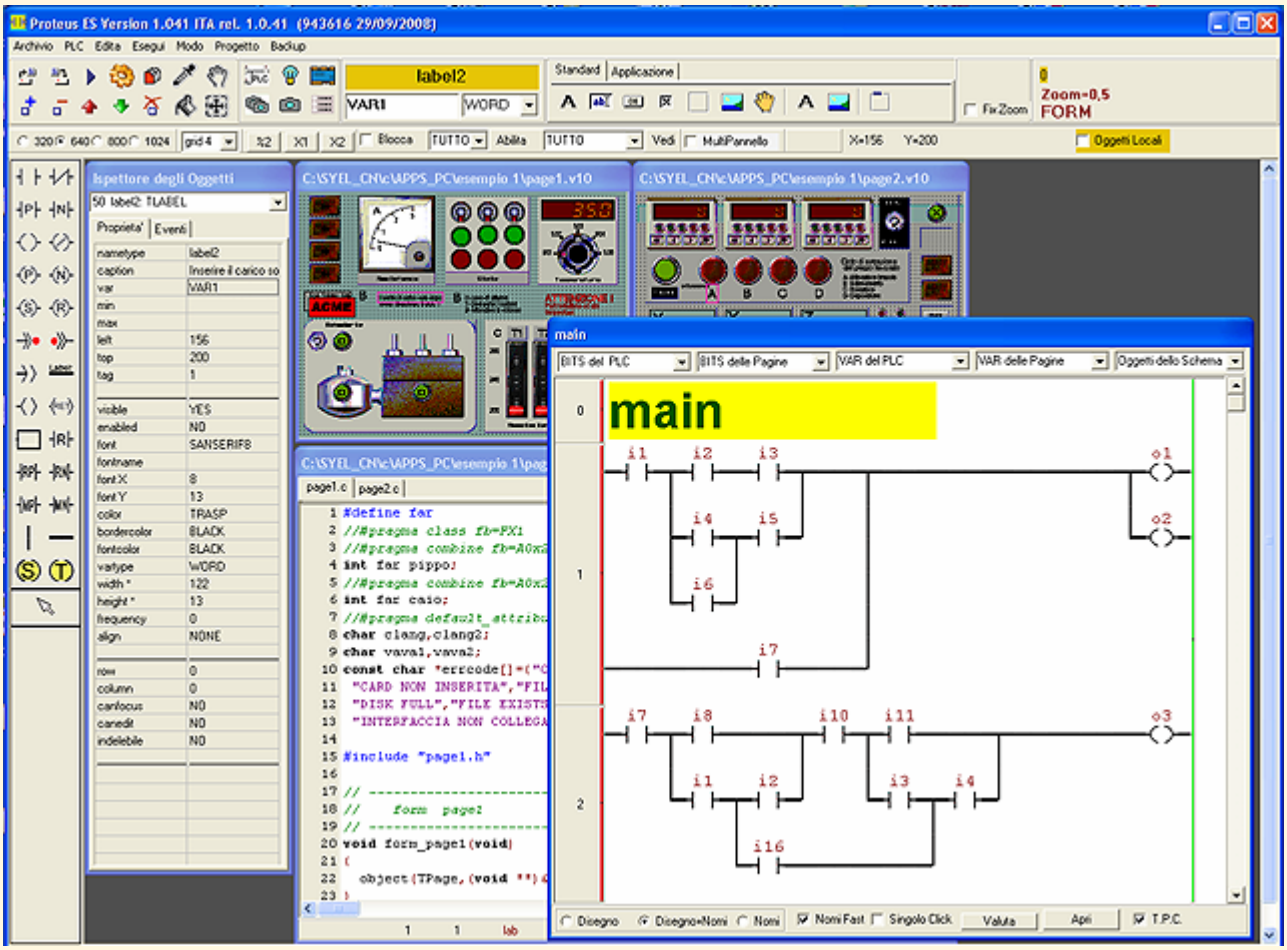
Proteus ES

Sistema di sviluppo

Proteus ES

Manuale d' uso

Quick Start



PROTEUS
Divinitá della mitologia greca, figlio di Oceano e Teti; capace di assumere qualsiasi forma.

PROTEUS è un sistema completo, composto da un tool di sviluppo software e da una serie di moduli hardware, capace di permettere un rapido sviluppo e realizzazione di un prodotto finito funzionante, anche senza una conoscenza approfondita di programmazione, partendo da un' ampia scelta di moduli precostituiti.
Il sistema Proteus presenta alcuni aspetti rivoluzionari, rispetto alla maniera classica di sviluppo di un' applicazione.

- Sviluppando una applicazione PLC con interfaccia grafica, le due componenti del progetto (il PLC e l' inerfaccia grafica) vengono sviluppate come un tutt' uno omogeneo, in un solo programma, evitando protocolli di colloquio, passaggi di parametri e altre noie, normalmente legate allo sviluppo di un tale progetto su un PLC classico.
- Un progetto creato con Proteus, senza cambiare niente, ma semplicemente ridefinendo il target sul menu del progetto, puo' essere fatto girare su configurazioni hardware completamente diverse, ad esempio: su un PLC S400 con monitor TFT, oppure con display alfanumerico, oppure su un PC con sistema operativo Windows Xp, oppure Windows CE oppure Linux, collegato con il PLC S400 remoto, oppure su PLC VK3 o V8.
- Proteus permette lo sviluppo di applicazioni per PLC senza display, con display LCD 4 righe X 20 caratteri, con monitor TFT da 3",5",10",15", con tastiera, mouse, touch-screen oppure su PC, in un ambiente di programmazione unico.

Introduzione

Con Proteus possiamo creare:

- semplici programmi di PLC scritti in **linguaggio Ladder**.
- semplici programmi di PLC scritti in **linguaggio C**.
- programmi di PLC utilizzando entrambi i linguaggi sopra detti.
- Programmi contenenti una interfaccia grafica ad oggetti ed un programma PLC.

La caratteristica principale e' l' estrema semplicita' d' uso e la rapidita' con cui e' possibile creare e testare un' applicazione.
E' possibile, in poco tempo e con alcuni click del mouse, creare a partire da zero una applicazione con interfaccia grafica, provarne il funzionamento sul PC, ed inviarla in esecuzione sull' hardware finale.
Protus e' nato per l' implementazione di **pannelli di controllo elettronici con interfaccia touch-screen**, che resta la sua funzione primaria, ma puo' essere utilizzato per la creazione di qualsiasi tipo di programma per i controlli VKx, Vx, Sx00 e derivati, con e senza monitor touch-screen.
E' possibile inoltre fare **eseguire l' applicazione** (grafica + PLC) **su un PC**, utilizzando uno o piu' PLC per l' uinterfaccia fisica con la macchina.
In fase di test si puo' simulare la macchina con un apposito programma su PC.

In un progetto sviluppato con Proteus la parte programmata in linguaggio Ladder e' separabile. Cio' rende possibile compilare l' applicazione, lasciando all' elettricista la **possibilita' di modificare sul campo e compilare separatamente la parte PLC**.

Contrariamente alla filosofia corrente dei PLC, in cui la parte PLC (residente sul PLC) e l' interfaccia grafica (di solito residente su un PC) sono due programmi ben distinti, creati separatamente e che colloquiano tra loro mediante protocollo seriale, le applicazioni create con Proteus fondono in un **unico oggetto** finale la parte PLC e la parte interfaccia grafica.
In questo modo:
- l' interfaccia grafica ha una completa visione e controllo del PLC.
- I due task girano sul medesimo hardware:
  - Entrambi sul PLC (in questo modo per la parte grafica non e' necessario un PC ma e' sufficiente un semplice monitor TFT)
  - Entrambi sul PC (in questo modo viene sfruttata al massimo la velocita' di esecuzione del programma, che e' caricata in una **estensione Real-Time** indipendente da Windows, e l' hardware s400 cieco contiene il programma standard di interfaccia input-output)

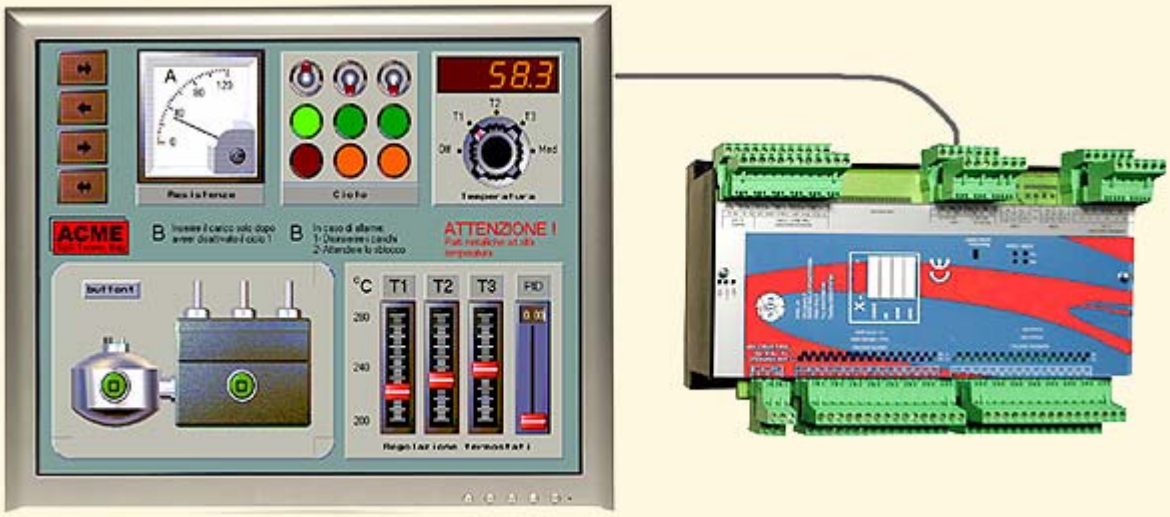
In ogni caso, l' intero programma (grafica e PLC) e' contenuto in un solo progetto, in cui i due oggetti si vedono reciprocamente, ognuno ha il controllo immediato delle variabili dell' altro, e non occorrono protocolli di colloquio, evitando le noie legate alla doppia programmazione e i ritardi e gli altri problemi relativi.

1– PROTEUS utilizzato per creare un PANNELLO ELETTRONICO

- conoscenza di programmazione necessaria: programmazione a contatti (Ladder, LD).

Proteus è stato creato primariamente per lo sviluppo di pannelli elettronici.
Un pannello elettronico consiste in un **monitor TFT** dotato di **touch-screen**, collegato con un **PLC**, che emula le funzioni dei classici componenti elettromeccanici presenti sulla macchina (interruttori, lampade spia, trimmers, amperometri ecc…), Esso espleta inoltre le funzioni di altri componenti di solito presenti sulla macchina (posizionatori, visualizzatori, termometri, PID).
Tali componenti appaiono sul monitor come disegni animati e sono manovrabili con un semplice tocco delle dita sul touch-screen.
Una parte fondamentale del progetto è la parte PLC, programmabile in **linguaggio LADDER** o in **linguaggio strutturato (C).** Tale sezione consente di collegare tra loro e con il mondo esterno i componenti sopra detti (interruttori, posizionatori ecc…), svolgendo inoltre la funzione del PLC della macchina.
La parte PLC è l'unica che ha bisogno di programmazione. Lo sviluppo della parte grafica del pannello e il piazzamento dei componenti, è effettuata con delle semplici operazioni di copia-incolla come in un programma di disegno.
I componenti introdotti sono giá completamente pre-programmati e pienamente funzionanti, e non resta che collegarli tra loro tramite il programma del PLC.

Nella libreria dei componenti inclusi nell'istallazione possiamo trovare:
-Interruttori, pulsanti, leds, lampade spia

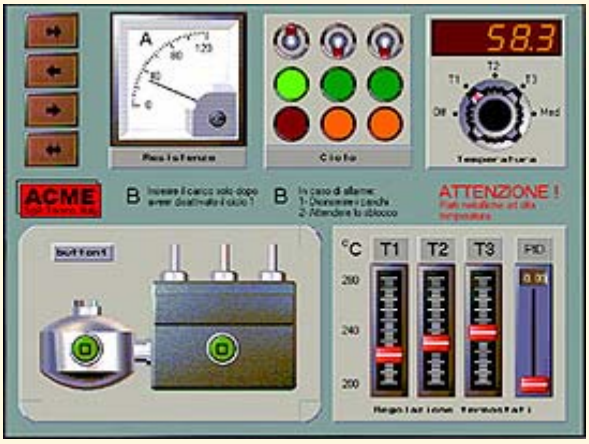


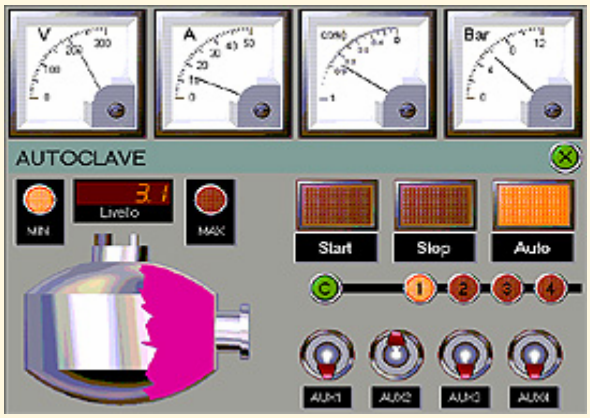
Possibile configurazione di un pannello elettronico: Monitor TFT 1024 x 768 15" + PLC S400.
In questa configurazione il PLC ha 32 ingressi e 32 uscite on-off, 8 ingressi analogici, 4 uscite digitali, 4 encoders, 4 linee seriali, interfaccia CAN-OPEN e puó controllare 4 posizionamenti tradizionali + altri su linea CAN-OPEN.

-Potenziometri rotativi e sliders
-Amperometri, Voltmetri, Wattmetri, Cosfimetri, manometri, indicatori RPM ed altri strumenti di visualizzazione, sia analogici che digitali.
-Posizionatori, interpolatori, Quasar, Mesar ed altri blocchi funzionali evoluti.

Si possono importare foto e disegni (anche animati) creati con programmi di grafica esterni.

In ogni momento dello sviluppo possiamo **testare la parte di programma sviluppata** e provare le funzioni del PLC, con un semplice click del mouse, emulando il pannello ed il PLC sul PC con cui stiamo sviluppando l' applicazione.





Esempio di pagine diverse per il pannello elettronico. Si passa da una pagina all' altra premendo un bottone sul pannello.

Una volta disposti tutti i componenti e scritto il programma PLC, con un click del mouse, possiamo generare il programma finale ed inviarlo all' hardware.

Il pannello elettronico è modificabile e riaggiornabile in ogni momento, semplicemente ricompilando il programma.Essendo tutto il progetto contenuto in un unico programma finale, l' operazione di aggiornamento è elementare, e puó essere fatta con la semplice inserzione di una penna USB, direttamente sulla macchina.

Al contrario del tradizionale pannello strumenti elettromeccanico, il pannello elettronico puó essere organizzato in **piú pagine**, e visualizzare di volta in volta solo le funzioni che interessano, puó mostrare a richiesta sinottici e pagine di allarmi, consente con estrema facilitá e a costo zero l' aggiunta di nuovi strumenti e la modifica del layout, e permette di avere a disposizione una gamma estesa di componenti (pulsanti, interruttori, visualizzatori, amperometri …) tenendo a magazzino una sola voce (il pannello).

E' possibile inoltre visualizzare le scritte sul pannello in **lingue diverse**, passando istantaneamente da una lingua all' altra.

2– Programmazione avanzata

Proteus puó essere utilizzato per creare programmi per tutta la gamma delle apparecchiature Syel serie S.

Il menu del progetto consente di specificare esattamante il target, che puo' essere un' **apparecchiatura cieca** (senza display) , oppure con **display alfanumerico** o con **monitor TFT** di vari formati con **tastiera** o **touch-screen**.

La programmazione usa il linguaggio **ANSI C**, di cui sono implementate tutte le funzioni standard, piú una serie di funzioni specifiche per la macchina, in grado di gestire la memoria Flash, i moduli di espansione seriali e CAN, le periferiche CAN-OPEN, i task, i timers, gli encoders, gli ingressi e le uscite digitali ed analogiche, le porte seriali, le memorie di massa USB ed altro.

Si possono creare **applicazioni multipagina**, utilizzanti gli oggetti prefabbricati, i componenti (label, edit, bitmap, timer ecc.) anche in forma di vettori e matrici.
E' possibile la programmazione **multitask**, la gestione personalizzata degli interrupts, dei protocolli seriali e delle periferiche CAN-OPEN.

Il progetto puó essere diviso in due parti: una, scritta in **linguaggio C**, ed una **sezione separata, programmata in linguaggio a contatti**, che puó essere **modificata ed adattata dall' istallatore** della macchina, senza toccare il resto del programma.

Quando si crea un nuovo progetto viene automaticamente generata una serie di files, inizialmente vuoti, che costituiscono il progetto:

Common.h In cui vengono dichiarate le variabili globali e le funzioni che saranno utilizzate nelle varie pagine e nel resto del programma. In questo file possiamo anche sviluppare il corpo delle funzioni che non fanno riferimento ad oggetti delle pagine (in questo punto del progetto, le pagine ed i relativi componenti non sono ancora state dichiarate).

Commonplc.h Qui possiamo sviluppare il corpo delle funzioni che usano oggetti e variabili delle pagine.

Start.c Questo codice fa parte del main del programma, e contiene il codice di inizializzazione, ovvero tutto ció che va fatto prima che il programma mandi in esecuzione la prima pagina.

Plc.c contiene il codice in linguaggio C del task del ciclo macchina. Deve contenere un codice aperto (che non si chiuda in un loop) che sará eseguito, insieme al codice generato dalla compilazione del programma scritto in

linguaggio a contatti, in un task separato, in timesharing con il codice di interfaccia uomo-macchina (il codice delle pagine grafiche).

Lang2.h contiene la definizione delle stringhe traducibili. E' generato automaticamente dal compilatore, e puó essere editato e modificato per fare apparire le scritte in lingue diverse.

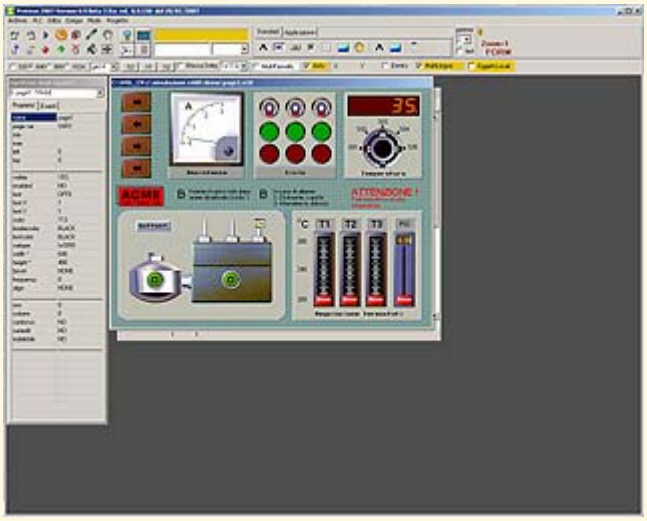
Pagexx.c E' Il codice delle pagine, e viene mostrato nella finestra di programmazione quando la pagina grafica corrispondente è selezionata. Consente di scrivere il codice degli eventi, interattivamente con la pagina grafica.

Oltre alle librerie standard, per sviluppare il programma applicativo abbiamo a disposizione due moduli di libreria (i cui headers sono x80.h e x80_obj.h) che contengono una serie di funzioni create per utilizzare le periferiche del PLC (encoders, porte seriali, can,usb ecc..) e per manipolare gli oggetti delle pagine grafiche.

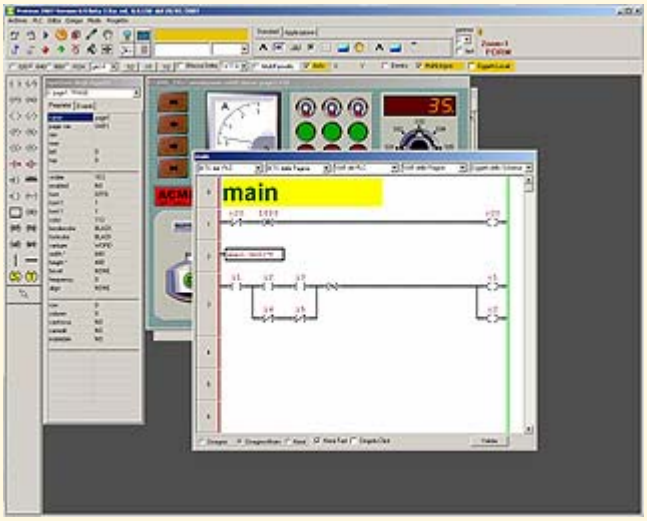
Tutte le funzioni di queste librerie sono descritte, con i relativi esempi, nel manuale di programmazione.

Proteus è attualmente lo strumento utilizzato internamente in SYEL per sviluppare le nuove applicazioni per la gamma S400 e derivati.

Fasi di sviluppo di un progetto



1– Si creano le pagine con gli oggetti da visualizzare, mediante operazioni copia-incolla dal menu degli oggetti standard a disposizione. Si possono aggiungere anche scritte, pannelli e foto e disegni personali creati con altre applicazioni.
E' possibile recuperare intere pagine o parti di esse da applicazioni sviluppate precedentemente, con il copia-incolla.



2– Si sviluppa il programma del PLC in linguaggio Ladder, collegando tra di loro ingressi e uscite del plc con gli oggetti delle pagine grafiche.
Il programmatore esperto puó fare la stessa cosa utilizzando il linguaggio C, e lasciando eventualmente una parte programmata in linguaggio Ladder, modificabile dall' istallatore, per la messa a punto finale sulla macchina.

3– Si puó testare il funzionamento dell' intero programma (pagine grafiche, PLC, ingressi e uscite, motori ecc.) sul PC da cui si sta sviluppando l' applicazione, interattivamente, senza uscire dall' ambiente di sviluppo. Questo permette di testare le funzionalitá dell' intero programma rapidamente e senza bisogno dell' harware finale.

4– Alla fine, si compila per il target finale e si invia il programma al PLC.

Esempio pratico di sviluppo di un progetto

Vediamo adesso come, in pochi minuti, con alcuni click del mouse, e' possibile istallare Proteus, creare a partire da zero una semplice applicazione, testarla sul PC ed inviarla in esecuzione all' hardware S400.

Per Istallare il programma Proteus, lanciamo il programma di istallazione **Install_proteus.exe.** A questo punto potremo trovare l' aplicativo **Proteus.exe** nella directory ***c:\syel_cn\c***.

1– Creazione del progetto

- Eseguire Proteus.Exe, e quindi nel menu principale, selezionare ***Archivio->Crea Progetto.*** Si aprira' una finestra di colloquio **SALVA NUOVO FILE CON NOME.**

- Creare una nuova directory a piacere, dove si vuole, entrare nella directory creata e, lasciando nel campo **Nome file** della finestra di colloquio il nome **page** , cliccare con il mouse sul bottone **Salva**.

- Apparira' una nuova finestra di colloquio: **DIRECTORY del PROGETTO**. Verificare che la directory sia quella che avevamo creato, lasciamo nel campo **Nome file** il nome **project.c**, clicchiamo col mouse sul bottone **Apri**.

- A questo punto si aprira' la finestra **Project Manager**. Si possono lasciare tutti i campi inalterati, o possiamo cambiare il nome del main, il titolo ecc. Per adesso non cambiamo nulla, salvo:

- Nella finestra **Software** selezionare **borland 5.5** o **DIRECTX** al posto di gcc_arm o S400, per testare l' applicazione che andiamo a costruire sul PC.
- Nella finestra **Schermo** impostiamo la risoluzione dello schermo voluta (320 x 240 se abbiamo uno schermo tipo **VK3** o **V5**, 640 x 480 se abbiamo uno schermo **V10** ecc…) **Adesso premiamo OK**.

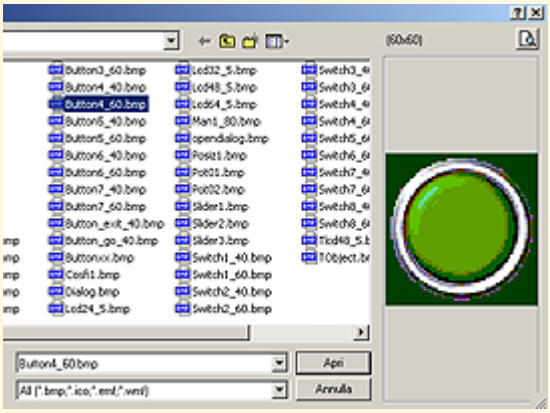
- Se abbiamo fatto correttamente quanto sopra detto, appariranno adesso sullo schermo tre finestre: **Ispettore degli Oggetti**, **directory\Page1.V10** e una **terza finestra** che per ora e' **vuota**, salvo il numero di colonna 1 in alto a sinistra.

- Cliccandovi con il mouse, portiamo in primo piano la seconda finestra (**directory\Page1.V10).** Iniziamo adesso la costruzione del nostro semplice applicativo.

2- Costruzione dell' interfaccia grafica

- Con il tasto sinistro del mouse fare un click sul bottone **oggetto** della barra standard (vedi figura a lato) , quindi cliccare sulla finestra centrale (Page1.v10) nel posto in cui vogliamo inserire il primo oggetto.

- Si aprira' un menu Apri, in cui selezioniamo l'oggetto **Button4_60.bmp**. L'oggetto apparira' sulla pagina nella posizione che avevamo scelto. Se vogliamo spostarlo, basta cliccare con il mouse sinistro sull' oggetto e, temendo premuto il bottone, spostarlo dove vogliamo. Se vogliamo un altro oggetto al posto di quello che avevamo scelto, basa un doppio click del mouse e si riaprira' il menu di scelta.



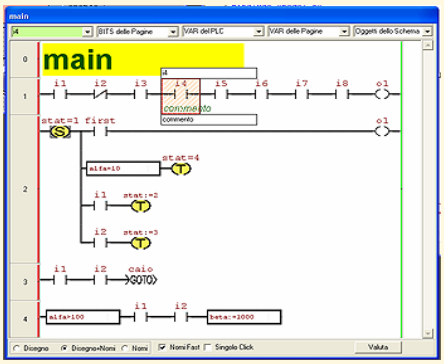
- Nello stesso modo, collocare sulla pagina un secondo oggetto, di tipo **Switch4_60.bmp**. Abbiamo adesso una pagina con un bottone, che dall' ispettore degli oggetti vediamo che si chiama **bu1** (campo var) ed una lampada spia **sw1**. (possiamo cambiare tali nomi, se vogliamo, sia dall' ispettore degli oggetti, sia cambiandone il nome nella barra in alto).

- **La costruzione dell' interfaccia grafica del nostro semplice oggetto e' terminata**. Adesso occorre solo collegare virtualmente tali oggetti tra loro ed al mondo esterno, e questo lo facciamo con la parte **PLC**.

3- L' interfaccia PLC

- Cliccare con il mouse sul bottone PLC della barra in alto. Si aprira' la finestra di programmazione in linguaggio LD, che per ora sara' vuota. Adesso andiamo ad eseguire i collegamenti che vogliamo.

- Nel nostro esempio decidiamo che premendo il bottone bu1 si attivi l' uscita 1 del PLC e contemporaneamente si accenda la lampada sw1.



(continua a pagina seguente)

- Selezionare il tipo **Contatto N.A.** dalla barra a sinistra (il primo in alto a sinistra), e quindi, nel foglio del PLC inserirlo **in riga 1** di **colonna 1** con un doppio click del mouse.
- Si apriranno due campi in corrispondenza del contatto. Quello **in alto per il nome** e uno **in basso per un eventuale commento**. Per **scegliere il nome** abbiamo **3 modi diversi**:

⇒ 1- Nella **pagina dell' interfaccia grafica** cliccare con il mouse sull' oggetto da associare (nel nostro caso il bottone verde **bu1**)

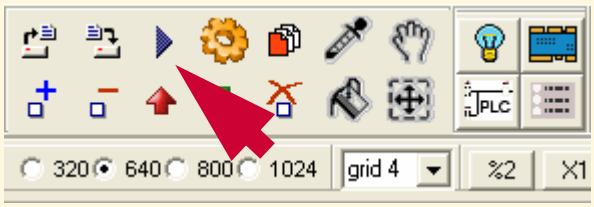
⇒ 2– Oppure: Nel menu **Bits delle Pagine** della finestra del PLC selezionare la voce **page1->bu1**.

⇒ 3– Oppure: **Scrivere direttamente** nel campo il nome dell' oggetto, nel formato **`nome_pagina->nome_oggetto`** (nel nostro caso **`page1->bu1`**).

- Selezionare il tipo Bobina N.A.dalla barra a sinistra ed inserirlo in **riga 1 colonna 2 o seguente**. Trattandosi di una bobina, essa sara' automaticamente traslata in ultima colonna. A questo punto, **selezionare il campo freccia** della barra a sinistra (l' ultimo in basso), posizionarsi sulla bobina, fare un doppio click col mouse e darle un nome, come visto sopra. Nel nostro esempio il nome sara' **sw1**.
- Adesso dobbiamo parallelare alla bobina della spia quella dell' uscita 1 del PLC, percio' selezioniamo il **campo Linea verticale** della barra a sinistra, posizioniamo tale linea prima della bobina con un doppio click, quindi selezioniamo ancora il tipo **Bobina N.A**. e inseriamola in **riga 2, ultima colonna**, dandole il **nome o1**, che potremo battere direttamente o prendere dal **menu Bits del PLC.**
- Avremo come risultato qualcosa che somiglia al disegno qui sopra.
-

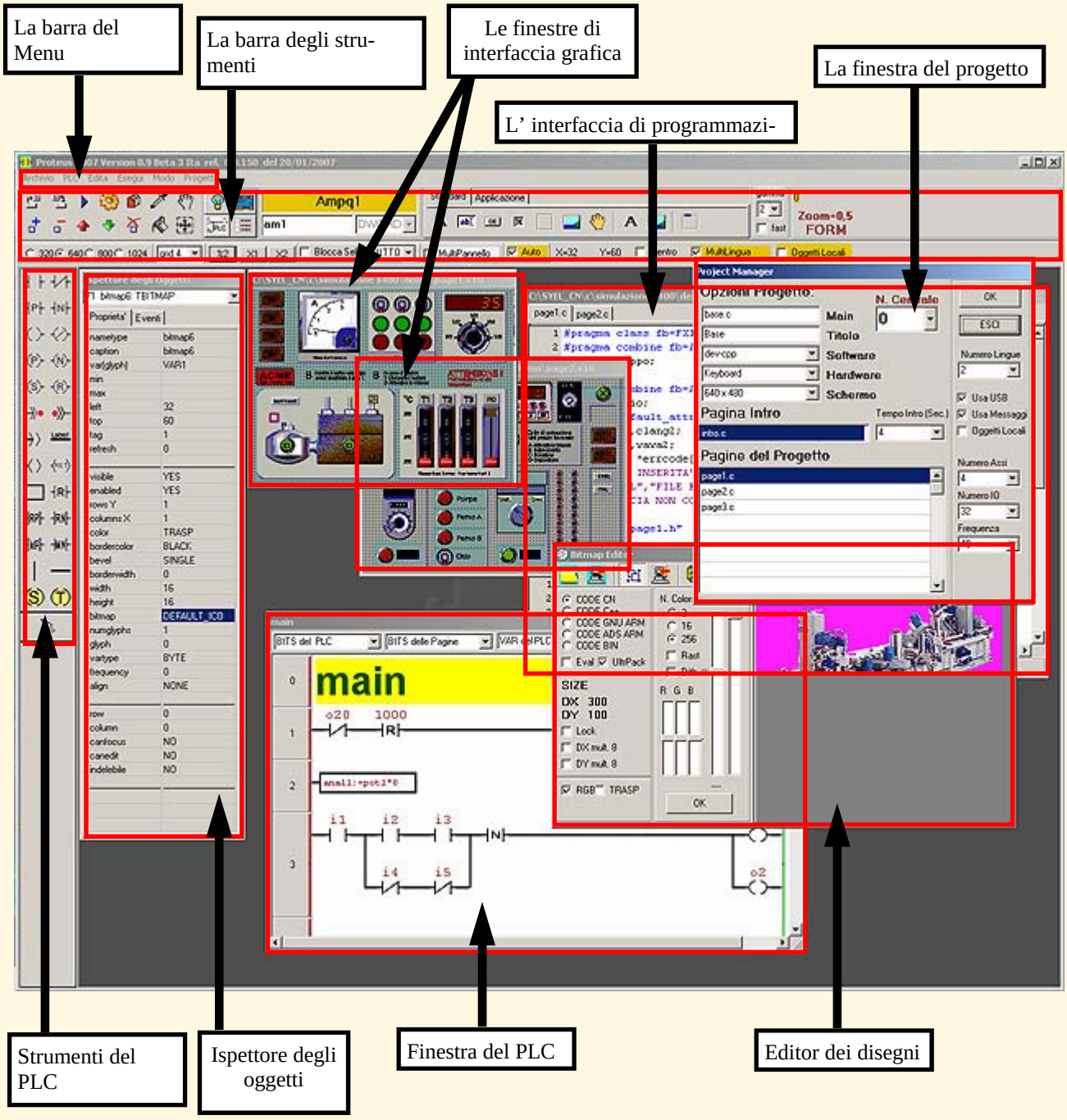
- **E' tutto. Il nostro progetto e' finito e pronto per essere testato ed inviato al PLC per essere eseguito.**

- **Per eseguire programma sul PC,** premere il tasto **Compila Tutto** sulla barra in alto. Si aprira' la finestra dell' applicazione (base.exe). Tale programma potra' essere anche lanciato in seguito una volta chiuso Proteus e si trova nella directory del progetto.



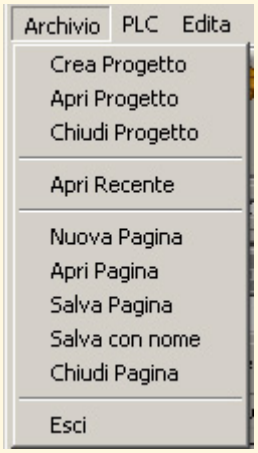
L'interfaccia

L' ambiente di sviluppo di Proteus si presenta con una finestra che racchiude una serie di elementi:



Normalmente solo alcune di queste finestre sono aperte contemporaneamente.
Vediamo ora la funzione dei componenti di ciascuno degli elementi spra indicati.

La barra del menu

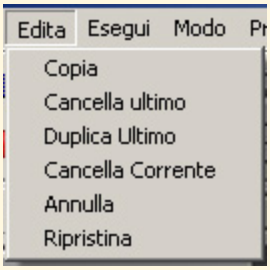


Menu Archivio:

Crea progetto, **apri progetto** e **chiudi progetto** servono rispettivamente per creare un nuovo progetto, per aprirne uno esistente e per chiuderlo.
Crea Progetto: Selezionando questo menu si apre una finestra di colloquio *Salva nuovo file con nome*. Cliccando sul bottone *Crea nuova cartella*, Per defaul verra' creata una cartella di nome *Nuova cartella*, che possiamo rinominare col nome che vogliamo. Premendo sul bottone *Apri* entriamo nella cartella creata, e scriviamo nel campo *Nome file* il nome che vogliamo dare alla *prima pagina* del progetto. Se non scriviamo niente, verrá assegnato il nome di default *Page1*. Premendo il bottone *Salva* si apre la finestra di dialogo *Directory del progetto*. Lasciamo come nome del progetto *Project.c* e premiamo il bottone *Apri*. Si aprirá la finestra del *PROJECT MANAGER*, in cui possiamo assegnare il target del progetto, ma vedremo questo piú avanti. Per ora premiamo il bottone *OK*. Si apriranno due finestre vuote: una con sfondo grigio per l' interfaccia grafica, ed una con fondo bianco per la programmazione degli eventi. Se è selezionato il *modo semplificato* si apre solo la finestra della grafica.
Apri Progetto: Selezionando questo menu si apre una finestra di colloquio *Carica file* Scegliere la directory di un progetto esistente, e cliccare sul nome di una pagina del progetto stesso.
Apri recente fa apparire un menu a discesa, contenente gli ultimi progetti aperti, permettendo di scegliere quale aprire in modo rapido.
Nuova pagina permette di aggiungere una pagina grafica al progetto. Non specificando un nome per la pagina, viene dato per default un nome progressivo Page1,Page2 ecc..
Apri pagina permette di aprire una pagina grafica del progetto. Si possono aprire contemporaneamente piú pagine.
Se siamo in *modo completo* insieme alla pagina grafica si aprirá anche la relativa pagina di programmazione degli eventi
Salva pagina salva le modifiche apportate alla pagina.
Salva con nome consente di salvare la pagina attuale con un nome diverso.
Chiudi pagina chiude la visualizzazione della pagina attuale.

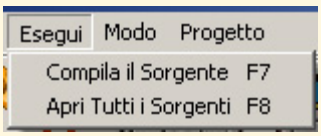
Menu PLC:

Permette di far apparire e di nascondere la finestra del PLC. La stessa funzione e' assolta dal *tasto PLC*, oppure dalla tastiera premendo *CTRL+P* e *CTRL+ALT+P* rispettivamente.



Menu Edita:

Copia: Selezionando questo menu, oppure con *CTRL+C* sulla tastiera, *copiamo in un buffer* l' oggetto grafico attualmente selezionato. La freccia del cursore sara' affiancata da un quadratino, che significa che il buffer degli oggetti non è vuoto. Cliccando con il *tasto sinistro del mouse* in una zona della finestra grafica, vi ricopiamo l' oggetto nel buffer. Cliccando invece sul *tasto destro del mouse* vuotiamo il buffer. Copiando un pannello, copieremo anche tutti gli oggetti in esso contenuti.
Cancella ultimo: Corrisponde al bottone 2 della barra degli strumenti, e cancella l' ultimo oggetto grafico creato.
Duplica ultimo: Corrisponde al bottone 1 della barra degli strumenti, e duplica l' ultimo oggetto grafico creato.
Cancella corrente: Corrisponde al bottone 5, oppure al tasto DEL. Cancella l' oggetto grafico correntemente selezionato.
Annulla: Corrisponde al bottone 3, annulla l' ultima azione. Puó essere usato recursivamente.
Ripristina: Corrisponde al bottone 4, annulla l' ultimo annulla. Puó essere usato recursivamente.



Menu Esegui:

Compila il Sorgente: usato solo in casi particolari (vedi piú avanti)
Apri tutti i sorgenti: Apre nella finestra di programmazione i sorgenti di tutte le pagine del progetto.

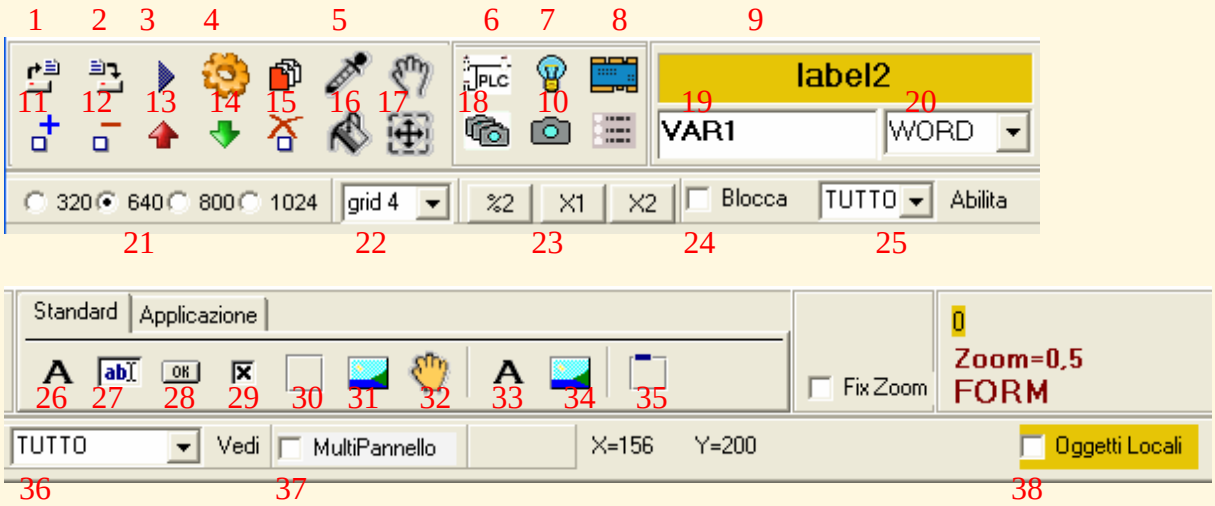
Menu Modo:

Permette di scegliere il modo di programmazione desiderato:
-**Modo Semplificato**: Adatto per creare applicazioni tipo pannello elettronico, con programmazione in linguaggio Ladder. Non mostra la finestra di programmazione degli eventi. Sulla barra degli strumenti mostra solo i componenti adatti ad essere usati su un pannello elettronico, e che non hanno bisogno di programmazione strutturata per essere usati. E' il modo piú semplice e rapido per sviluppare un' applicazione classica PLC+Pannello grafico.
-**Modo Completo**: Per il programmatore esperto, mette a disposizione tutti gli strumenti di programmazione dell' ambiente Proteus, permette di specificare metodi ed eventi, ed abilita la programmazione in linguaggio C.

Menu Progetto:

Apre la finestra del Project Manager. Equivale al bottone sottostante con l' icona della lampadina.

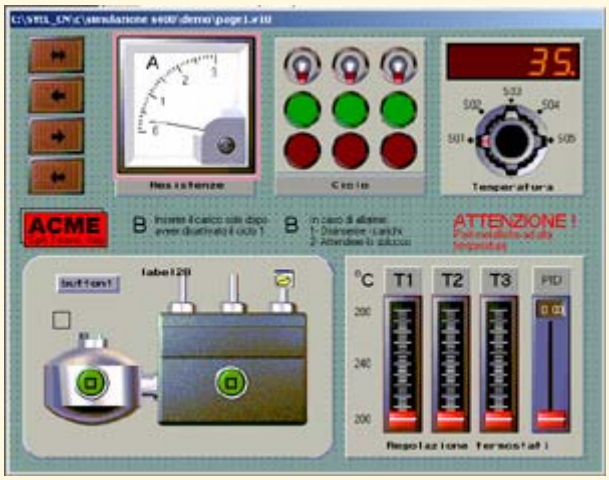
La barra degli strumenti



La barra degli strumenti ha numerosi bottoni e campi, che hanno le seguenti funzioni:

1– Apre una pagina grafica (e la relativa pagina di programma se siamo in modo completo)
2– Salva una pagina grafica (e la relativa pagina di programma se siamo in modo completo)
3– Compila il progetto, e genera l' applicativo. Se siamo in emulazione PC, apre la finestra di emulazione se la compilazione è andata a buon fine
4– compila la pagina attuale.
5– seleziona l' oggetto contenitore
6– apre la finestra di programmazione in linguaggio Ladder
7– apre la finestra del progect manager
8– apre la finestra di colloquio per inviare il programma al PLC.
9– nome dell' oggetto attualmente selezionato.
10– crea un' immagine jpeg del form nella directory del progetto
11– duplica ultimo oggetto creato
12– elimina ultimo oggetto creato
13– annulla ultima operazione fatta (Undo)
14– annulla ultima operazione di annulla (Redo)
15– cancella l' oggetto attualmente selezionato
16– mette l' oggetto attualmente selezionato dentro l' oggetto contenitore.
18– crea le immagini bmp e jpeg di tutti gli oggetti grafici del form (nelle sottocartelle bitmap e jpeg del progetto)
19– nome della variabile associata all' oggetto attualmente selezionato
20– tipo della variabile associata all' oggetto attualmente selezionato
21– dimensione della finestra grafica in pixels
22– passo della griglia per la disposizione degli oggetti
23– zoom della finestra grafica
24– blocco degli oggetti (quando questo campo e'selezionato non possiamo spostare gli oggetti con il mouse)
25– filtro per scegliere quale tipo di oggettt sono selezionabili con il mouse
26-35 bottoni per scegliere il tipo del componente da piazzare nella finestra grafica:
26– label
27– campo di edit
28– bottone
29– checkbox
30– pannello
31– bitmap (disegno o foto di tipo BMP)
32– oggetto (da selezionare nella libreria degli oggetti)
33– label statica
34– bitmap statico
35– schedario multipannello.
36– se il pannello selezionato contiene altri pannelli invisibili permette di scegliere quale visualizzare
37- permette di scegliere, in uno schedario, se selezionare l' intero schedario o la singola scheda.
37– edit automatico (per default). Viene deselezionato solo per operazioni particolari con i bottoni 6,7,16,17.
38– se selezionato, gli oggetti sono associati alla pagina che li contiene, e possiamo utilizzare lo stesso nome per oggetti diversi in pagine diverse, per contro saremo obbligati ad indicare nel nome dell' oggetto anche la pagina relativa (ad es. page1->label1). Se non selezionato gli oggetti sono globali, ma dovremo far attenzione a non chiamare con lo stesso nome oggetti in pagine diverse.

Le finestre di interfaccia grafica



In queste finestre, ognuna delle quali è associata ad una pagina dell' applicazione, e'possibile piazzare, definire e spostare componenti ed oggetti grafici che faranno parte dell' applicazione. Fra i componenti abbiamo:

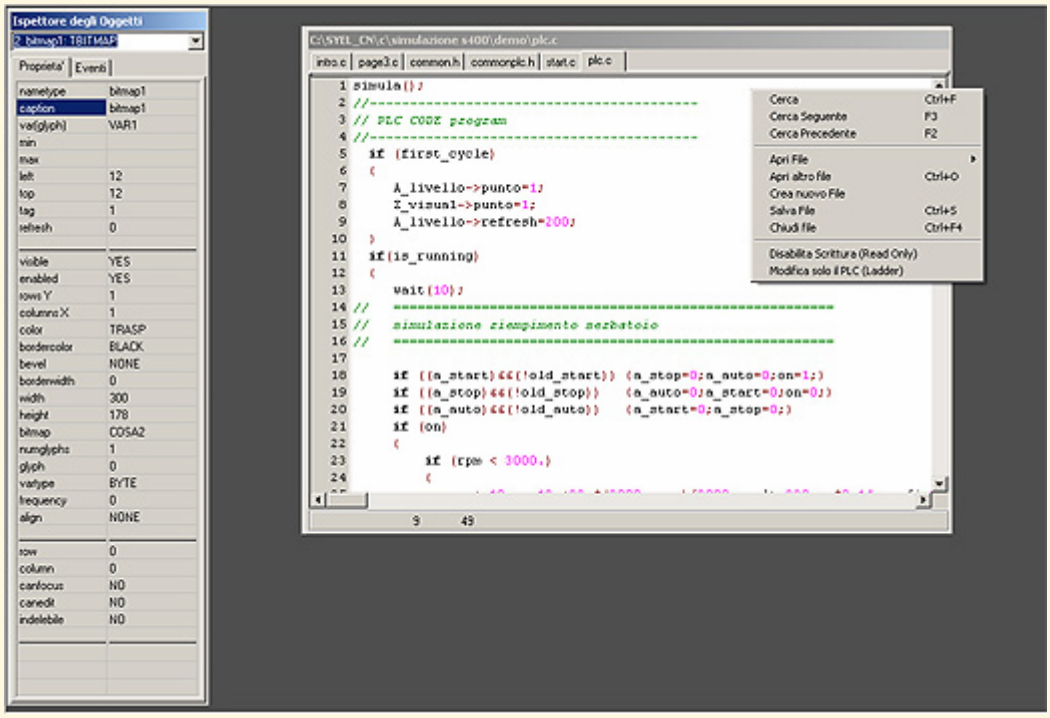
- Label
- Bitmap
- Campi di edit
- Bottoni
- Oggetti

Un Oggetto è un particolare tipo di componente che consiste in un bitmap, generalmente animato, associato ad un programma di libreria, giá pronto, che esplica tutte le funzioni relative all' oggetto stesso.

Ad esempio:
Ad un oggetto interruttore è associato un bitmap che mostra l' interruttore nella condizione in cui si trova (on o off) e un programma che si occupa di gestire il tocco del dito sul touch-screen per commutare l' interruttore, e che associa il relativo stato alla variabile associata ad esso.
Ad un oggetto amperometro è associata l' immagine di un amperometro, con il numero di divisioni e la scala adatti al fondo scala scelto, ed un programma che disegna la lancetta in movimento, leggendo il valore dall' ingresso analogico o dalla variabile che noi avremo indicato.

Per aggiungere un componente od un oggetto alla pagina (che quando viene creata è inizialmente vuota) , basta selezionare con il mouse il tipo di componente desiderato sulla barra degli strumenti, e quindi, con un altro click del mouse sulla finestra della pagina, piazzarlo dove vogliamo. In ogni momento possiamo spostare un oggetto selezionandolo con il mouse e trascinandolo, tenendo premuto il bottone sinistro del mouse stesso. Per i componenti ridimensionabili, come i pannelli, possiamo ridimensionarli tenendo premuto il bottone destro del mouse, oppure con il bottone sinistro, dopo aver portato il cursore in prossimitá dell' angolo in basso a destra del componente stesso.
Piazzando un oggetto dentro un pannello, esso sará automaticamente associato al pannello stesso, per cui, spostando il pannello, si sposterá con esso.
I pannelli possono essere recursivamente piazzati uno dentro l'altro.
I componenti possono essere associati ad una variabile in maniera automatica e biunivoca.
Ad esempio, se il campo di edit Edit1 è associato alla variabile x1, se il valore di x1 cambia, il campo di edit si aggiorna automaticamente, mentre se entriamo in input del campo (toccando con un dito sul campo stesso) e variamo il valore visualizzato, la variabile x1 assumerá il valore impostato manualmente.
Con un doppio click del mouse sul componente, si apre una finestra che consente di cambiare in modo intuitivo le proprietá del componente stesso. In alternativa, nell'ispettore degli oggetti è riportata la lista di tutte le proprietá dell' oggetto selezionato, che potremo cambiare a volontá.
Se lavoriamo in modo di programmazione completo, accanto alla finestra grafica, sará presente una finestra di programmazione, e l' ispettore degli oggetti avrá una scheda "Eventi". In tal caso potremo associare ad ogni azione relativa all' oggetto, un codice di gestione dell' evento stesso. Ció richiede la conoscenza del linguaggio C e della programmazione ad eventi (tipo Visual Basic, Delphi, Borland C++ Builder).

L'ispettore degli oggetti e l' interfaccia di programmazione



Sono le due finestre che, in **modo di programmazione completo**, permettono di *scrivere il codice del programma* e di *assegnare proprietá ed eventi* ai singoli oggetti di ogni pagina.
In **modo di programmazione semplificato**, cioè per creare semplici applicazioni di tipo PLC con interfaccia grafica, non c' é necessitá di usare queste finestre.
Si capisce da questo la ...*doppia anima* di Proteus, che puó essere usato dall' istallatore di PLC senza conoscenza di programmazione in linguaggi strutturati, ma puó anche essere utilizzato dal programmatore esperto, che puó sfruttare senza compromessi tutte le potenzialitá offerte dall' hardware e dal software.

Esaminiamo ora i campi dell' ispettore degli oggetti e le modalitá di programmazione relative.

Proprietá:

Nametype: E' il nome del componente. Se in una pagina vi sono piú componenti con il solito nome, i successivi sono dei cloni del primo e ne ereditano proprietá ed eventi, *salvo le proprietá individuali*, che sono *Caption, Var, Left, Top, Tag e Refresh*. Un componente è individuato da: **`nome_pagina->nome_componente`**.
All' interno di una pagina, la pagina stessa puó essere individuata dal puntatore **Local**.
All' interno di un evento di un oggetto, l' oggetto stesso puó essere individuato dal puntatore **me** (oppure **This**).
Ad esempio, *se siamo in pagina page1*:

```
void Event(edit1_onentry)()
{
   page2->edit3->caption="abc"; //cambia il caption di edit3 in page2
   Local->edit2->caption="123"; //cambia il caption di edit2 in page1
   me->top=100;                 //cambia la posizione y di edit1 in page1
}
```

Usando i puntatore Local e me possiamo creare un codice indipendente dalla pagina e dall' oggetto, e quindi utilizzabile senza modifiche in pagine diverse e per oggetti diversi.

Caption: E' una stringa associata al componente. Per alcuni componenti come Panel e Bitmap questo campo non è visualizzato, ma esiste comunque, essendo una proprietá individuale.
Nel campo caption possiamo scrivere un testo lungo al massimo 32 caratteri, che sará il **testo** della stringa relativa.
In alternativa possiamo battere un testo preceduto da un asterisco, in tal caso questo sará il **nome** della stringa relativa, oppure un testo seguito da un indice tra parentesi quadre, oppure un testo seguito da parentesi quadre aperta-chiusa, nel caso di vettore di componenti.

Ad esempio:
caption equivale a:

stringa 123	me->caption="stringa 123"
*stringa2	me->caption=(char *)stringa2
stringa3[x]	me->caption=(char *)stringa3[x]
stringa4[]	me->caption=(char *)stringa4[_index] (solo per vettori di componenti)

Var: E' il nome della variabile associata al componente. Per default viene assegnata la variabile VAR1, che è predefinita. Quando si assegna una variabile ad un oggetto, occorre che questa variabile sia definita nell' header del progetto (common.h) ed occorre settare la proprietá vartype del componente conformemente al tipo della variabile.
Il nome della variabile puo' essere l' elemento di un vettore, ma non un' espressione (deve essere qualcosa che puó stare a sinistra dell' uguale in una assegnazione).
Per vettori e matrici di componenti si possono usare variabili seguite da parentesi quadre aperta-chiusa per gli indici automatici.
Ad esempio, *sono nomi validi*:
alfa alfa[2] beta[num1*3+5] varx[] (solo per vettori di componenti) vary[][] (solo per matrici di componenti)
Ma *non sono nomi validi*:
Alfa+1 alfa+beta

Left Top Height Width: Indicano la posizione del componente all' interno della pagina, oppure del pannello se sono contenute in un pannello, e le dimensioni del componente. Normalmente si modificano spostando e ridimensionando l' oggetto con il mouse, ma per piccoli allineamenti fuori griglia o per altri motivi possono anche essere modificati manualmente impostando un nuovo valore.

Tag: E' un campo numerico generico, che puó essere usato liberamente dal programmatore. Nel caso di oggetti cloni puó essere usato per associarvi un indice che distingue un oggetto dall' altro. Nel caso di *bottoni* e di *checkbox*, se viene *posta ad 1 la proprietá refresh*, il tag identifica il bottone (o il checkbox) nell' insieme di bottoni che hanno la stessa variabile (ovvero dello stesso gruppo). (premendo il bottone la variabile assume il valore del tag, cambiando il valore della variabile, il bottone il cui tag corrisponde a tale valore, risulterá premuto).
Nel caso di un pannello invisibile il Tag e' l' indice di visualizzazione (1,2..). In tal modo sara' possibile mediante il menu a tendina (punto 36 a pag 15) scegliere quale visualizzare nella finestra di Edit. Questo e' particolarmente utile in caso di finestre parzialmente o completamente sovrapposte).

Refresh: Normalmente a zero, tale variabile viene posta ad 1 per bottoni e checkbox mutuamente esclusivi (vedi Tag), e per campi di edit se vogliamo che il campo venga ridisegnato periodicamente anche se il valore della variabile relativa non cambia.

Frequency: Indica la *frequenza minima* (in millisecondi) *di aggiornamento del campo*. Ad esempio, se per un campo di edit mettiamo la proprietá frequency=50 il campo verrá aggiornato ogni 50 millisecondi, anche se il valore della variabile cambia piú velocemente. Se posto a zero, il campo verrá aggiornato ogni volta che cambia il valore della sua variabile.

Align: Proprietá valida solo per componenti di tipo panel, indica l' allineamento del pannello rispetto al suo contenitore (la pagina o il pannello in cui è contenuto). Ad esempio possiamo mettere la proprietá align=TOP per la barra del titolo, ed in tal modo essa si posizionerá sempre in alto, con la larghezza uguale a quella del pannello che la contiene.

Canfocus, Canedit: Un campo di edit è editabile quando entrambe queste proprietá sono ad 1. Un pannello con proprietá canfocus ad 1 puó contenere campi di edit selezionabili con il tabualatore da tastiera.

Visible: Un oggetto è visibile se ha questa proprietá ad 1. Un oggetto che contiene altri oggetti (ad esempio un pannello) se posto invisibile (visible=0) rende invisibili anche tutti gli oggetti in esso contenuti.

Enabled: Un oggetto con questa proprietá ad 1 è sensibile al tocco del dito sul touch-screen.

Rows, Columns: Per pannelli, label e campi di edit, serve per *creare vettori o matrici di componenti*. Con columns > 1 si crea un vettore orizzontale, con rows > 1 un vettore verticale, se entrambi i campi sono > 1 si crea una matrice di componenti. I campi *var* e *caption* relativi possono essere indicizzati con l' indice automatico (parentesi quadre aperta-chiusa)

Color: E' il colore dello sfondo del componente. Con *un doppio click* del mouse sul *campo color* nell' ispettore degli oggetti, si apre una finestra di dialogo per la scelta del colore.
Fontcolor: E' il colore del testo per componenti Edit, Label, Button. Come nel caso della proprietá color, con *un*

doppio click del mouse sul *campo color* nell' ispettore degli oggetti, si apre una finestra di dialogo per la scelta del colore

Bordercolor: E' il *colore del bordo* per *pannelli, label e bitmap*, mentre per campi di *Edit* e *Checkbox* e' il colore della scritta (caption) a destra del campo stesso.

Bevel: E' lo stile del bordo, e puó essere selezionato con il menu a discesa associato nell' ispettore degli oggetti.

Borderwidth: E' lo spessore del bordo in pixel. Deve essere maggiore o uguale a zero.

Bitmap: Per componenti di tipo Bitmap e StaticBitmap, è il nome del bitmap da disegnare. Con un doppio click sul campo bitmap nell' ispettore degli oggetti o sul bitmap stesso nella finestra grafica, si apre un menu per la ricerca del file che contiene il disegno (*il file deve essere di tipo .BMP*). Scelto il file, si apre la finestra di Editor dei disegni, con la quale si puó ritoccare e ritagliare la parte di immagine che ci serve (vedi capitolo Editor dei disegni). In alternativa, se il disegno appare giá nel progetto ed è giá stato importato, possiamo scrivere nel campo, direttamente il nome del bitmap stesso. L' immagine puó contenere uno o *piú disegni, tutti della stessa dimensione, affiancati in senso orizzontale*. Nel secondo caso, con la proprieta **Numglyphs** indicheremo il numero di disegni di cui il bitmap è composto, mentre la proprietá **glyph** indica quale di questi disegni deve essere visualizzato.
Quando il programma va in esecuzione, *il numero del disegno* che viene visualizzato è quello indicato dalla *variabile associata* al componente bitmap stesso var(glyph).

Vartype: E' il tipo della variabile associata al componente. Deve corrispondere al tipo effettivo della variabile: ad esempio se la proprietá var (var(glyph) nel caso di bitmap) è alfa1 di tipo short int, il relativo campo Vartype deve essere WORD.

La corrispondenza è:

char, unsigned char	BYTE
short int, unsigned short int	WORD
Long int, unsigned long int	DWORD
float, double	REAL
char *, char []	STRING

Altre proprietá: **min, max, row, column, indelebile, mobile**: Si usano in casi particolari, che per ora omettiamo di considerare.

Metodi (o eventi):

La seconda scheda dell' ispettore degli oggetti è dedicata alla gestione degli eventi.
Ogni oggetto della pagina, compresa la pagina stessa, che è l'oggetto numero zero, ha associato un certo numero di eventi, che, se associati ad una funzione, la chiamano qualora si verifichino.
Per creare una funzione per un evento, basta scriverne il nome nel campo relativo, oppure, con un doppio click sul campo, lasciare che Proteus generi un nome di default (scelta consigliata).
La funzione sará automaticamente creata nella finestra di programmazione. Con un doppio click sul nome nell' ispettore degli oggetti, il cursore della finestra di programmazione si porterá sulla funzione stessa.
Piú eventi, anche di componenti diversi, possono essere associati alla stessa funzione.

All' interno del codice di una funzione, possiamo usare certe variabili predefinite:

`(char)_byte`	é la variabile associata se di tipo BYTE
`(short int)_word`	é la variabile associata se di tipo WORD
`(long int)_dword`	é la variabile associata se di tipo DWORD
`(_double)_real`	é la variabile associata se di tipo REAL
`(char *)_string`	é la variabile associata se di tipo STRING

Queste variabili possono essere usate in lettura e scrittura: ad es:

```
if(_byte > 2) _byte++;
```

`(short int)x_x`	é la coordinata x assoluta dell'oggetto (primo punto a sinistra)
`(short int)y_y`	é la coordinata y assoluta dell'oggetto (primo punto in alto)
`(OBJ *)me`	é il puntatore all' oggetto stesso
`(TYPE *)Local`	é il puntatore alla pagina corrente

`(unsigned char) page_end;`	Se posta ad 1 si esce dalla pagina corrente
`(unsigned char) page_init;`	Vale 1 durante l' inizializzazione della pagina

`(char) __autorefresh;`	Vale 1 se l' oggetto è in autorefresh
`(char) __clone;`	Vale 1 se l' oggetto è un clone
`(short int) mouse_x;`	Ascissa del mouse
`(short int) mouse_y;`	Ordinata del mouse
`(char) _numglyphs;`	Numero di disegni da cui è composto il bitmap
`(char) _keyup;`	Il touch screen è stato appena rilasciato
`(char) _keydown;`	Il touch screen è stato appena premuto
`(char) _keypress;`	Il touch-screen è premuto
`(char) _abilita_azione_draw;`	il disegno dell' oggetto è stato modificato (valido in in on_exit)
`(char) _abilita_azione_repaint;`	l'oggetto è stato completamente ridisegnato (valido in on_exit)
`(char) _abilita_azione_click;`	e' stata eseguita l'azione on_click (valido in on_exit)
`(char) _riga;`	riga attuale del campo di edit
`(char) _colonna;`	colonna attuale del campo di edit
`(char) _changed;`	l'oggetto è cambiato

Eventi:

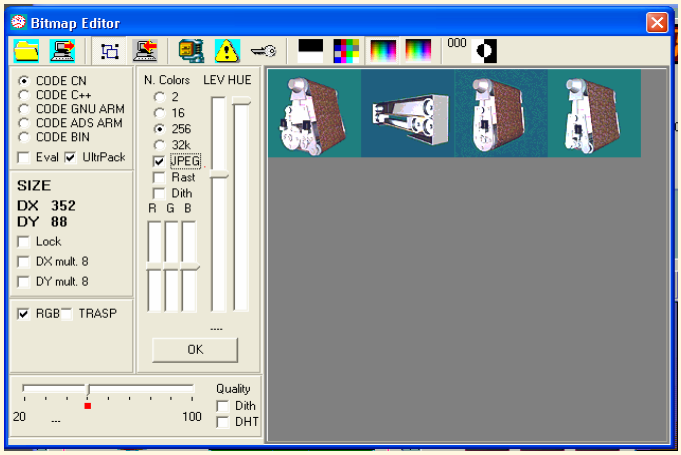
On_first	Questo evento viene chiamato, se definito, quando si apre una nuova pagina
On_create	Chiamato quando si ridisegna la pagina
On_entry	Chiamato periodicamente, sempre, anche se l'oggetto non è abilitato e non visibile
On_draw	Chiamato quando l'oggetto deve essere ridisegnato
On_click	Chiamato quando si clicca sull' oggetto
On_mousedown	Chiamato quando il mouse (o il touch-screen) viene premuto
On_mousepress	Chiamato periodicamente finché il mouse (o il touch-screen) resta premuto
On_mouseup	Chiamato quando il mouse (o il touch-screen) viene rilasciato
On_exit	Chiamato periodicamente quando si esce dall' oggetto

Anche se nessuna di queste funzioni è definita, l' oggetto funziona regolarmente (si autoaggiorna quando cambia la variabile, un campo di edit visualizza e consente di modifivare la propria variabile ecc..). Gli eventi sopra elencati, qualora definiti dal programmatore, consentono di *fare cose in piú* rispetto alla normale gestione della pagina.
Nel caso di programmazione semplificata, in linguaggio Ladder, non occorre definire alcun evento.

Il programmatore puó all' interno di un evento o altrove, modificare le proprietá di qualunque oggetto di qualsiasi pagina. Le proprietá modificabili sono:

`Pagina->oggetto->Left`	
`Pagina->oggetto->Top`	
`Pagina->oggetto->Width`	
`Pagina->oggetto->Height`	
`Pagina->oggetto->Caption`	
`Pagina->oggetto->Tag`	
`Pagina->oggetto->Color`	
`Pagina->oggetto->BorderColor`	
`Pagina->oggetto->FontColor`	
`Pagina->oggetto->BorderStyle`	
`Pagina->oggetto->Glyph`	
`Pagina->oggetto->DecimalPointPosition`	
`Pagina->oggetto->Digits`	
`*Pagina->oggetto->Visible`	(usare come puntatore)
`*Pagina->oggetto->Enabled`	(usare come puntatore)

L' editor dei disegni



Per scegliere l' immagine da visualizzare in un campo di tipo Bitmap, si usa l' editor dei disegni, che viene richiamato con un doppio click del mouse su un oggetto di tipo bitmap sulla finestra grafica, o con un doppio click sulla relativa proprietá nell' ispettore degli oggetti.
Il file di tipo bitmap (BMP) viene visualizzato nella finestra a destra, e puó essere ritagliato, spostando l' angolo in alto a destra tenendo premuto il tasto sinistro del mouse, e l' angolo in basso a sinistra premendo il tasto destro del mouse.
Solo la parte di immagine che si vede nel riquadro verrá visualizzata nell' applicazione.
Possiamo utilizzare i cursori LEV, HUE, R, G e B per cambiare la tinta del disegno.
Selezionando il campo TRASP possiamo rendere trasparente un colore, che sceglieremo cliccando sull' immagine con il mouse. Il relativo cursore orizzontale permette di regolare la similitudine di colore per la trasparenza. Se l' immagine è composta da piú bitmap affiancati (nell' esempio della figura sono due bitmap) occorre fare attenzione a ritagliare una porzione di immagine che abbia la larghezza DX multipla del numero di immagini (ad es. se sono 2 disegni DX deve essere multiplo di 2, se sono 3 disegni, deve essere multiplo di 3 ecc..)
Premendo il tasto OK il disegno viene convertito in codice e la finestra si chiude.
La profondita' di colore massima per l' immagine editata e' selezionata con l' opzione N. Colors, per cui, abilitando N. Colors=32K e Jpeg vengono campionate tutte le possibili modalita' di colore.
Nella finestra del progetto sceglieremo la profondita' di colore massima con cui compilare l' applicazione.
Per ogni immagine la profondita' di colore effettiva sara' il minimo tra la profondita' di colore del progetto e quella della singola immagine.
I campi CODE CN, CODE C++ ecc. sono ininfluenti.
Per la massima efficienza del codice, si consiglia di utilizzare N. Colors=256, di non utilizzare il campo Dith, e comunque selezionare sempre la modalitá UltraPack.
L' opzione Jpeg consente di generare programmi piu' compatti, ma il prezzo e' una maggior attesa all' avvio dell' applicazione, dovuta al tempo di decodifica delle immagini Jpeg.

Il bitmap editor è la sola maniera di importare dei disegni nell' applicazione, in quanto il codice C non è in grado di leggere direttamente dei BMP o dei JPEG, i quali debbono essere trasformati in codice binario compresso.

La finestra del progetto

La finestra del progetto, che si puó aprire in qualsiasi momento per modificare le opzioni, consente di specificare il target.

Main è il nome dell' applicativo finale, ad esempio se main=base.c, il programma finale si chiamerá base.com per il PLC o base.exe per il PC.

Titolo è il titolo dell' applicativo (usato solo nel caso di programma per il PC).

Software puo' essere gnu_arm (per Vk5 e V8), arm (per applicazioni di sistema su VK5 e V8), S400 (il modulo di espansione PLC), oppure dev-cpp (programma in emulazione su PC) oppure ancora borland 5.5 (per PC, se vogliamo utilizzare questo compilatore, liberamente scaricabile dal sito http://trial.borland.com/survey.aspx?sid=33)., oppure ancore DirectX (il Programma gira su PC utilizzando Directdraw per la visualizzazione)

Hardware specifica l' interfaccia uomo-macchina, che puó essere il touch-screen, la tastiera, entrambi, oppure nessuna interfaccia in caso di PLC senza display (cieco).

Schermo indica la risoluzione dello schermo, che puó essere 320 x 200, 640 x 480, 800 x 600, 1024 x 768, oppure LCD (4 righe x 20 caratteri), oppure NO nel caso di PLC cieco.

Pagina Intro è una eventuale pagina di introduzione da visualizzare all' accensione della macchina per il tempo indicato nel campo **Tempo Intro**.

Pagine del progetto è la lista ordinata delle pagine che fanno parte del progetto. Tale lista viene generata ed aggiornata automaticamente ogni volta che aggiungiamo una pagina al progetto, ma puó anche essere modificata manualmente. Cliccando su un campo, si apre il menu delle pagine del progetto. Inoltre le pagine possono essere inserite ed eliminate a mano, scrivendi direttamente nei campi della lista.

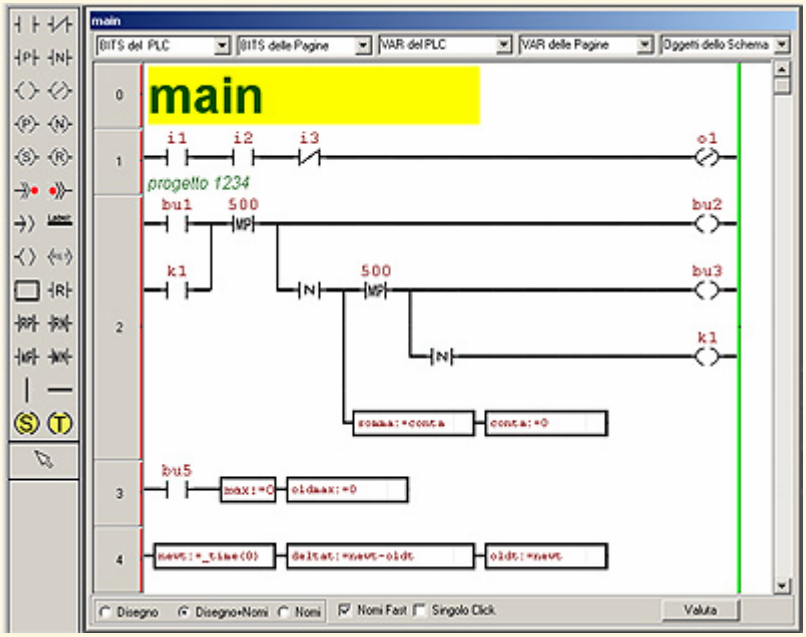
Numero Lingue indica il numero delle lingue in cui intendiamo visualizzare le pagine del progetto.
Per ogni stringa (caption ecc..) verrá generato automaticamente una serie di stringhe nel file del vocabolario di traduzione, corrispondente al numero di lingue scelto. Per questo è bene decidere dall' inizio, onde evitare di dover inserire manualmente le stringhe nel vocabolario, quante lingue vogliamo utilizzare.

Ad esempio, se per il campo label1 utilizziamo la caption="stringa di prova", troveremo nel vocabolario, ovvero nel file *LANG2.H* nella directory del progetto, la riga seguente:

```
const char *L_stringa_di_prova[]={"stringa di prova","stringa di prova","stringa di prova"};
```

La stringa è ripetuta tante volte quante specificate nel campo numero lingue. Per default viene ripetuta n volte la stringa originale. Sta a noi editare manualmente il file LANG2.H e tradurre le varie ricorrenze della stringa nelle lingue volute (non occorre che le stringhe abbiano lo stesso numero di caratteri)
Se la variabile Lang vale 1 verrá visualizzata la prima stringa, se vale 2 la seconda ecc.
Se cambiamo il valore della variabile Lang tutte le stringhe visualizzate cambiano istantaneamente.

Finestra e strumenti del PLC



Aprendo la finestra del PLC appare anche la barra degli strumenti relativi. Con questa barra possiamo selezionare il componente da inserire nello schema.
Quando il cursore e' dentro la finestra del PLC, un quadrato grigio indica la posizione di edit attuale. Se non è selezionato il campo SINGOLO CLICK possiamo piazzare il componente con un doppio click del mouse, altrimenti basta un singolo click (sconsigliato).
Nel quadrato attorno al cursore appare, in un angolo, il simbolo del componente scelto. Se quest' ultimo è barrato in rosso significa che non puó essere piazzato nella posizione attuale.
Per cancellare un componente, posizionare il cursore su di esso, e quindi premere il tasto DEL. Se il componente è una linea verticale, per cancellarla occorre selezionare il componente linea verticale, altrimenti non riusciremo a piazzare il cursore sulla linea stessa, visto che si trova tra due celle adiacenti.
Se siamo *in modalitá DISEGNO*, possiamo piazzare velocemente i componenti, senza assegnare i nomi.
Se siamo *in modo DISEGNO+NOMI*, una volta piazzato un componente, appare la finestra di edit per assegnare il nome (in alto) ed eventualmente un commento (in basso). Se non vogliamo assegnare subito il nome, possiamo uscire dall' edit con un click del mouse. Potremo assegnarlo quando vogliamo, selezionando come componente la freccia, e cliccando sulla casella dello schema a cui vogliamo assegnare o cambiare il nome. *In modo NOMI* la barra degli strumenti scompare e possiamo assegnare o cambiare i nomo ed i commenti ai componenti.
Il nome puó essere battuto direttamente da tastiera, oppure cercato nei menu a discesa in alto nella finestra, oppure puó essere assegnato cliccando nella finestra grafica sul componente relativo.
Per inserire una nuova riga vuota tra due preesistenti occorre un doppio click a sinistra sulla barra grigia con i numeri delle celle. Per cancellare una riga, selezionarla tutta da sinistra a destra (dalla barra rossa a quella verde) e quindi premere il tasto DEL. Nella pagina del PLC possiamo selezionare piú celle, tenendo premuto il bottone sinistro del mouse, e possiamo effettuare operazioni di taglia-incolla anche di intere aree, con i tasti CONTROL+C e CONTROL+V.
Con il bottone VALUTA possiamo valutare la correttezza dello schema, Se viene individuato un errore (una cella senza nome, oppure una connessione mancante) viene segnalato l' errore e il cursore, tratteggiato in rosso, si porta sulla cella dove c' è l' errore. Nel caso in cui l' errore sia in una SUB, la pagina visualizzata sará quella della SUB in cui è stato riscontrato l'errore. La valutazione si ferma al *primo errore* trovato. Per continuare, occorre correggere l' errore, chiudere la valutazione premendo ancora il bottone VALUTA, e quindi premere ancora il bottone VALUTA per una nuova scansione dei files del progetto. La valutazione parte sempre dall' inizio del programma principale, indipendentemente dalla pagina visualizzata. Per un corretto editing, è meglio richiudere la finestra di valutazione degli errori, premendo ancora sul bottone VALUTA.
Da notare che un singolo componente (un filo orizzontale od un contatto) in prima colonna, non connesso ad altro, non genera errore. Questo per consentire certe funzioni speciali di debug, che vedremo dettagliatamente nel manuale del debug.
Per scorrere verticalmente lo schema, utilizzare la barra di scorrimento a destra. La barra a sinistra mostra il numero della cella. Una cella puó essere composta da 1 a 8 linee. Una linea puó contenere fino a 8 contatti piú la bobina. Per linee piú lunghe utilizzare il simbolo di continuazione, che permette di continuare la linea su un' altra cella.
I simboli utilizzabili sono quelli standard della programmazione a contatti (contatti, bobine, monostabili,

derivatori, ritardi, logic blocks, linee verticali ed orizzontali, goto, sub, labels, return) piú i simboli di STATO e TRANSIZIONE che permettono di implementare macchine a stato. Per una spiegazione dettagliata dell' uso dei componenti del PLC si rimanda al manuale di programmazione.

Alcuni tasti hanno funzioni particolari:

DEL o CTRL+Z permette di cancellare la casella di un oggetto, nel caso lo volessimo sostituirlo con un altro (si possono piazzare oggetti solo in caselle vuote o contenenti un filo orizzontale)
CTRL+X come DEL, ma il componente attuale diviene quello cancellato (equivale a battere un DEL e dopo andare a scegliere nella barra dei componenti del PLC quello appena cancellato)
CTRL+C il componente attuale diviene quello che si trova sotto il cursore.
CTRL+V inserisce nella casella del cursore il componente selezionato (equivale ad un doppio click).
SPACE trasla a destra la riga attuale a partire dal cursore.
INSERT trasla in basso le righe della cella attuale a partire dal cursore.
BACKSPACE annulla ultima operazione (undo).
F3 cerca una casella che abbia lo stesso nome di quella sotto il cursore. Se il cursore è su una SUB apre lo schema che contiene la SUB. Se il cursore è su un return, oppure se è su una casella bianca e siamo in una SUB, ritorna allo schema precedente.
La maniera piú semplice per creare un nuovo schema (cioè una nuova pagina di PLC, ovvero una SUB), consiste nell' aggiungere allo schema attuale una funzione SUB, dargli un nome e quindi battere il tasto F3. Il nuovo schema si aprirá automaticamente.
Le SUB possono essere annidate recursivamente una dentro l'altra, fino ad un livello di 100.
Ricordiamo che una funzione SUB deve essere l' ultima della propria linea, anche se non è in colonna 8.
Per passare da uno schema all' altro possiamo anche usare *il tasto Apri*.
I tasti CTRL+X, CTRL+z, CTRL+C e DEL possono essere usati per cancellare e spostare intere aree dello schema, selezionate tenendo premuto il bottone sinistro del mouse.

Come trasferire il programma compilato sul PLC:

Nel caso di **PLC S400** si puó utilizzare una porta seriale del PC (RS-232, eventualmente su espansione USB, meglio se veloce. A tale scopo é possible aprire una finestra di colloquio seriale cliccando sull' icona "invia programma all' hardware" come da figura.

Nel caso di **PLC ARM** vi sono anche altri metodi:

- Mediante chiave USB, con programma autoexec.bat che carica automaticamente l' applicazione sulla SD del PLC (unico metodo in caso di PLC cieco)
- Mediante chiavetta USB e copia-incolla da explorer del PLC (caso di PLC V8 con touch screen e shell di Windows)
- Mediante connessione Ethernet, via TFTP
  In tal caso é possible utilizzare una qualsiasi utility TFTP per Windows oppure battere direttamente da finestra di Dos o con Start->Esegui …

`TFTP -I 192.168.1.201 PUT filename c:\filename`	per copiare su SD del PLC
`TFTP -I 192.168.1.201 PUT filename RUN`	per mandare un esecuzione
`TFTP -I 192.168.1.201 PUT filename INSTALL`	per mandare un autoesecuzione

Quando il PLC é in attesa di programma, supposto che l' indirizzo IP del PLC sia 192.168.1.201

E' possible cambiare l' indirizzo IP ed altri parametric quail il numero di periferica e l' indirizzo MAC, mediante il file CONFIG.SYS sull' SD del PLC. (vedi pag.27)

Sequenza di accensione del PLC ARM:

- Se all' accensione é premuto un tasto di funzione viene eseguita la funzione relativa:
- Viene cercato ed eventualmente eseguito il file D:\AUTOEXEC.BAT (su chiavetta USB)
- Viene cercato ed eventualmente eseguito il file C:\CONFIG.SYS (su SD)
- Viene cercato ed eventualmente eseguito il file C:\AUTOEXEC.BAT (su SD)
- Viene cercato ed eventualmente eseguito il file C:\SYS.BIN (su SD)
- Si entra in modo attesa programma, su porte seriali 1 e 2 ed eventualmente su Ethernet se il PLC ha l' espansione Ethernet.

Con riferimento al punto 1 sopra esposto:

- Se il PLC non ha la tastiera, occorre tener premuto il touch screen all' accensione per far apparire la tastiera dei tasti funzione
- Le funzioni previste sono:
- F1 Taratura Touch Screen
- F2 Upgrade del Firmware
- F3 cancella C:\AUTOEXEC.BAT
- F4 Cancella C:\SYS.BIN
- F5 Entra nel programma di test hardware

La porta Ethernet é abilitata in autoconfig:

- puó ricevere messaggi sul proprio indirizzo IP
- Se riceve messaggi verso un indirizzo IP il cui ultimo byte é uguale al proprio numero di periferica + 200, lo prende come proprio indirizzo IP.

I Programmi AUTOEXEC.BAT e CONFIG.SYS sono programmi di BATCH, ed accettano una serie di comandi:

prefissi:

```
@commando       non stampare il comando
:labelname      label di riferimento che precede il comando
REM             riga di commento
IF_EQ a b ...   esegue il comando se a=b
IF_NE a b ...   esegue il comando se a!=b
IF_KEY n ....   esegue il comando se $KEY=n (uguale a IF_EQ $KEY
                n ...)
ON_ERROR ....   esegue il comando se errorlevel>0
```

comandi:

```
BATCH filename pa1 pa2  esegue recursivamente un file batch (pa1->%1,
                        pa2->%2)
EXEC  filename          carica e manda in esecuzione un eseguibile
COPY  source dest       copia un file
DEL   filename          cancella un file
RENAME oldname newname  rinomina un file
CD    directoryname     cambia la directory corrente
MD    directoryname     crea una sottodirectory
RD    directoryname     cancella una sottodirectory
INPUT string t COMANDO  stampa string e attende un carattere ($KEY) da
                        tastiera. Se esiste il parametro t ed e' pas
                        sato un tempo pari al valore di questo pa
                        rametro (in secondi) esegue il comando (es.
                        INPUT opzione 10 GOTO label1)
GOTO  label             salta ad una label
RETURN errorlevel       ritorna settando errorlevel
ECHO  string            scrive un messaggio
ECHO  OFF               sopprime la scrittura per default
ECHO  ON                ripristina la scrittura per default
CLS   n                 cancella lo schermo e punta il cursore all'
                        inizio se e' presente il par. n non cancella
                        le prime n linee.
FONT font dx dy         passa al font specificato
COLOR fg bg             setta i colori del font e di background (se bg
                        omesso = trasp)
LOCATE x y              setta il cursore a coordinate x,y
LLOCATE line col        setta il cursore a linea line e colonna col
LINE x y dx dy col      disegna la linea
RECT x y dx dy col      disegna il rettangolo
RECTFILL x y dx dy col  disegna il rettangolo pieno
PRINT                   equivale ad ECHO
>                       equivale ad ECHO
BAUD num                setta la baud-rate di ricezione seriale
PERIF                   setta il numero di periferica del PLC
MAC_FAMILY              setta I primi 3 bytes del mac address
MAC_NUMBER              setta gli ultimi 3 bytes del mac address
```

I comandi e gli operandi sono separate tra loro da caratteri **spazio** o **virgola**.

La libreria degli oggetti comprende:

Pulsanti, interruttori, commutatori, selettori.
Potenziometri rotativi e sliders
Amperometri,Voltmetri,Wattmetri AC e DC,Cosfimetri.
Visualizzatori di quota
Posizionatori per motori AC e DC.
Controller PID di temperatura.

Altri oggetti fornibili su richiesta:

Modulo di Interpolazione lineare-circolare.
Interprete di interpolazione linguaggio ISO.
Modulo di Levigatura
Modulo di gestione Trifase (Volts RMS, Ampere RMS, Watt, Voltampere, Var, Cosfi per terne asimmetriche squilibrate con e senza terra).
Modulo DSP (filtri Fir, Iir, Adattivi, FFT diretta ed inversa).

Dati tecnici dei programmi realizzati con Proteus

Programma su S400 con monitor V5/V10:

Numero massimo di pagine di un progetto:	50-200
Numero massimo di oggetti per pagina:	1000
Tempo di loop del PLC	1 mSec/1200 linee
N. max. di timers del PLC	65000
N. max. di task	3
N. max. di routines a tempo	32
Risoluzione del display	320X240-1024X768
Dimensioni max. dei bitmap	Dimens. Schermo
N. di colori dei bitmaps	256,32K,Jpeg

Programma su VK3/V8 Arm:

Numero massimo di pagine di un progetto:	1000
Numero massimo di oggetti per pagina:	1000
Tempo di loop del PLC	1 mSec/5000 linee
N. max. di timers del PLC	65000
N. max. di task	32
N. max. di routines a tempo	100
Risoluzione del display	320X240-800X600
Dimensioni max. dei bitmap	Dimens. Schermo
N. di colori dei bitmaps	256,32K,Jpeg

Programma su PC con S400 cieco esterno:
(PC Pentium4 2GHz, 512MB Ram, OS Windows XP)

Numero massimo di pagine di un progetto:	65000
Numero massimo di oggetti per pagina:	1000
Tempo di loop del PLC 1	1 mSec/150000 linee
N. max. di timers del PLC	65000
N. max. di task	16
N. max. di routines a tempo	32
Risoluzione del display	320X200-1024X768
Dimensioni max. dei bitmap	Dimens. Schermo
N. di colori dei bitmaps	256,Jpeg

Per i programmi che girano su PC, il task del PLC gira al di fuori del task switch-ing di Windows, garantendo un funzionamento Hard RealTime, con tempo mas-simo di latenza garantito inferiore a 1 mSec.