nel proprio browser l'indirizzo http://192.168.0.210/monitor/. Si visualizzerà la seguente pagina web in cui nella sezione sinistra della pagina è disponibile l'elenco degli I/O presenti a bordo pannello e delle variabili PLC suddivise nei gruppi logici definiti dal programmatore.



Per visualizzare lo stato di un I/O o di una variabile o di un gruppo di variabili espandere l'albero delle grandezze disponibili fino ad individuare gli elementi di cui si desidera visualizzare il valore. È possibile selezionare singole variabili, selezionando la checkbox a sinistra del nome della variabile, o direttamente interi gruppi selezionando la checkbox accanto al nome del gruppo.

Terminata la fase di selezione premere il tasto “GET VALUES” per ottenere il valore delle grandezze selezionate.
Modificare o mantenere la selezione effettuata e premere nuovamente il tasto “GET VALUES” per aggiornare i valori delle grandezze visualizzate. Per annullare la selezione di una variabile o di un gruppo cliccare nuovamente sulla checkbox alla sinistra del nome della variabile o del gruppo.
Il tasto “CLEAR” consente di cancellare la visualizzazione dei valori delle grandezze selezionate.



17.7 Remote PLC

Il pannello operatore TPAC tramite la libreria PLCLIB , un insieme di funzioni messe a disposizione del programmatore permette di:
creare un’applicazione su PC che acceda al sistema I/O del TPAC: ingressi/uscite analogiche/digitali sia locali sia su bus CAN.
Interagire con il PLC programmato a bordo pannello utilizzando un set di apposite variabili scrivibili da remoto.

La libreria PLCLIB, è disponibile in formato windows dll sia OMF sia COFF.

17.8 Descrizione di PLCLIB

La libreria PLCLIB esporta tramite l’header plclib.h i tipi di dato e le funzioni di seguito riportate.

Tipi di dato

plcuservar_t

Il tipo plcuservar_t è utilizzato per allocare in memoria l’array contenente le variabili associate al PLC con cui si è stabilita la comunicazione. Tale array, quando non più necessario, deve essere eliminato dalla memoria a cura del programmatore.
Il tipo plcuservar_t è strutturato come segue:

```
Typedef struct plcuservar {
      char name[32];
      char type[16];
}plcuservar_t;
```

Il campo name contiene il nome della variabile ed è utilizzato per le operazioni di lettura e scrittura. Per la descrizione completa del significato di ciascuna variabile e del suo utilizzo si rimanda alla documentazione del PLC.
Il campo type contiene il tipo utilizzato nella rappresentazione della variabile. I tipi disponibili sono elencati nella tabella seguente.

TIPO	DIMENSIONE (BYTE)	RANGE
BOOL	1	Da 0 a 1
SINT	1	Da –128 a +127
USINT	1	Da 0 a 255
INT	2	Da –32768 a + 32767
UINT	2	Da 0 a 65535
DINT	4	Da –2^31 a + 2^31 – 1
UDINT	4	Da 0 a 2^32 – 1
BYTE	1	Da 0 a 255
WORD	2	Da 0 a 65535
DWORD	4	Da 0 a 2^32 – 1
STRING[N]	N	Stringa terminata con ‘\0’

17.9 Funzioni

Plc_init

NOME	plc_init
PROTOTIPO	int plc_init(char *hostname)
DESCRIZIONE	Inizializza la comunicazione con il plc a bordo pannello.
PARAMETRI	Char * hostname: può essere sia il nome macchina del pannello, sia il suo indirizzo IPv4 nella notazione standard IP1.IP2.IP3.IP4
VALORI RITORNATI	0 se l'inizializzazione della comunicazione è fallita; intero positivo descrittore del canale di comunicazione con il PLC per inizializzazione andata a buon fine.

Plc_release

NOME	plc_release
PROTOTIPO	int plc_release(int plcID)
DESCRIZIONE	Chiude la comunicazione con il plc a bordo pannello.
PARAMETRI	Int plcID: descrittore del canale di comunicazione con il PLC ottenuto in fase di inizializzazione
VALORI RITORNATI	0 se la comunicazione si è conclusa correttamente intero negativo in caso di errore

Plc_var_list

NOME	plc_var_list
PROTOTIPO	int plc_var_list(int plcID, plc_uservar_t **list)
DESCRIZIONE	Fornisce l'elenco delle variabili definite per il PLC cui e' associato il descrittore plcID e le memorizza in una struttura di tipo plc_uservar_t.

PARAMETRI	Int plcID: descrittore del canale di comunicazione ottenuto in fase di inizializzazione plcuservar_t ** puntatore all'array in memoria che contiene l'elenco delle variabili.
VALORI RITORNATI	numero delle variabili lette se l'operazione si è conclusa con successo intero negativo o zero in caso di errore

Plc_var_read

NOME	plc_var_read
PROTOTIPO	int plc_var_read(int plcID, char *label void *data)
DESCRIZIONE	Legge il valore della variabile identificata dalla label e lo memorizza nel buffer data definito dall'utente.
PARAMETRI	Int plcID: descrittore del canale di comunicazione ottenuto in fase di inizializzazione char * label puntatore alla label che identifica la variabile da leggere void *data puntatore al buffer in cui memorizzare il valore della variabile
VALORI RITORNATI	0 se la lettura è avvenuta correttamente intero negativo in caso di errore

Plc_var_write

NOME	plc_var_write
PROTOTIPO	int plc_var_write(int plcID, char *label void *data)
DESCRIZIONE	Scrive il valore presente nel buffer data fornito dall'utente nella locazione di memoria corrispondente alla variabile identificata dalla label.
PARAMETRI	Int plcID: descrittore del canale di comunicazione ottenuto in fase di inizializzazione char * label puntatore alla label che identifica la variabile da leggere void *data puntatore al buffer in cui è contenuto il valore della variabile da scrivere in memoria.

VALORI RITORNATI	0 se la scrittura è avvenuta correttamente intero negativo in caso di errore
NOTE	Nel caso in cui si interagisca con il PLC programmato a bordo pannello la funzione plc_var_write dovrebbe essere utilizzata per scrivere unicamente valori relativi alle variabili appartenenti alla RemoteArea. La scrittura di qualunque altra variabile può determinare comportamenti non deterministici per il programma PLC.

17.10 Note per l'utilizzo PLCIB in ambiente Turbo C++

Per l'utilizzo della PLCIB in ambiente TurboC++ occorre utilizzare la libreria in formato OMF.
Il progetto Turbo C++ dovrà includere i seguenti file:
plclib.h – Header contenente l'interfaccia alla dll;
plclib_omf.lib – Import library per la dll, da importare nel progetto o da richiamare esplicitamente con la direttiva

```
#pragma comment(lib, "plclib_omf.lib")
```

plclib_omf.dll – Dll che verrà richiamata in fase di esecuzione del programma, da copiare nella cartella in cui risiede l'eseguibile generato.

17.11 CONFIGURAZIONE PER ACCESSO IN VPN

È possibile accedere alle funzionalità remote del TPAC oltre che dalla stessa LAN in cui è inserito il pannello da un qualunque computer collegato in Internet come se si trovasse sulla stessa LAN in cui è collegato il pannello. Ciò richiede l'attivazione e la configurazione del server VPN presente sul pannello. Una VPN (Virtual Private Network) è una rete privata instaurata tra pc che usano un mezzo di trasmissione pubblico (internet) avvalendosi di strumenti di cifratura e autenticazione della comunicazione per renderla sicura.
L'utilizzo della connessione VPN con il TPAC richiede, in generale, la configurazione del TPAC stesso, l'installazione e la configurazione del client VPN sul/sui pc che si vorranno utilizzare per connettersi al pannello.

Per semplificare l'utilizzo di questa funzionalità del pannello a quanti non abbiano particolari esigenze di sicurezza informatica sui dati gestiti, il TPAC dispone di una configurazione standard mentre il CD d'installazione fornisce l'eseguibile del client vpn e una serie di configurazioni valide per i client.

In configurazione standard il pannello è accessibile in VPN all'indirizzo 10.8.0.1.
Per i client che si vorranno connettere al pannello sarà necessario installare sul/sui PC l'eseguibile openvpn-2.0.9-install.exe. Inoltre una delle configurazioni fornite dovrà essere copiata nella sottocartella config del percorso d'installazione del client VPN.

Qualora si abbia l'esigenza di una configurazione ad hoc, oltre ad installare il client VPN su tutti i PC che si devono connettere al pannello, è necessaria la creazione di un'infrastruttura a chiave pubblica che prevede la creazione di una chiave pubblica e di una chiave privata per il TPAC e per tutti i pc coinvolti in modo da garantire la sicurezza delle connessioni.

17.12 Creazione infrastruttura PKI

Dal CD di installazione del TPAC copiare in una qualsiasi cartella del pc l'eseguibile openvpn-2.0.9-install.exe.

L'installazione si avvia con un doppio click sul nome dell'eseguibile: il wizard che si apre guida nell'installazione del software che di default avviene in C:\Programmi\OpenVPN. Se necessario è possibile scegliere un percorso alternativo di installazione.

NOTA: il tutorial fa riferimento al percorso di installazione standard. Nel caso si decida di installare Open VPN in una posizione differente sostituire nelle operazioni da effettuare il percorso scelto al percorso standard.
Open VPN è supportato da sistemi operativi microsoft Windows 2000 e successivi. Inoltre è necessario installarlo ed eseguirlo con un utente che abbia privilegi di amministratore della macchina. Questa restrizione imposta dai sistemi operativi Windows e non da Open VPN si può aggirare eseguendo OpenVPN come servizio, ciò ne consente l'utilizzo anche a utenti non amministratori dopo l'installazione.

Al termine dell'installazione all'interno della directory C:\Programmi\OpenVPN sarà stata creata una cartella easy-rsa che contiene i programmi necessari alla creazione dell'infrastruttura PKI.

17.13 Autorità di Certificazione

Affinché si instauri la comunicazione tra il TPAC ed un generico pc é necessario che entrambi controllino il certificato fornito dalla controparte: il pc verifica il certificato del TPAC e viceversa il TPAC verifica il certificato del pc. Perché la verifica vada buon fine é necessario che entrambi i certificati siano firmati dalla stessa autorità di certificazione che ne abbia validato i contenuti.

In Avvio/Start -> Esegui digitare la parola "command" e premere invio. Questa operazione consente di aprire una console DOS. Con il comando cd spostarsi nella console nella cartella C:\Programmi\OpenVPN\easy-rsa. Lanciare il file batch:

> init-config

Con un editor testuale aprire il file vars.bat e modificare i parametri che si trovano alla fine del file come segue ad esempio:

export KEY_COUNTRY="IT."
export KEY_PROVINCE="Italia"
export KEY_CITY="Torino"
export KEY_ORG="MiaAzienda"
export KEY_EMAIL= info@miaazienda.it

Quindi eseguiamo in sequenza i seguenti comandi:

> vars
> clean-all
> build-ca

Il comando clean-all elimina una eventuale infrastruttura a chiave pubblica precedentemente creata. L'ultimo comando premette di generare il certificato a chiave privata della Autorità di Certificazione. Avendo inserito tutti i dati richiesti nel file vars.bat occorrerà digitare durante la procedura il Common Name per l' Autorità di Certificazione, mentre per le altre domande poste dal

comando sarà sufficiente premere invio per mantenere il valore preimpostato. Il Common Name si può scegliere arbitrariamente, ad esempio “MiaAziendaCA”

Al termine della procedura la directory C:\Programmi\OpenVPN\easy-rsa\2.0\keys conterrà i file ca.crt e ca.key rispettivamente certificato e chiave privata dell’ Autorità di Certificazione.

17.14 Chiave di sessione

La connessione VPN richiede anche una chiave di sessione che viene ricreata automaticamente di default ogni ora. Per costruire tale chiave si usano i parametri generati lanciando da console nella directory C:\Programmi\OpenVPN\easy-rsa il comando

> build-dh

questo comando genera un file dh1024.pem posizionato nella cartella C:\Programmi\OpenVPN\easy-rsa\2.0\keys.

17.15 Certificato e chiave privata del TPAC

Operando sempre dalla console nella cartella C:\Programmi\OpenVPN\easy-rsa lanciamo il comando:

> build-key-server TPAC010

Anche in questo caso mantenere i dati di default che vengono proposti mentre per il Common Name scegliere ad esempio “TPAC010”.
Al termine della procedura verrà richiesto di firmare il certificato con la chiave privata dell’ Autorità di Certificazione. Rispondere di sì.

17.16 Certificato e chiave privata del client

Operando sempre dalla console nella cartella C:\Programmi\OpenVPN\easy-rsa lanciamo il comando:

> build-key mio_pc

si procede come nel paragrafo precedente dove come Common Name si può inserire “mio_pc”. Occorre anche in questo caso accettare la firma del certificato: si otterranno i file mio_pc.crt e mio_pc.key rispettivamente certificato e chiave privata del client.
Se i pc che si vogliono utilizzare per connettersi al TPAC sono più di uno basta ripetere la procedura cambiando il Common Name che li individua.

A questo punto per realizzare la VPN occorre trasferire i file generati nelle opportune directory per il pc e per il TPAC.

17.17 Configurazione VPN TPAC

Prelevare dalla cartella SAMPLE VPN presente sul cd d'installazione del TPAC il file sample_server.conf. Modificare la configurazione del server secondo le proprie esigenze di sicurezza. Non è possibile modificare il protocollo utilizzato (udp), il device (tun) e il percorso per i file contententi i certificati e la chiave pubblica. Rinominare il file sample_server.conf in tpac010.conf.
Digitare nel proprio browser l'indirizzo http://192.168.0.210/vpn.cgi per accedere alla pagina di impostazione. Nella sezione VPN Configuration eseguire con il tasto browse l'upload dei seguenti file:
ca.crt
dh1024.pem
TPAC010.key
TPAC010.crt
TPAC010.conf

17.18 Configurazione VPN del/dei client

Prelevare dalla cartella SAMPLE_VPN presente sul cd d'installazione del TPAC il file sample_client.conf. Eventualmente modificare la configurazione perché sia congruente con quella impostata per il server. Rinominarlo con il Common Name stabilito per il pc che si intende configurare, nell'esempio "mio_pc". Editare il file mio_pc.conf e sostituire nelle righe:

cert ./client.crt
key ./client.key

i nomi dei file generati come certificato e chiave privata del client

cert ./client.crt → cert ./mio_pc.crt
key ./client.key → key ./mio_pc.key

Copiare i file:
ca.crt
mio_pc.crt
mio_pc.key
mio_pc.conf

all'interno della directory C:\Programmi\OpenVPN\config
A questo punto la configurazione è completa.
Se i pc che si vogliono utilizzare per connettersi al TPAC sono più di uno basta ripetere la procedura utilizzando la coppia chiave/certificato utilizzata dal pc (Vedi paragrafo Certificato e chiave privata dei client).

17.19 Indirizzo di connessione al TPAC in VPN

Quando la connessione VPN é attiva, il TPAC é raggiungibile all'indirizzo 10.8.0.1 in configurazione standard, altrimenti è raggiungibile all'indirizzo specificato nel proprio file tpac010.conf

NOTA: Quando ci si connette al TPAC da una rete diversa dalla LAN cui il pannello é connesso occorre accertarsi che il gateway che gestisce le comunicazione da/verso internet della LAN cui é connesso il pannello permetta le comunicazioni UDP sulla porta 1194 e tale traffico venga rediretto verso il pannello.

18.0 OPERAZIONI SU INTERFACCIA USB

Il TPAC dispone di una porta USB cui è possibile connettere uno ed un solo hub USB in modo da espandere il numero di dispositivi di memoria di massa usb collegabili al pannello. Attualmente sono supportati hub con un massimo di 4 porte. Il numero e lo stato delle porte usb effettivamente utilizzabili è reso disponibile al programmatore tramite il vettore USBstatus, il vettore USBfeedback informa il programmatore della disponibilità del sottosistema usb ad evadere l'operazione richiesta. (cfr manuale TPAC). E' possibile effettuare operazioni di copia e cancellazione di file presenti su due diversi dispositivi usb connessi al pannello. Le operazioni di copia e cancellazione si intendono riferite a singoli file o a intere directory. Per le directory le operazioni sono ricorsive.

Nome	**Descrizione**	**Parametri di ingresso / Valore restituito**
Usb_copy	Copia il file specificato nella destinazione prescelta. Copia la directory specificata con tutto il suo contenuto nella destinazione prescelta.	Input: SRC : STRING Percorso relativo sul dispositivo usb del file o della directory. DST: STRING percorso relativo sul dispositivo usb in cui copiare il file o la directory specificata. SRC_INDEX: USINT indica il dispositivo USB su cui si trovano i dati da copiare (da 1 a 4). DST_INDEX: USINT indica il dispositivo USB su cui si trovano i dati da copiare (da 1 a 4).
		Output: Codice d'errore UINT * 0 - OK * 1 – Errore: risorsa occupata /operazione fallita
Usb_delete	Cancella il file specificato dal dispositivo usb richiesto. Cancella la directory specificata e tutto il suo contenuto dal dispositivo usb richiesto.	Input: NAME : STRING Percorso relativo sul dispositivo usb del file o della directory da cancellare. INDEX: USINT indica il dispositivo USB su cui si trovano i dati da cancellare (da 1 a 4).
		Output: Codice d'errore USINT * 0 - OK * 1 – Errore: risorsa occupata /operazione fallita

Usb_mkdir	Crea la directory specificata sul dispositivo usb richiesto.	Input: NAME : STRING Percorso relativo sul dispositivo usb della directory da creare. INDEX: USINT indica il dispositivo USB su cui si trovano i dati da cancellare (da 1 a 4).
		Output: Codice d'errore USINT * 0 - OK * 1 – Errore * 2 – Directory esistente

19.0 APPENDICE

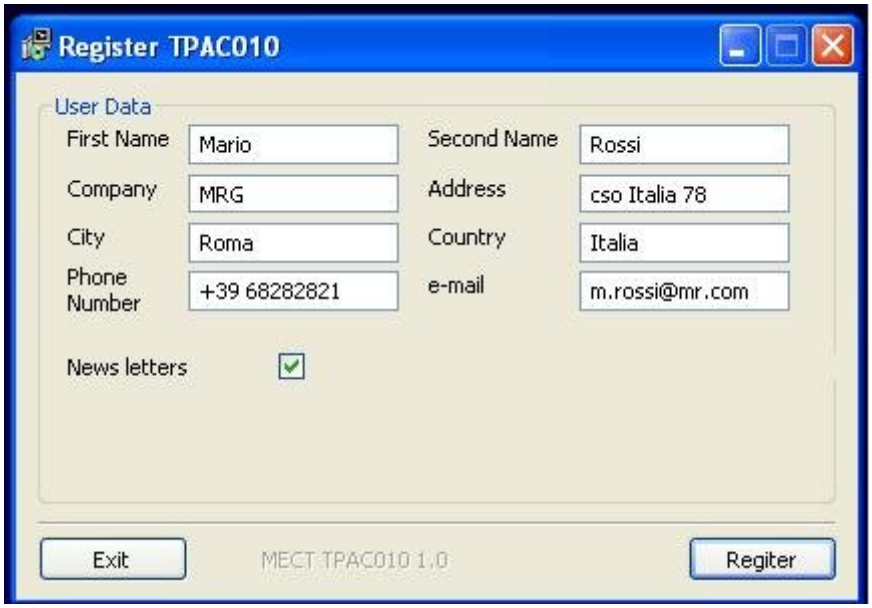
19.1 Registrazione Software

I software dell'ambiente di sviluppo
LogicLab
PageLab e
NetBuilder,

per essere utilizzati devono essere registrati dopo l'installazione.

19.2 Richiesta codici di registrazione LogicLab e PageLabe e NetBuilder

La parte finale del programma di installazione propone un form nel quale vengono richiesti alcuni dati personali dell'utente e la richiesta di iscrizione alla newsletter della MECT.



Riempito il form, premendo Register sarà creata una e-mail da inviare a MECT con la richiesta dei codici di registrazione dei software LogicLab, PageLab e NetBuilder. La mail contiene anche il numero seriale del disco sul quale sono stati installati i software.

Se l'installazione dei software avviene su un PC non collegato ad internet sarà necessario copiare la mail ed inviarla a MECT.

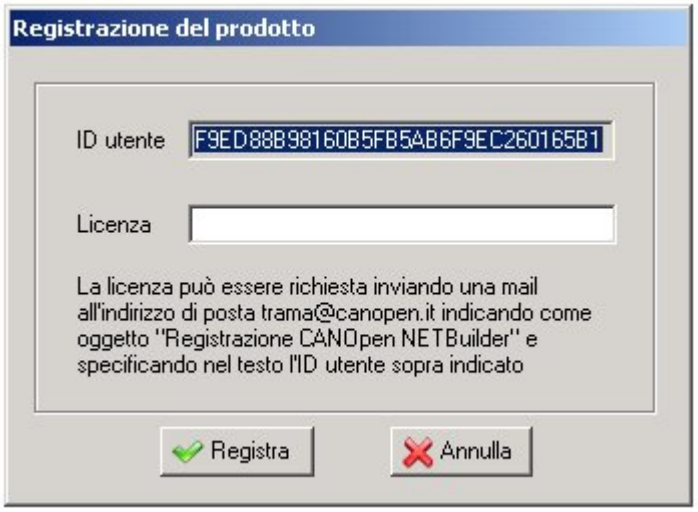
Se si vuole effettuare la registrazione successivamente, sul CD di installazione è presente il programma eseguibile register.exe, lanciando il quale sarà proposto il form per l'invio dei dati a MECT.
Importante: l'esecuzione del programma register.exe deve avvenire sul PC sul quale sono stati installati i software di sviluppo.

19.3 Registrazione LogicLab e PageLab

Ricevuti i codici di licenza da MECT, per effettuare la registrazione dei software Axel (**LogicLab** e **PageLab),** copiare il corrispondente codice di licenza utilizzando le istruzioni mostrate a video dalla finestra di registrazione.

19.4 Registrazione NetBuilder

Ricevuto il codice di licenza da MECT inserirlo nel form di registrazione.



19.5 Aggiornamento software

Sul TPAC sono inseriti diversi software che possono essere aggiornati indipendentemente l'uno dall'altro. Possiamo raggruppare i software in tre categorie:
Software di sistema (sistema operativo e librerie)
Software di servizio (impostazioni e configurazioni. Es. tastiera virtuale)
Software utente (programmi PLC HMI e configurazioni reti CAN)
L'aggiornamento dei software della categoria 1 e 2 avviene esclusivamente via chiave USB, l'aggiornamento dei software del gruppo 3 invece può avvenire sia attraverso chiave USB che via LAN (per mezzo dei software di sviluppo LogicLAb, PageLab e NetBuilder).

19.5.1 Aggiornamento del software di sistema

Questi aggiornamenti software sono forniti direttamente da MECT ai suoi clienti. Ricevuto il file di aggiornamento, denominato **sysupdate.sh**, l'utente lo deve copiare su una penna USB. Riavviando il TPAC con la penna inserita, il dispositivo si riprogramma automaticamente con gli aggiornamenti.

E’ possibile caricare i file sulla penna USB tramite dashboard: nella sezione **Settings…** di **Target** è presente il campo F**irmware upgrade**, premendo il pulsate **Browse** si visualizzerà una finestra per la selezione del file di aggiornamento sysupdate.sh; una volta selezionato il file lo si scarica sulla chiavetta premendo il tasto **Export current configuration.**

19.5.2 Aggiornamento del software di servizio

Per software di servizio si intendono tutti quei software che sono modificati dall’utente ma che non sono legati ad uno specifico progetto.
Per aggiornare i file utente come per esempio i file della tastiera virtuale, la procedura è la seguente: una volta creati su PC i file (ad es. skbd.gif e skbd.txt), si copiano sulla penna USB da inserire nel TPAC. Dopo il riavvio del TPAC con la penna USB inserita, i file saranno caricati e disponibili per l’utilizzo.

19.5.3 Aggiornamento del software utente.

Per file utente si intendono i programmi applicativi PLC, HMI e le configurazioni CANOpen. Nell’utilizzo del TPAC possiamo identificare due tipi di utenti: il programmatore e l’utilizzatore. Il primo ha la possibilità di aggiornare via rete LAN, i programmi in esecuzione sul TPAC attraverso i sistemi di sviluppo PageLab, LogicLab e NetBuilder. Il secondo non necessariamente possiede un sistema di sviluppo o le macchine sulle quali è montato il TPAC non sono raggiungibili dalla rete Ethernet; in questo caso l’aggiornamento dei programmi avviene attraverso la chiave USB. Ricevuto il file di aggiornamento, denominato **sysupdate.sh**, l’utente lo deve copiare su una penna USB. Riavviando il TPAC con la penna inserita, il dispositivo si riprogramma automaticamente con gli aggiornamenti. La creazione del file sysudate.sh avviene a seguito del comando **Download current application** inviato attraverso l’interfaccia web.