La guida di ***VISUAL PARSIC©.3.xx***

For Windows© 3.1/95/98/XP/2000/NT

Compilatore grafico per micro PIC di Microchip©

Tutorial

In caso di necessità potete contattarci ai seguenti recapiti:

Germany :

Swen Gosh +49 (0)4121/22039
www.parsic.de

Italy:

Parsic Italia + 39 0544 927468 fax 178 6040078
e-mail : parsicitalia@libero.it
www.parsicitalia.it

Introduzione.

Compilatore grafico per microprocessori PIC di Microchip

Parsic e' un assembler grafico che permette di gestire in modo automatico la compilazione di un certo numero di ingressi analogici (variabili con continuità), I/O digitali (variabili a due valori) .
La limitazione nel numero di I/O gestibili e' legata al tipo di microprocessore impiegato . Il software permette di configurare blocchi logici software secondo le necessità del progettista, considerando, però, che il programma complessivamente realizzato non superi la capacità di memoria del microprocessore adottato.
E' opportuno, prima di procedere al progetto circuitale definitivo , realizzare uno schema elettrico di principio per avere l'idea, più accurata possibile,della quantità di blocchi logici necessari da impiegare nel progetto e, da qui, procedere alla scelta del microprocessore.

Operazioni pratiche per lo sviluppo di un progetto.

Dopo l'installazione del programma sul vostro PC :

- 1 predisporre i **Settings di Parsic**;
- 2 procedere al progetto collegando i blocchi logici circuitali ;
- 3 procedere alla simulazione funzionale dei blocchi circuitali;
- 4 salvare il file sorgente .**PIC** (schema elettrico) ed il file di testo **.ASM**
- 5 traduzione del file sorgente .**ASM** in file **HEX**, utilizzando MPASM
- 6 programmazione del microprocessore.

Requisiti di sistema

Parsic richiede i seguenti requisiti minimi di sistema:

- computer IBM-compatibile 486 o modello superiore;
- Windows 3.1/95/98/XP/NT/2000;
- memoria tipo harddisk con almeno 5 Mbyte di spazio libero;
- scheda video con risoluzione grafica minima di 800x600;
- mouse doppio tasto.

Parsic versione Demo.

La versione Demo di Parsic consente lo sviluppo di qualunque schema elettrico. L'utente può valutare le potenzialità di Parsic, ma non potrà accedere alle seguenti funzioni:

- conversione dello schema nel formato testo ASM;
- salvare i file sorgente;
- stampare lo schema elettrico.

Inoltre la versione demo non è aggiornata e può contenere altre limitazioni funzionali, qui non specificate.

Parsic versione V3.xx

La versione esecutiva di Parsic consente di sviluppare i progetti senza alcuna limitazione ma, per funzionare correttamente, e' necessario che l'utilizzatore sia in possesso del numero di codice operativo di abilitazione,oltre la chiave Hardware fornita assieme alla copia del CD.
La procedura di concessione delle licenze d'uso e' rigorosa e il possessore di copia non autorizzata di Visual Parsic (duplicazione dei programmi) è perseguito a termine di Legge (DPR 518 del 29 dicembre 1992 e segg.) . I dati identificativi delle licenze di concessione d'uso del programma (installati sul computer) , devono corrispondere ai dati identificativi riportati sul CD originale e relativa licenza in formato PDF. Eventuali discordanze rendono il programma privo di licenza (copia abusiva). Gli unici documenti identificativi del programma riconosciuti dal Distributore Parsic Italia sono:

- il codice programma e codice chiave Hardware riportato sulla licenza ;
- il codice impresso a marcatura laser riportato sul supporto CD originale Parsic Italia;

Per ogni tipo di assistenza tecnica (sostituzione CD danneggiato, sostituzione chiave Hardware,richiesta nuova chiave Hardware aggiuntiva, ecc) , richiesta a Parsic Italia, l'utente dovrà fare riferimento ai codici personali rilasciati nella concessione d'uso. Parsic Italia non procederà ad alcun tipo di assistenza tecnica in caso di mancanza di tali requisiti.

Procedura per l'installazione di Parsic e start del programma

Chiave Hardware:

prima di procedere all'installazione di Visual Parsic**, a computer spento,** collegate alla porta parallela DB25 la chiave Hardware. I possessori di chiave USB collegheranno la stessa in una delle prese disponibili sul proprio computer.

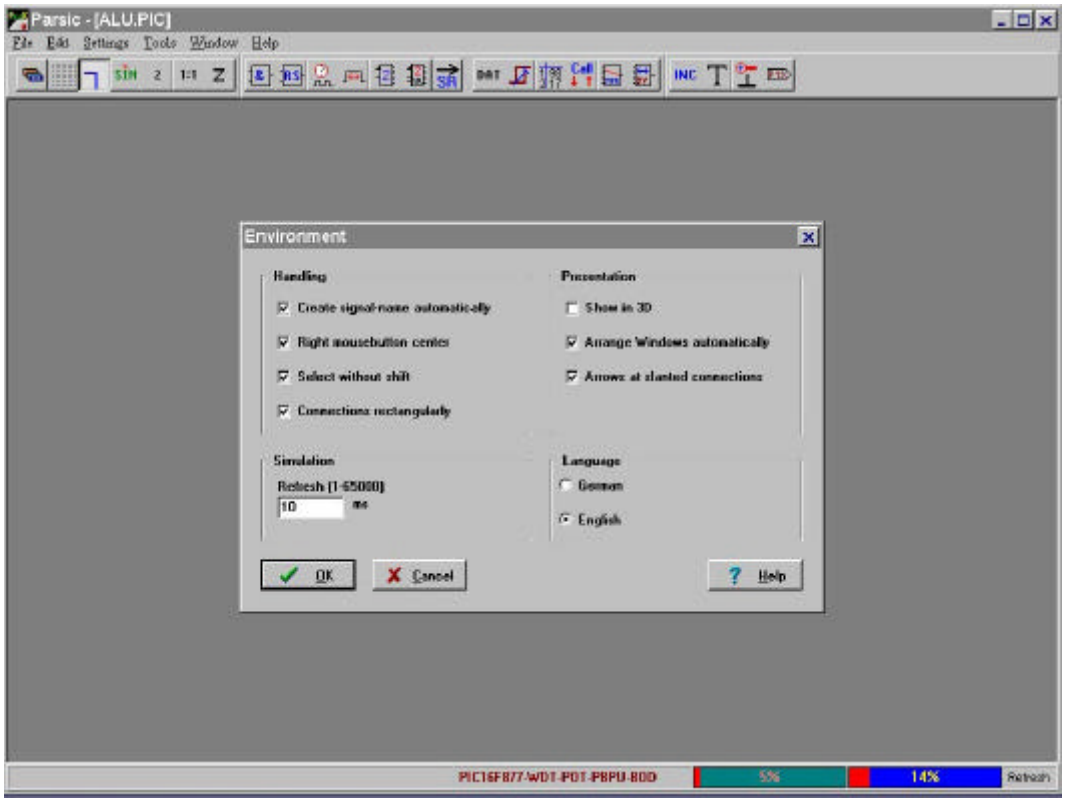
Installazione dei driver della chiave Hardware:

La procedura è automatica. In caso di necessità, installare i driver manualmente che sono posizionati nella cartella C:programmi/Parsic/GSS

L' installazione del programma Visual Parsic è completamente automatica.Posizionate il CD nel lettore ed avviate il Setup.
La procedura di installazione partirà immediatamente dopo l'avvio, e dura qualche minuto.

Predisporre PARSIC all'avviamento: impostazione della lingua operativa.

Le operazioni che adesso saranno suggerite serviranno a **semplificare** l'avviamento del programma, nella prima fase di installazione, senza entrare nel dettaglio funzionale dei **menu**.
Può verificarsi che alla prima installazione Parsic sia operativo in una lingua diversa dall'italiano.Per visualizzare in lingua italiana il programma :
portare il puntatore del mouse sulla barra delle applicazioni, e cliccare prima su **SETTINGS (EINSTELLUNGEN)** e poi su **ENVIRONMENT (UNGEBUNG).**
Aprendosi la finestra di dialogo rappresentata in figura, predisporre l'uso di PARSIC spuntando le varie voci. Se non riuscite a comprendere il significato dei termini in lingua tedesca, copiare i set raffigurati qui' di seguito:



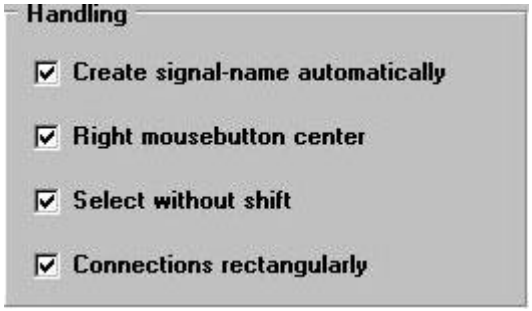
Chiudere le **ENVIRONMENT (UNGEBUNG)** selezionando **OK**; uscire dal programma **(FILE + EXIT oppure DATEI + BEENDEN)** e riavviare. Questa operazione è necessaria per rendere esecutivi i nuovi set .

Menu setting.

Le Environment

Portare il puntatore del mouse sul menu **Settings** e poi azionare il pulsante **Environment**s. Attraverso la finestra di dialogo delle environments, si possono modificare le funzioni operative del programma, elencate qui' di seguito:

nel riquadro **Handling**



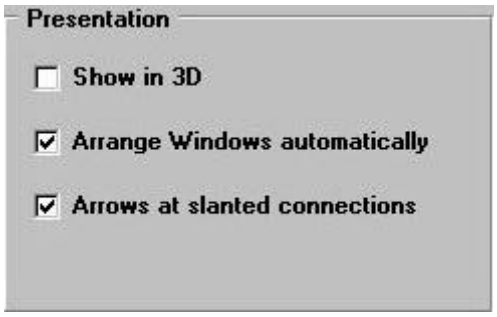
Create signal names automatically: attivando questa funzione le linee di collegamento vengono codificate automaticamente con target significativi diversi. Le linee codificate Sx.x ,sono orientate a segnali con singolo bit, quelle codificate Sx, sono orientate a segnali con singolo byte.

Right mouse button center : attivando questa funzione il cursore del mouse si posizionerà durante il movimento al centro dei blocchi funzionali, oppure non selezionandolo, a margine degli stessi.

Select without shift : quando attivate questa funzione,potrete selezionare gli oggetti o le funzioni accessorie senza l'ausilio del tasto shift. Se la opzione non sarà spuntata la selezione degli oggetti sarà praticata sempre con l'ausilio del tasto shift

Connections rectangularly : attivando questa funzione (consigliata) le connessioni delle linee virtuali sono ad angolo retto. Diversamente esse saranno libere di orientarsi su 360º

il riquadro **Presentation**

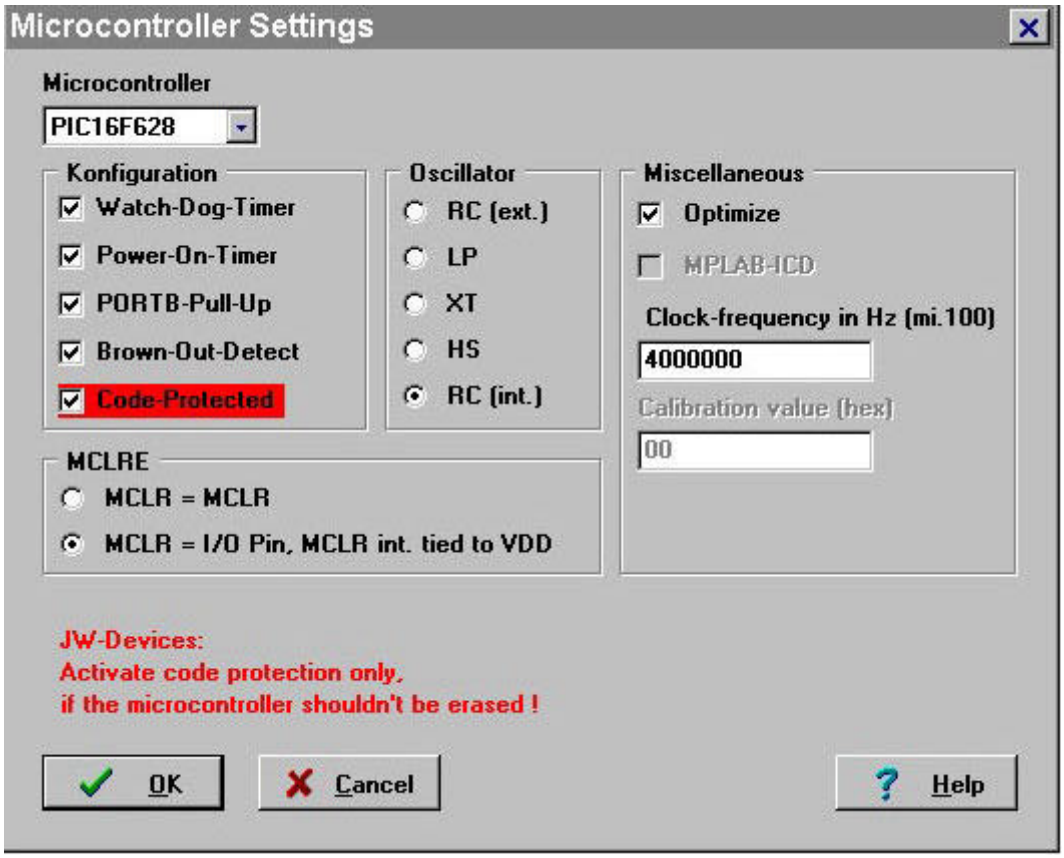


Show in 3D	: selezionando la visualizzazione in 3D gli oggetti e le linee di collegamento saranno presentate in grafica in 3D.
Arrange Windows automatically	: selezionando arrange, le dimensioni del foglio di lavoro sono disposte automaticamente dal programma.
Arrows at slanted connections	: se non e' stato selezionato **connections rectangularly**, l'inizio e la fine di ogni interruzione delle linee di collegamento agli oggetti e' marcata da una freccia direzionale di orientamento. Diversamentele linee di collegamento resteranno collegate agli oggetti in movimento su qualunque piano del foglio di lavoro

Simulations Refresh ms	: impostando un periodo compreso tra 1 e 65000 ms, durante la fase di simulazione si determina il l'aggiornamento degli stati logici dei blocchi funzionali e delle linee di collegamento circuitale
Language	: e' la scelta della lingua operativa del programma.

Menu' setting. Microcontroller.

Posizionare il puntatore del mouse sul menu **SETTINGS** e poi trascinarlo su **MICROCONTROLLER,** apparirà la seguente finestra di dialogo :



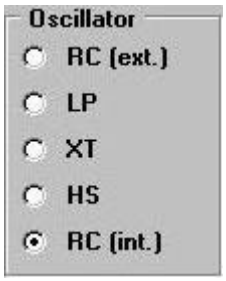
Selezionando il pulsante raffigurato qui di seguito si potrà scegliere uno dei 50 tipi di PIC che il programma e' in grado di assemblare.

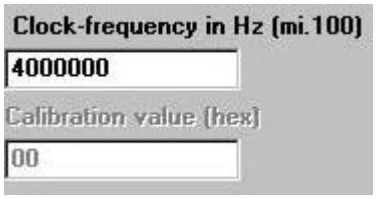
Elenco dei PIC selezionabili:

PIC16C54C	**PIC16C71**	**PIC16F84**	**PIC16C620A**	**PIC12CE508A**
PIC16C55A	**PIC16C72**	**PIC16F84A**	**PIC16C621A**	**PIC12CE509A**
PIC1656A	**PIC16C72A**	**PIC16F627**	**PIC16C622A**	**PIC12CE518**
PIC16C57C	**PIC16C73**	**PIC16F628**	**PIC16CE623**	**PIC12CE519**
PIC16C58B	**PIC16C73A**	**PIC16F873**	**PIC16CE624**	**PIC12C671**
PIC16C61	**PIC16C73B**	**PIC16F874**	**PIC16CE625**	**PIC12C672**
PIC16C62B	**PIC16C74**	**PIC16F876**	**PIC16C711**	**PIC12CE673**
PIC16C63A	**PIC16C74A**	**PIC16F877**	**PIC16C712**	**PIC12CE674**
PIC16C64A	**PIC16C74B**		**PIC16C715**	
PIC16C65B	**PIC16C76**		**PIC16C716**	
PIC16C66	**PIC16C77**			
PIC16C67	**PIC16C84**			

Menu setting. Microcontroller : l'oscillatore.

Nel riquadro **Oscillator** si imposta il tipo di oscillatore che si desidera impiegare e la sua frequenza di funzionamento :





RC (ext)	:	la frequenza di clock e' determinata dalla rete RC collegata al PIC;
LP	:	low-power osc. risonatori con range fino a 200 KHz ;
XT	:	quarzi o risonatori con range da 200 KHz a 4 MHz ;
HS	:	quarzi o risonatori con range da 4 a 20 MHz ;
RC	:	oscillatore RC interno (non disponibile su tutti i PIC).
Clock frequency	:	la frequenza di clock dell'oscillatore in Hz ;

Calibration value : questa opzione e' importante per quei Pic equipaggiati di oscillatore interno e che sono dotati di finestra di cancellazione (**Jw-tipe** versione finestrata del PIC) . Il valore (**hex**) da inserire nell'opzione **calibration** è indicato nelle specifiche Microchip. Dopo il reset del Pic, questo valore viene scritto nel **OSCCAL-register** in modo che l'oscillatore interno funzioni il più vicino possibile a 4 MHz. Un parametro elevato del **Calibration value** (**hex**) produce un corrispondente elevato valore d'oscillazione. Questo valore deve essere posto alla fine dei dati contenuti nella ROM, sotto forma di comando e deve essere visto dal microprocessore prima dello start del programma.

Configurazione .

Watch-Dog Timer : letteralmente "cane da guardia".Interviene nel caso il programma si blocchi. L'impulso di reset viene prodotto dopo un periodo di circa 18ms, dall'arresto del programma.

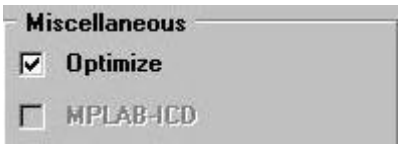
Power On Timer : se attivato,ritarda per un periodo di 72ms circa, lo start del micro, dopo il ciclo di reset.

PORTB-PullUp : se attivato, i PortB RB.0....RB.7 vengono collegati internamente alle resistenze di pull-up (+Vdd).

Brown-Out Detect : attiva il reset del micro in caso che la tensione di alimentazione scenda ad un valore inferiore a 4V. L'attivazione di quest'opzione porta ad un aumento del consumo di corrente di circa 500 µA.

Menu setting. Il Master clear (MCLR).

MCLR (Master Clear)	: selezionando **MCLR = MCLR** le operazioni di reset del micro avvengono collegando elettricamente questo terminale al circuito esterno.
MCLR = I/O Pin,MCLR int. tied to Vdd	: il **MCLR** è utilizzato come terminale I/O . Lo stesso **MCLR** è collegato internamente al mcp, via software, all'alimentazione +Vdd

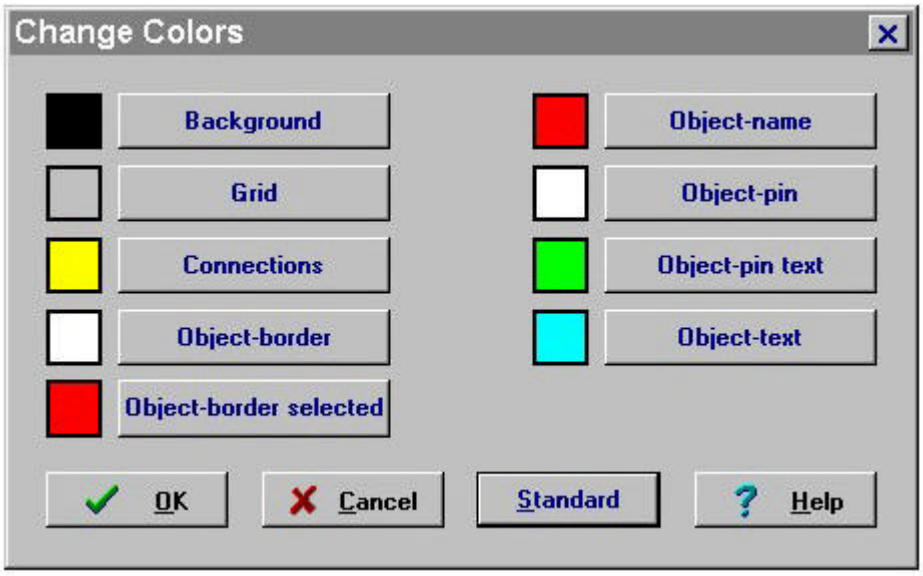


Optmize
Attivando **Optmize**, il compilatore procede alla migliore verifica del listato assembler eliminando le linee di programma superflue.

MPLAB-ICD
Se MPLAB e' attivo sono liberate le seguenti aree RAM e ROM per l' Iincircuit-Debagger:
NOP comand all'indirizzo 0;
Indirizzo $70 ed $1EB nella RAM (1EF non e' usato da Parsic)
Le ultime 256 words dell'area ROM non sono usate.

Menu setting . Colors: come impostare i colori del piano di lavoro.

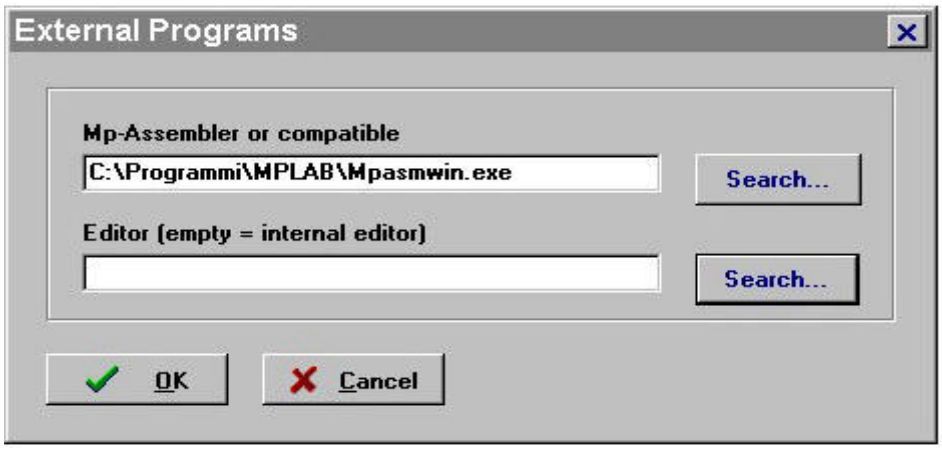
Muovere il puntatore del mouse sul menu **SETTINGS,** cliccare sul pulsante **Colors**
Nella prima installazione del programma quest'operazione è spesso necessaria. Non trascurare questo passaggio, diversamente in alcuni casi non sarà possibile vedere gli oggetti e le linee di collegamento sullo schermo.



Suggeriamo,inizialmente, di operare con i colori **standard** di sistema .

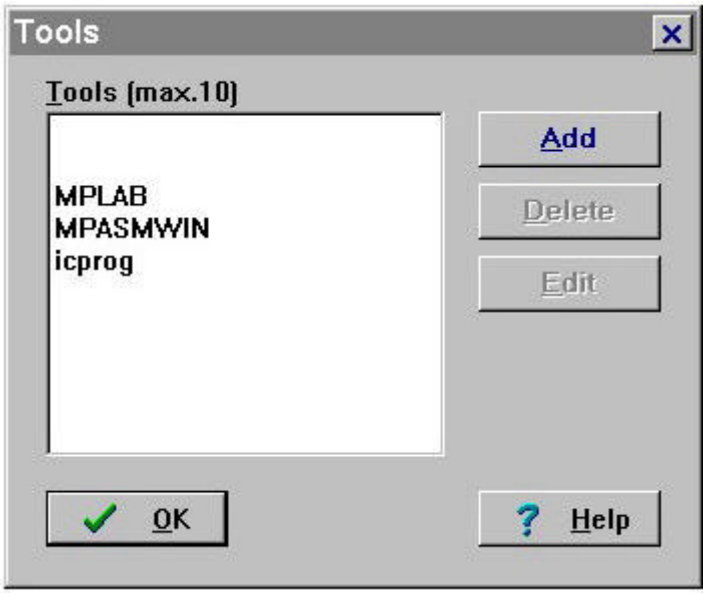
Menu setting. Programs.

PARSIC consente di operare con 10 programmi applicativi diversi come, ad esempio, **MPLAB.** Portarsi ancora con il puntatore su **SETTINGS** e poi su **PROGRAMS**. Inserire il nome/percorso del programma **MPLAB (MPASMWIN**) completo della sua estensione,cosi come raffigurato.
Utilizzare **SEARCH**, per individuare nella libreria il percorso che PARSIC deve effettuare per aprire **MPASWIN** o un altro programma similare.Utilizzando questa procedura, al termine di un progetto, dopo l'operazione **SAVE (Save As..)**, provare ad attivare la funzione **BUILD.. . (F10).**
Parsic provvederà a salvare il file sorgente **.ASM** del circuito in progetto e lancerà automaticamente **MPASM .** Il programma, tra le altre funzioni, produce un file con estensione **.PJT** utile ad operare con **MPLAB**. Si potrà lanciare **MPLAB** oppure **MPASMWIN** o **ICPROG**, direttamente dalla barra delle applicazioni operando sul pulsante **TOOLS** (collegamenti) .



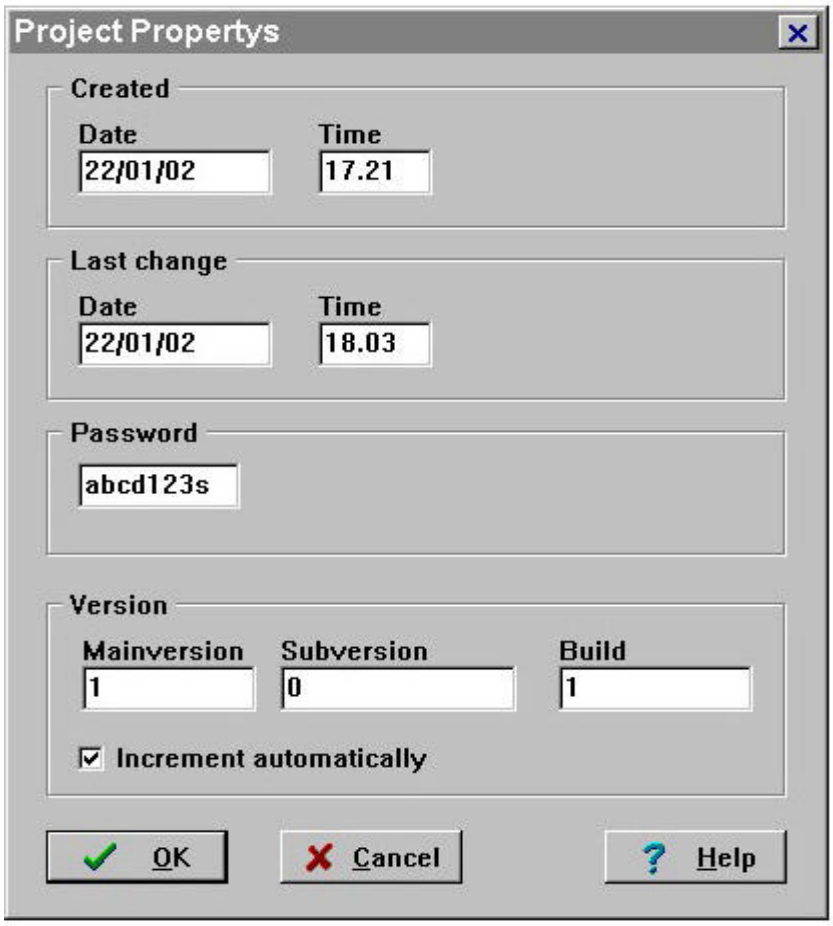
Menu setting. Collegamenti ad altri programmi : Tools.

Portare il puntatore del mouse sul menu **SETTINGS** e poi trascinarlo su **Tools**, cliccare. All'interno della finestra di dialogo rappresentata in figura, potrete inserire i programmi che più utilizzate durante le operazioni progettuali.



Menu setting. Project Propertys , i dati di progetto.

Spostare il puntatore del mouse sul menu **SETTINGS** e poi azionare il pulsante **Project**. I dati contenuti in questa maschera sono relativi al progetto che si vuole iniziare. Questi dati saranno aggiornati automaticamente ad ogni nuova sessione del progetto, se nel riquadro **Increment automatically** verrà spuntata la relativa casella. Inserendo la **Password** il progetto è protetto da accessi non autorizzati (impiego in multiutenza di Parsic).



Mouse click.

Gran parte delle attività del programma sono svolte impiegando il mouse.
Questo accessorio svolge l'importante funzione di accedere alle funzioni del programma con il semplice "click" dei suoi pulsanti. Il puntatore del mouse, secondo la sua posizione, assume tre funzioni diverse:

- selezione di una funzione programma o di un blocco logico;
- selezione di un collegamento circuitale ;
- selezione della modifica di un blocco logico (movimento del b.l. in un punto qualunque dello schema, cancellazione del b.l. , modifica di parametri,ecc.)

La selezione di un menu avviene portando il cursore su una delle risorse che si desidera attivare (file, settings,tool, ecc.), mentre la selezione di un blocco logico funzionale dello schema elettrico, si attiva posizionandosi sui pulsanti della toolbar.
In ogni caso, dopo essersi posizionati su una qualsiasi funzione, si dovrà agire sul tasto sinistro del mouse per attivarla.
Operando sui pulsanti della toolbar, selezionata una funzione (tasto sinistro), si dovranno lasciare i tasti del mouse liberi e si dovrà proseguire trascinando l'oggetto nello schema elettrico. Una volta in posizione , cliccare col tasto sinistro del mouse per visualizzarlo. Più avanti, questa manovra sarà descritta dettagliatamente. Il cursore del mouse modifica la propria simbologia in base alla posizione assunta sul blocco logico:

 portando il cursore sopra un blocco funzionale , attivando l'icona raffigurata a lato, si potranno ottenere le seguenti funzioni :

- mantenendo premuto il tasto sinistro del mouse, il blocco logico può essere spostato in un punto qualunque dello schema elettrico;
- cliccando una sola volta sul blocco logico con il tasto sinistro, e premendo sul tasto delete, esso può essere rimosso (cancellato) dallo schema elettrico;
- portando l'icona sul bordo dell'oggetto e cliccando il tasto destro, si attiva la finestra di dialogo dalla quale e' consentito modificare i dati funzionali del blocco logico.

Utilizzare Cntrl +  per spostare le connessioni.

Questa icona si attiva quando il cursore del mouse e' posizionato su una delle terminazioni del blocco logico. Mantenendo il tasto sinistro premuto e spostandosi con il cursore, procederemo al collegamento elettrico del terminale; invece, rimanendo sullo stesso terminale , cliccando con il tasto destro, procederemo alla identificazione del terminale, attraverso l'attivazione della finestra di dialogo che apparira' subito dopo la selezione (**Anschluss info**).

Uso delle combinazioni dei tasti.

Durante la costruzione dello schema elettrico l'uso combinato dei pulsanti della tastiera facilita ed abbrevia le operazioni di Windows. A lato di ogni pulsante funzionale,nella barra dei menu , sono riportate le sequenze dei comandi abbreviati di ogni singola funzione.

Uso dei pulsanti della Toolbar.

La toolbar si presenta come un complesso di pulsanti, sotto forma di icone, che vengono azionati con il tasto sinistro del mouse, quando il cursore e' posizionato su di essi. La toolbar si divide in due gruppi di selezione :

questo pulsante consente di selezionare il gruppo pulsanti della toolbar ;

inserisce o disabilita la **Griglia** di riferimento sullo schermo;

definisce la specifica del collegamento : si consiglia di lasciarlo sempre inserito;

quando azionato avvia il processo di **Simulazione** del circuito in progetto;

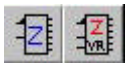
la funzione di **Zoom** in/out dello schema elettrico;

selezione degli operatori **AND, OR, XOR**;

selezione del **Flip-Flop** Set-Reset;

selezione del **Timer** o base dei tempi (clock);

selezione del blocco **Monostabile** su evento;



Contatori con risoluzione a 8 oppure 16 bit ;

Schift-register;

Selezionatore ingresso analogico o costante digitale;

Blocco comparatore **Schmitt - trigger;**

gestione **Tabelle e Soubroutine**;

Multiplexer ;

De-codificatore ;

funzione **Include** ;

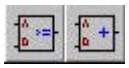
editor;

potenziale di riferimento ;

etichetta **Label** ;

Interrupt , PWM, PFM;

monostabile tipo **One-Shot** ;



Comparatore digitale ed **Operatore Matematico** ;

gestione **EEPROM**;

funzione **Limiter** o filtro digitale;

modulo **LCD** ;

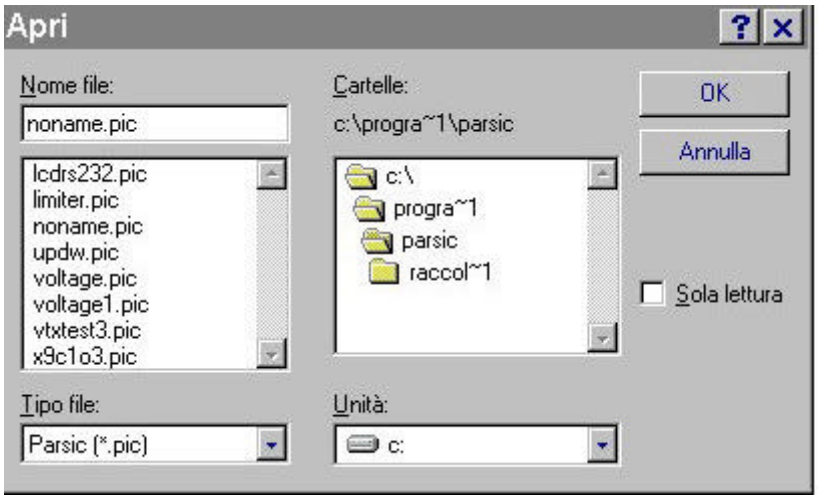
funzione **Sleep** ;

gestione linea seriale **RS232 e 485**.

La descrizione funzionale dei blocchi logici e' spiegata dettagliatamente in avanti.

Load dei file, selezione dello zoom in-out. La Function Plan.

Per accedere ad un progetto precedentemene memorizzato,portarsi con il puntatore del mouse sul menu **File** e poi scorrere fino al pulsante **Open**, cliccare. Parsic riconosce i file con estensione **.pic** che,una volta selezionati, li visualizzera' sul display.

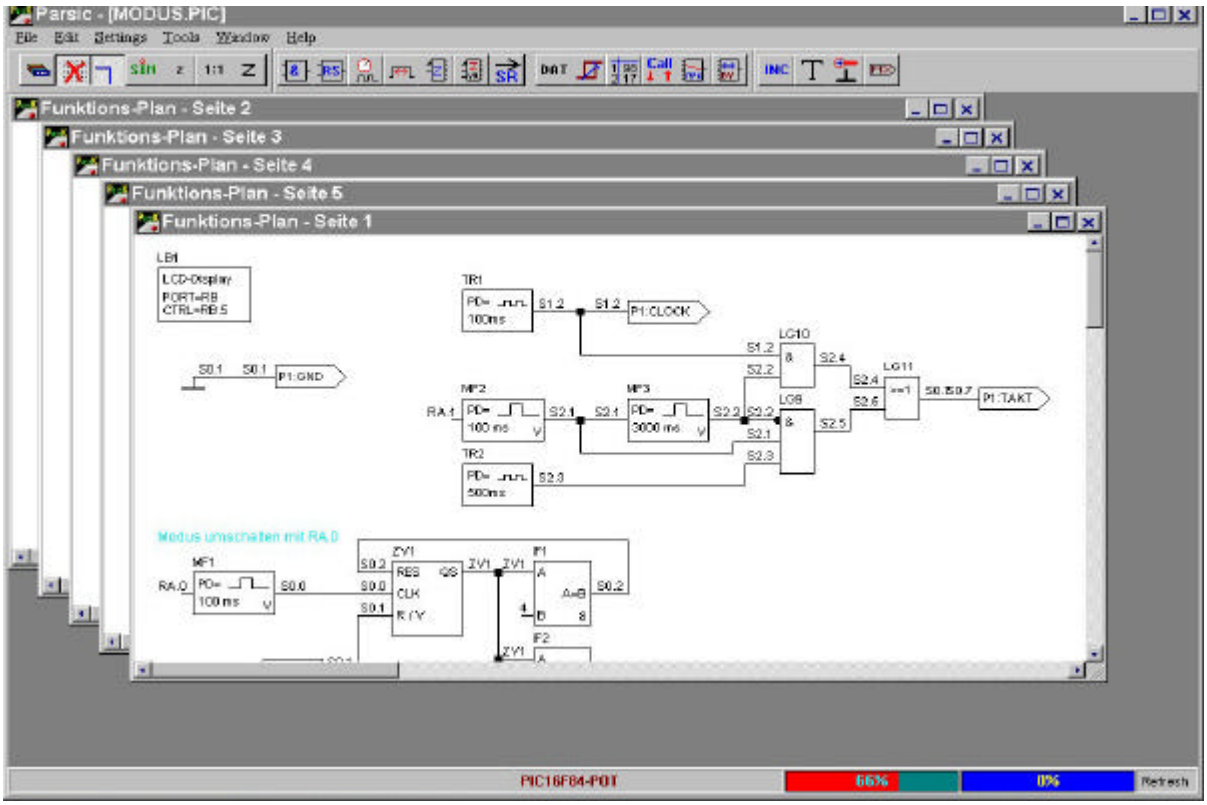


Se lo schema elettrico ha dimensioni molto grandi, tanto da individuare con difficolta' i blocchi circuitali, e' bene usare i tasti (**Zoom**) **z** (in) e **Z** (out) **.** In caso di progetti che richiedo estensioni circuitali molto ampie, si consiglia di usare l'opzione **Function Plan** del menu **Window** , che permette la distribuzione dello schema elettrico su piu' fogli schematici. Attivando questa funzione, si raccomanda di utilizzare il blocco funzionale **Label**, necessario per collegare le terminazioni elettriche tra una pagina e l'altra.

Vedi funzione Label 

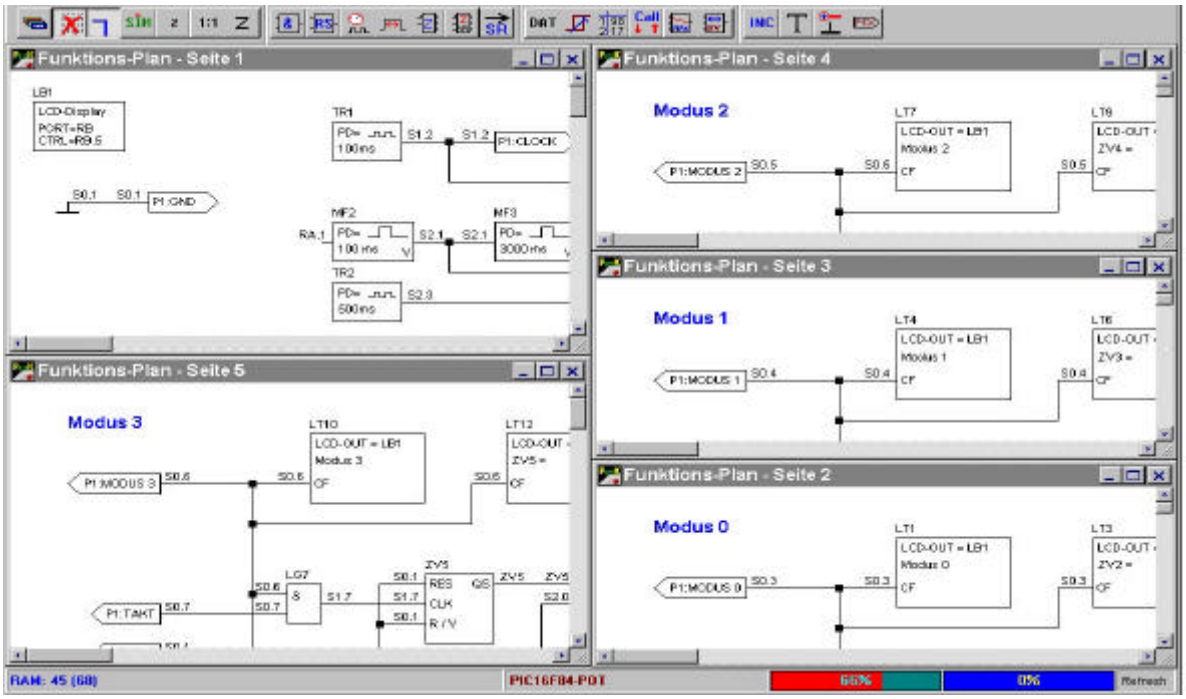
Il menu Window : gestione dei fogli schematici.

Le utilita' del menu **Window** consentono di lavorare su schemi di grande dimensioni, operando su piani di lavoro diversi. Dopo avere aperto il file dimostrativo **Modus.pic**, portare il puntatore del mouse sul menu **Window**, cliccare con il tasto sinistro spostando il puntatore su **Cascade** (Shift + F5). Lo schema elettrico del file Modus.pic sara' distribuito su piu' pagine,come mostrato nella figura seguente:



Posizionandosi sul bordo delle **Funktions-Plan**, si potranno visualizzare le pagine come se scorressero l'una su sull'altra.
Le terminazioni di continuita' elettrica, tra le pagine, sono distinte dall' etichetta di identificazione **Label .**

Portandosi di nuovo sul menu **Window** e posizionando il cursore del mouse sul pulsante **Tile** (SHift + F4)**.** gli schemi elettrici si disporranno come nella figura seguente:



I pulsanti **Minimize ed Arrange** servono a ridurre le pagine ad icone,il primo, mentre il secondo pulsante a disporre in maniera lineare il piano di lavoro.
Per ritornare sulla pagina iniziale, a schermo pieno,e' necessario riportare il puntatore del mouse sul menu **Window** e poi selezionate **Special**. Utilizzando la Funktions Plan, **per commutare le pagine**, utilizzate i tasti numerici.

Selezione della griglia e dei collegamenti virtuali.

Selezionando la griglia di posizionamento, questa potrebbe essere di aiuto per posizionare ordinatamente i componenti nello schema elettrico. I collegamenti elettrici si dispongono con angolazioni a **90º** se nel menu **Settings/Environments/Handling** e' stata spuntata la voce **Connections rectangurarly** . Se non viene selezionata la voce **Connect...** , allora il collegamento elettrico delle linee e' libero di orientarsi su 360º : consigliamo il collegamento a 90º.
Consigliamo pure di mantenere attivo il tasto di selezione di collegamento virtuale, per evitare di visualizzare sullo schermo una miriade di etichette di collegamento.

Il comando Delete.

Precedentemente, nel paragrafo **mouse click**, e' stato spiegato come procedere alla cancellazione di un oggetto o di una linea di collegamento.
L'azione e' valida se nel menu **Setting/Environments/Handling** e' stata spuntata la voce **Select without shift.** Gli oggetti dello schema elettrico si possono cancellare diversamente, selezionando questi tenendo premuto il tasto **shift** della tastiera e premendo contemporaneamente il tasto **sinistro del mouse**. La rimozione si verifica premendo il tasto **delete** oppure portandosi con il cursore del mouse sulla barra dei menu, selezionando il menu **Edit** e poi **Delete**. Consigliamo di mantenere spuntata la voce **Select without shift** .

Volendo procedere alla cancellazione di vaste aree del circuito elettrico, posizionandosi con il cursore del mouse su un punto periferico dello schema , mantenere il tasto sinistro premuto, poi muoversi con il cursore verso il perimetro opposto dell'area in cui vi trovate.
Mano a mano che si procede, gli oggetti che sono contenuti all'interno dell'area tracciata dal cursore del mouse saranno selezionati . Lasciando libero il tasto sinistro del mouse, per eliminare gli oggetti selezionati è necessario adoperare il tasto **delete** . Volendo procedere alla pulizia dello schermo, portare il puntatore del mouse sul menu **Edit**, trascinandolo poi sul pulsante **Select Oll**, agire sul tasto **delete** per eliminare tutti gli oggetti dalla schermo.

Selezione con shift.

Per selezionare un'area dello schema elettrico,mantenere premuto il pulsante shift della tastiera e premere contemporaneamente il pulsante sinistro del mause. Spostare ora,il cursore del mouse lungo l'area dello schema che si vuole selezionare. Quest'area è evidenziata dagli oggetti che si colorano di rosso. Mantenere il pulsante shift della tastiera premuto e premere il pulsante delete per cancellare il contenuto selezionato.

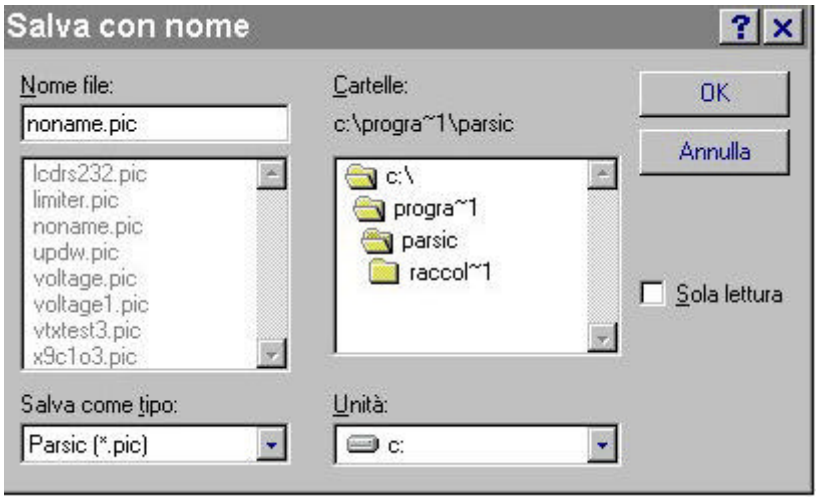
Copia di una porzione di schema.

Per duplicare una porzione o tutto lo schema elettrico,procedere in questo modo.
Non spuntare Select without shift .

Mantenere premuto il tasto shift della tastiera ed il tasto sinistro del mouse. Con il puntatore del mouse selezionare l'area dello schema che si vuole riprodurre. Lasciate il tasto shift ed il tasto del mouse dopo la selezione.
Premete ora in sequenza sulla tastiera i tasti Control e C e dopo i tasti Control e V.
Si noterà che l'area selezionata si doppierà. Portate il puntatore del mouse su uno qualunque degli oggetti doppiati e trascinate l'oggetto, mantenendo il tasto sinistro del mouse premuto, nell'area in cui si desidera copiare lo schema. Rilasciate il tasto del mause e deselezionate con un doppio click .

Il comando Save.

Si raccomanda di usare spesso il comando Save durante la fase progettuale. Vi mette al riparo da errori di manovra, involontaria perdita dei dati, disattenzioni....
Per procedere, portando il puntatore del mouse, sul menu **File** selezionare **Save (Save as..)**. Il nome che si dovra' dare al file non potra' essere piu' lungo di 8 lettere e deve necessariamente avere l' estensione **.pic (**Parsic non riconosce altre estensioni) .La selezione abbreviata per questa operazione e' **Cntr + S .**
Prendere la buona abitudine di salvare il file sorgente, dalla prima fase del vostro lavoro, ossia dalla selezione **New**. Il figura seguente mostra la finestra di dialogo **Save:**



Salvare il file di testo ASM.

Per salvare il file sorgente **ASM** (file di testo) prodotto da Parsic, portare il puntatore del mouse sul menu **File e poi** selezionare **Save Source (Save Source as..)** . Il nome che si dara' al file non potra' essere piu' lungo di 8 lettere e deve avere l' estensione .**ASM**. La selezione abbreviata per questa operazione e' **Cntr + Q.**

Conversione del file di testo nel file compilato .HEX

La procedura per ottenere la conversione del file di testo .**ASM** in formato **Intel Hex** e' molto semplice. Con Parsic e' possibile applicare due metodi per ottenere lo stesso risultato.
Il primo consiste nell'avviare **MPASM**, selezionandolo dal menu **Tools.** All'apertura del menu, attivare **MPASM** e procedete nell'elaborazione del file **.ASM**, che si trova posizionato nella directory di **Parsic**. Si procede,da questo punto in poi, secondo le istruzioni di **MPASM**.
Il risultato che si ottiene, a video, e' quello di una rapida presentazione del processo di traduzione del file **.ASM** ,che terminera' con la produzione di una serie di file, cosi' di seguito elencati, che saranno memorizzati nella directory di Parsic:

nomefile. HEX
nomefile. LST
nomefile. ERR
nomefile. COD

In caso di errore, il file .HEX non sara' prodotto e l'utente dovra' utilizzare il file .ERR per individuare l'anomalia.

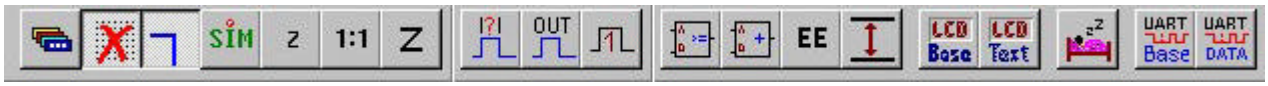
Il secondo metodo : dopo aver salvato il file con estensione **.pic** , portarsi col il cursore del mouse sulla barra delle applicazioni, selezionare il menu **File e poi Build**... (selezione rapida **F10**)
Si aprira' la finestra di dialogo, **Salva con nome** e si procedera' a salvare il file di testo con la stessa procedura vista in precedenza. Completata l'operazione dare **OK**. Si notera', nell'immediato, che il programma avviera' **MPASM** e che si produrranno, in conseguenza , gli stessi **file** di cui sopra.
Ricordiamo che, affinche' questa procedura abbia successo, si dovranno seguire le operazioni di set suggerite al paragrafo, **Menu Settings : Programs** e **Tools**

Il comando SIM.

Prima di attivare la simulazione funzionale dei circuiti, e' necessario che venga impostato correttamente il tempo di campionamento dei segnali, che dovra' essere compreso tra **1 e 65000** ms (**Simulation refresh**). Diversamente, la funzione di simulazione dei circuiti potrebbe non funzionare .
Per attivare il processo bisogna portare il cursore del mouse sulla toolbar e azionare il pulsante **SIM**. Le linee di collegamento elettrico virtuale si attiveranno, cambiando tonalita' di colore, e si potranno notare le variazioni dello stato logico dei dispositivi . Durante la fase di simulazione si puo' intervenire sui componenti logici, forzando il livello dei segnali in circolo per mezzo del puntatore mouse, posizionandosi **direttamente all'ingresso** delle terminazioni dei blocchi logici Se durante il processo di simulazione si attivano i messaggi di errore, il processo verra' interrotto : procedete alla correzione dei circuiti, secondo le indicazioni che verranno date.

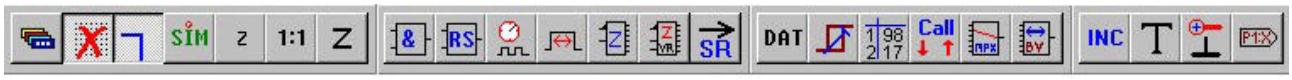
Costruzione dello schema a blocchi.

L' esempio pratico che segue, e' utile a comprendere come procedere alla costruzione di uno schema logico funzionale. La **toolbar** consiste in un complesso di pulsanti con la forma di icone , che rappresentano i **blocchi logici funzionali** del programma. I pulsanti, quando azionati, modificano il loro aspetto cambiando colore. Il programma **non ammette** la selezione contemporanea di più blocchi fuzione.



La procedura di costruzione di uno schema funzionale a blocchi è molto semplice e si basa sul funzionamento delle black-box. Posizionarsi con il cursore sulla toolbar e selezionare (ad esempio) la funzione booleana **AND** .Essa si presenta, appunto, come una scatola nera la cui caratteristica e' il segno grafico che la distingue, ed un numero ancora non definito di terminazioni elettriche in-out.
Una volta selezionata, questa dovra' essere trascinata e liberata in un punto qualunque dello schermo, cosi' come avviene negli ambienti di programmazione grafica con la funzione **Drag & Drop**. Selezionata la funzione, attaccato al puntatore del mouse, vi apparirà un oggetto di forma geometrica regolare. **Non azionate più i tasti del mouse** ma trascinate l'oggetto in un'area libera dello schermo. Trovandovi nella posizione desiderata, azionare **UNA SOLA VOLTA** il pulsante sinistro del mouse : l'oggetto viene liberato e visualizzato. **Per svincolare il puntatore del mouse, cliccare una volta il tasto destro.**

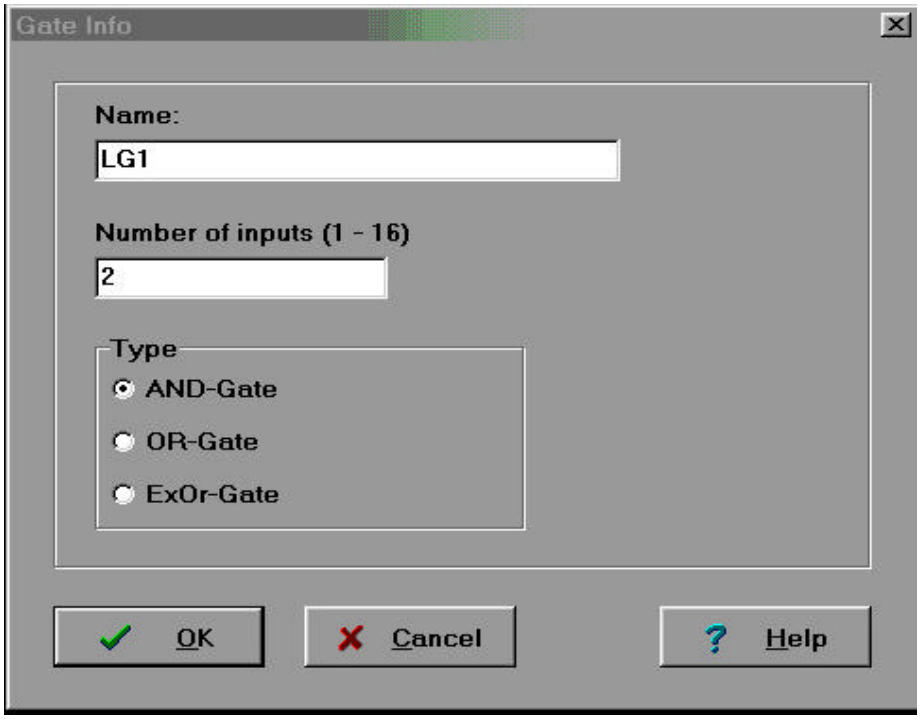
Esecuzione pratica:



portare il puntatore mouse sul pulsante  e selezionarlo

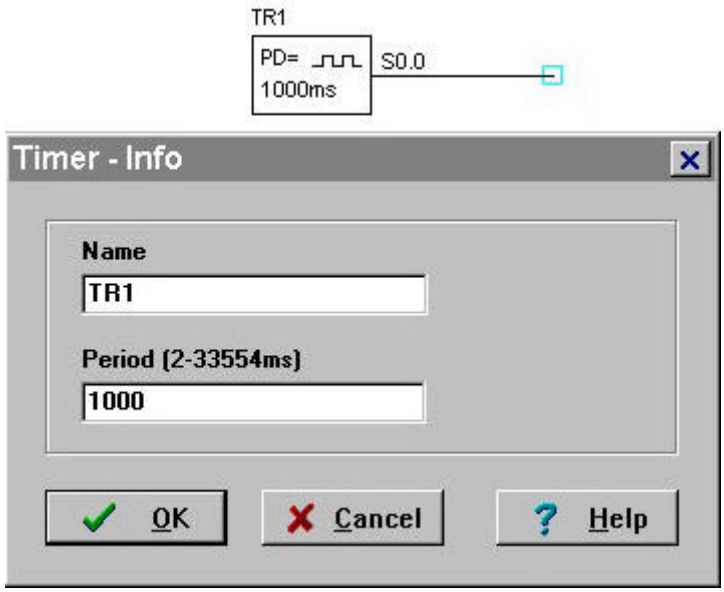
senza agire piu' sui pulsanti trascinare l'oggetto in un'area libera dello schermo. Premendo una sola volta il pulsante sinistro del mouse esso verra' liberato. Cliccate sul tasto destro per svincolare il puntatore mouse dalla funzione selezionata.
Portandosi con il puntatore sull'oggetto, esso cambia la geometria della sua icona secondo come viene posizionato. (vedere il paragrafo **mouse click)**.

Quando il puntatore e' sopra l'oggetto, provare a cliccare con il tasto destro del mouse .Deve apparire questa finestra di dialogo nella quale si puo' modificare le caratteristiche elettriche dell'operatore booleano precedentemente selezionato. Si puo' scegliere un operatore booleano, diverso da quello che adoperemo nell'esempio, e si puo' modificare il numero dei gate di ingresso (massimo 8) Il gate di uscita non e' modificabile.Questa opzione consente di creare funzioni logiche booleane complesse:



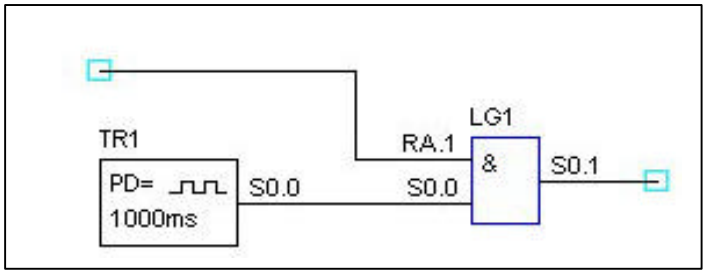
(lasciare inalterata la logica AND e procedete nell'esempio).

In caso di errore volendo eliminare un blocco funzionale o una linea: portarsi con il puntatore del mouse sull'oggetto, cliccare con il tasto sinistro. L'oggetto modifichera' il colore del suo contorno geometrico: azionare il tasto delete sulla tastiera del computer per cancellare il dispositivo. Il funzionamento del tasto **delete** della tastiera dipende dalle predisposizioni che prima sono state impostate sul pannello **Settings + Environment**. Controllare che sia spuntata la casella **Select without shift**.
Posizionato l'operatore booleano **AND**, provare a collegarlo ad un **PORT** di ingresso del PIC ed a un generatore di **clock**. Posizionarsi sui pulsanti della toolbar, selezionare il generatore di **clock (TIMER)** e posizionarlo nelle vicinanze della logica **AND**.
Ora, provare a cambiare il **set del generatore**, posizionandosi su di esso con il puntatore mouse. Quando appare la finestra di dialogo, di seguito raffigurata, portare **il time base** del timer a **1000ms** (1 sec).



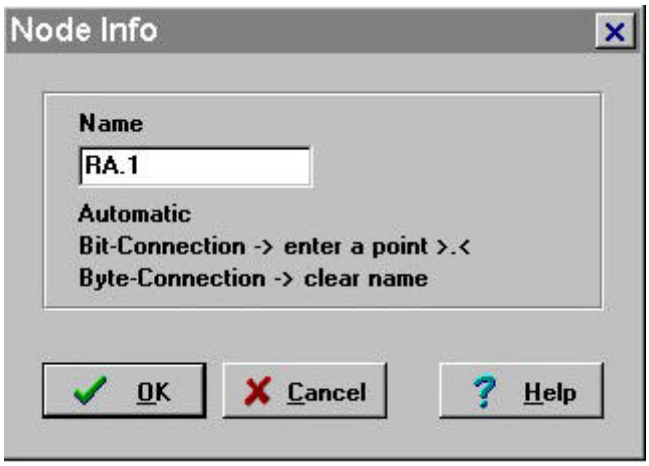
Con il cursore del mouse, collegare il teminale di uscita del timer ad uno dei pin di ingresso dell'operatore booleano. L'altro pin di ingresso della logica **AND**, sarà collegato al gate di ingresso del pic: **PORT RA.1**

Il circuito, al termine dei collegamenti, dovrà apparire come in figura:

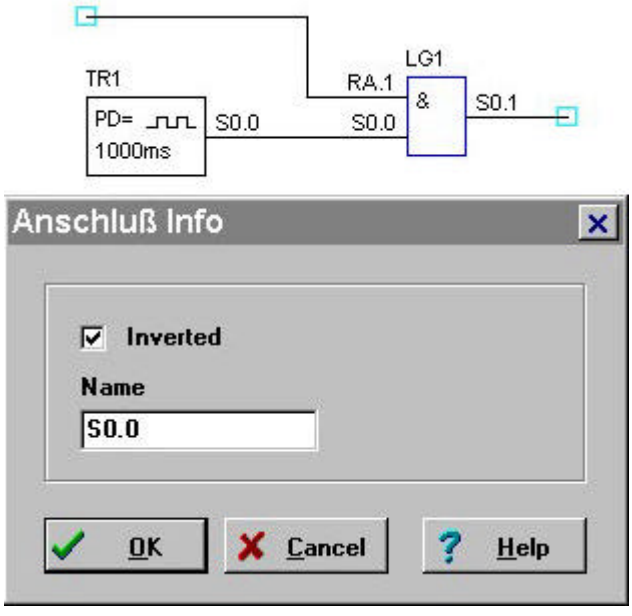


Attenzione, per specificare che **LG1**, pin 1, è collegato al **PORT RA.1**, bisogna portare il puntatore del mouse sul terminale **1** della logica **AND**. Quando il puntatore del mouse ssarà posizionato sul

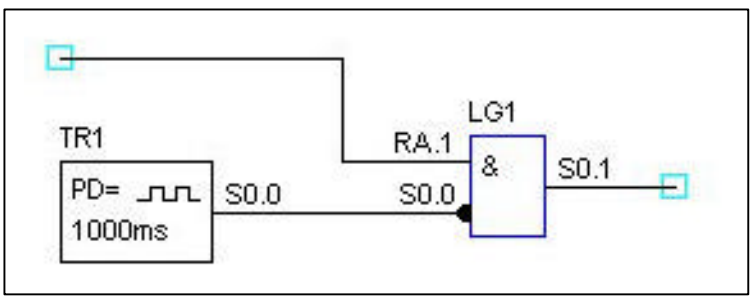
terminale della logica, azionare il tasto destro. Aprendosi la finestra di dialogo definire l' indirizzo di collegamento del **PORT RA.1,** come mostrato nella figura seguente:



E' importante che tutti i gate d'ingresso o di uscita del micropic siano identificati con i nomi **RA.x, RB.x, RC.x ecc.,** riconosciuti dal Programma**.** Accertarsi **sempre** che queste indicazioni siano correttamente posizionate nei terminali dello schema elettrico funzionale, in particolare, prima di lanciare il programma **MPASM** .
Si può collegare un operatore booleano tipo **NOT** all'ingresso o all'uscita di uno qualsiasi degli oggetti logici utilizzati da PARSIC: ecco un esempio. Portare il puntatore del mouse sul collegamento di ingresso di **LG1, S0.0,** cliccare con il tasto destro, fino a che appare la seguente finestra di dialogo: spuntare **INVERTED.**

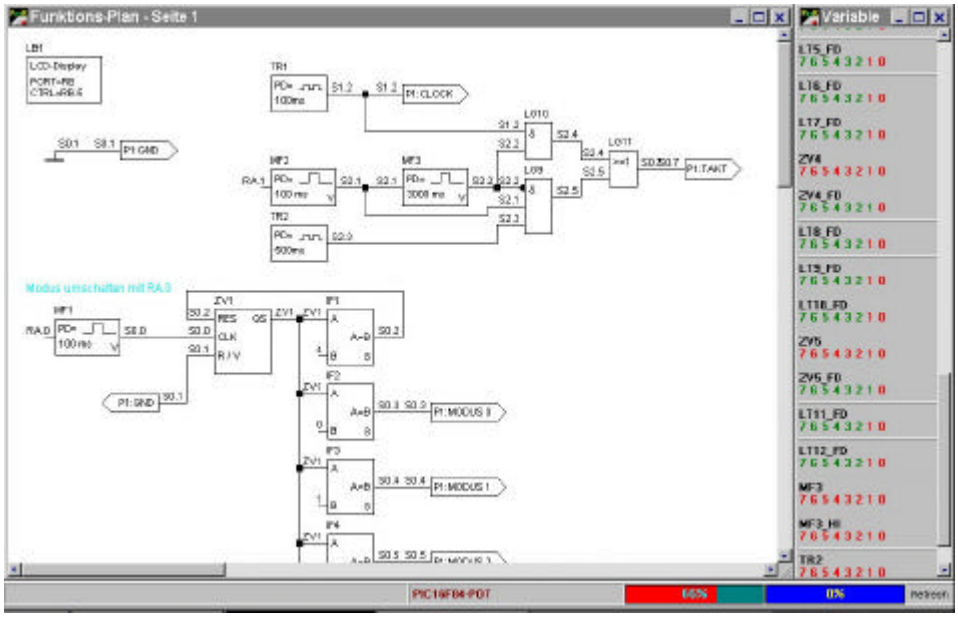


si otterrà il seguente risultato:

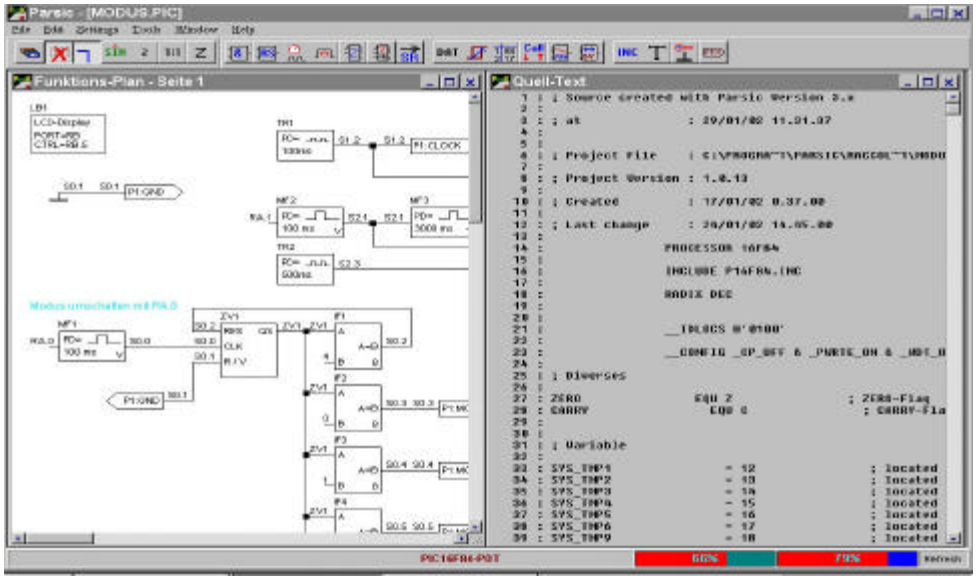


Una volta realizzato questo semplice circuito logico, si può procedere al collaudo funzionale dello stesso. Portarsi con il puntatore sulla barra delle applicazioni, premere il tasto **SIM.**
Si procederà, alla **simulazione funzionale del circuito**. Nel caso siano stati commessi degli errori nei collegamenti o di impostazione dei blocchi logici funzionali, vi appariranno alcuni **messaggi di errore**. Alcuni messaggi sono in lingua tedesca, ma sono facilmente interpretabili. (N.B. vedi traduzione dell' elenco messaggi. Le prossime versioni di PARSIC saranno in lingua italiana) .

Il menu **Window** contiene due funzioni utili per seguire la compilazione del sorgente assembler, mano a mano che si procede nello sviluppo dello schema elettrico.
I due tasti funzionali sono **Used Variable e Source**. Quando attive, alla sinistra dello schermo compaiono due documenti: il primo e' una tabella dove sono specificate le variabili utilizzate nel sorgente , il secondo e' il file sorgente dello schema elettrico in formato testo . Non tentare la modificare di questi documenti perché il programma non lo consente. Le immagini che seguono, mostrano come appaiono sullo schermo le funzioni **Used Variable e Source**, una volta selezionate:



used variable



Source

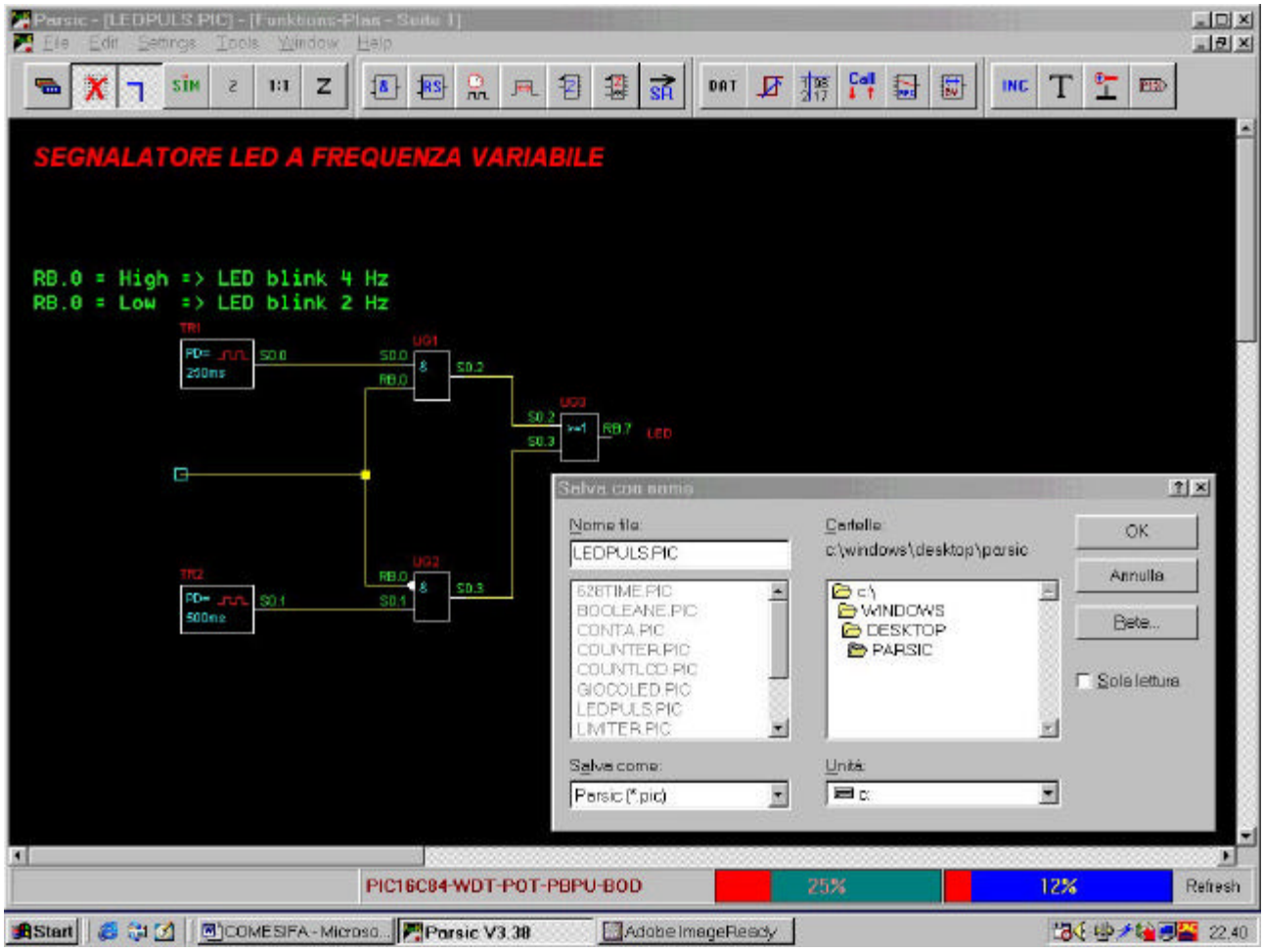
Come salvare il file sorgente.

Parsic produce tre tipi di file:

il primo con estensione **.PIC**	è il file oggetto del circuito progettato;
il secondo con estensione **. PJT**	e' il file di testo che si utilizzerà con **MPLAB**;
il terzo con estensione **.ASM**.	e' il file di testo che si utilizzerà con **MPASMWIN** .

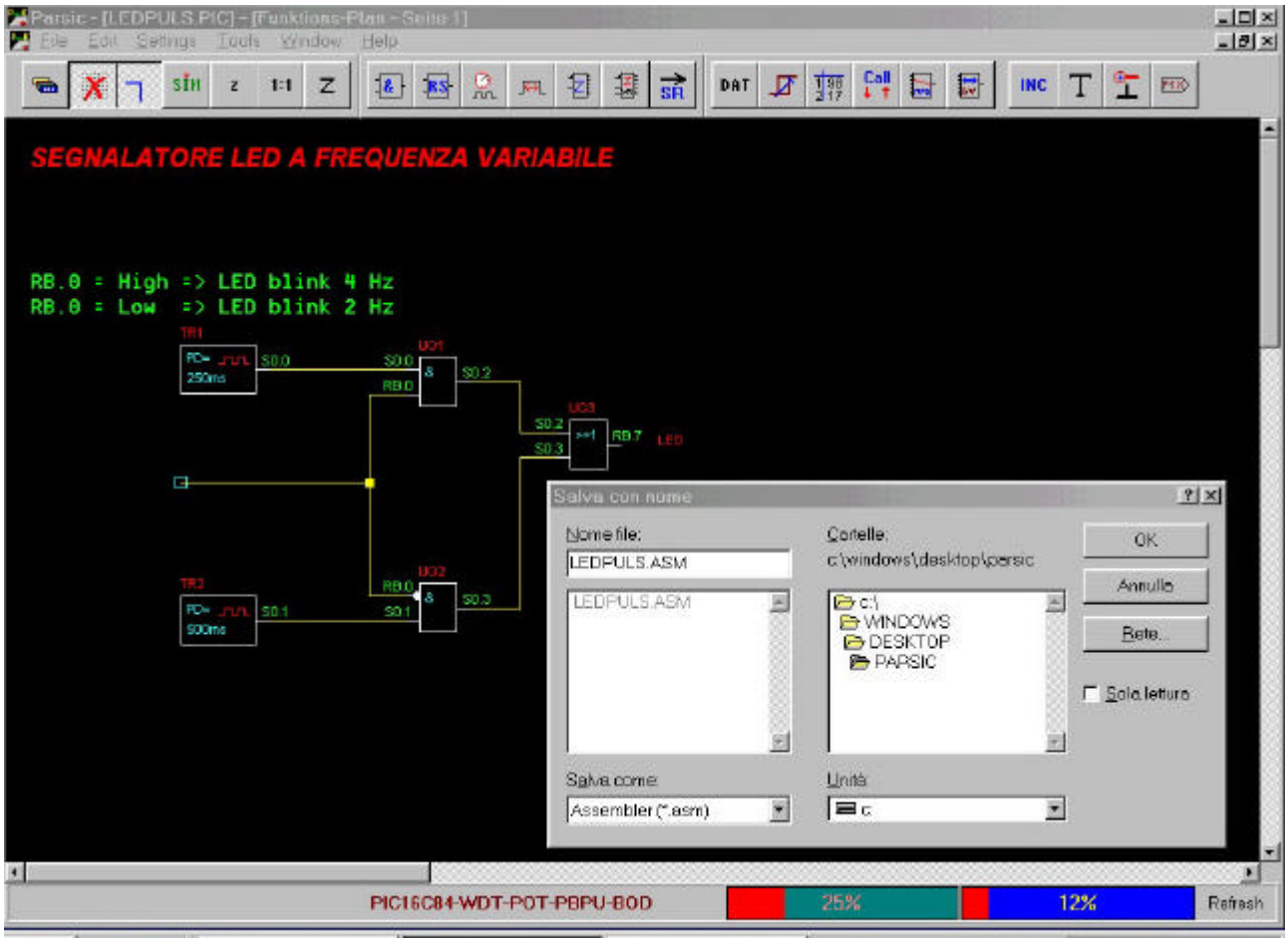
Gli esempi che seguono mostrano sono la sequenza pratica delle operazioni da compiere al termine della fase di progetto.

La procedura Save.pic



Portare il puntatore del mouse sul menu **File** e poi agire sul pulsante **SAVE (Save As...)** (diversamente, operare con la selezione breve **Cntr + S**.) Nominare il progetto con 8 lettere e l' estensione **.pic** . Dare **OK** per chiudere l'operazione.

La procedura Save. Asm:



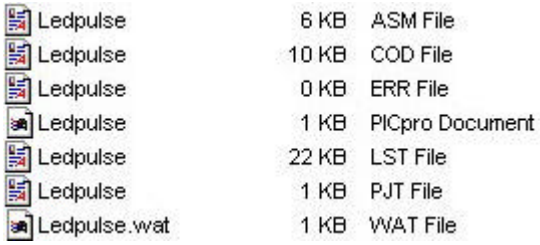
Per eseguire questa operazione, portare il puntatore del mouse sul menu **File** e poi selezionare il pulsante **SAVE SOURCE** (**Save Source as**..) Diversamente operare con la selezione breve **Cntr. + Q** . Nominare il progetto (lo stesso nome del precedente file) con estensione **.asm**.

Si può utilizzare, per questa procedura, una **selezione più rapida** con lo scopo di produrre, oltre al codice **ASM**,anche il codice **HEX** necessario alla programmazione del pic ed il codice **.pjt** per operare con **MPLAB**

Portarsi con il puntatore del mouse sulla barra delle applicazioni di window, selezionare **File** e poi agire sul pulsante **BUILD... (F10)**

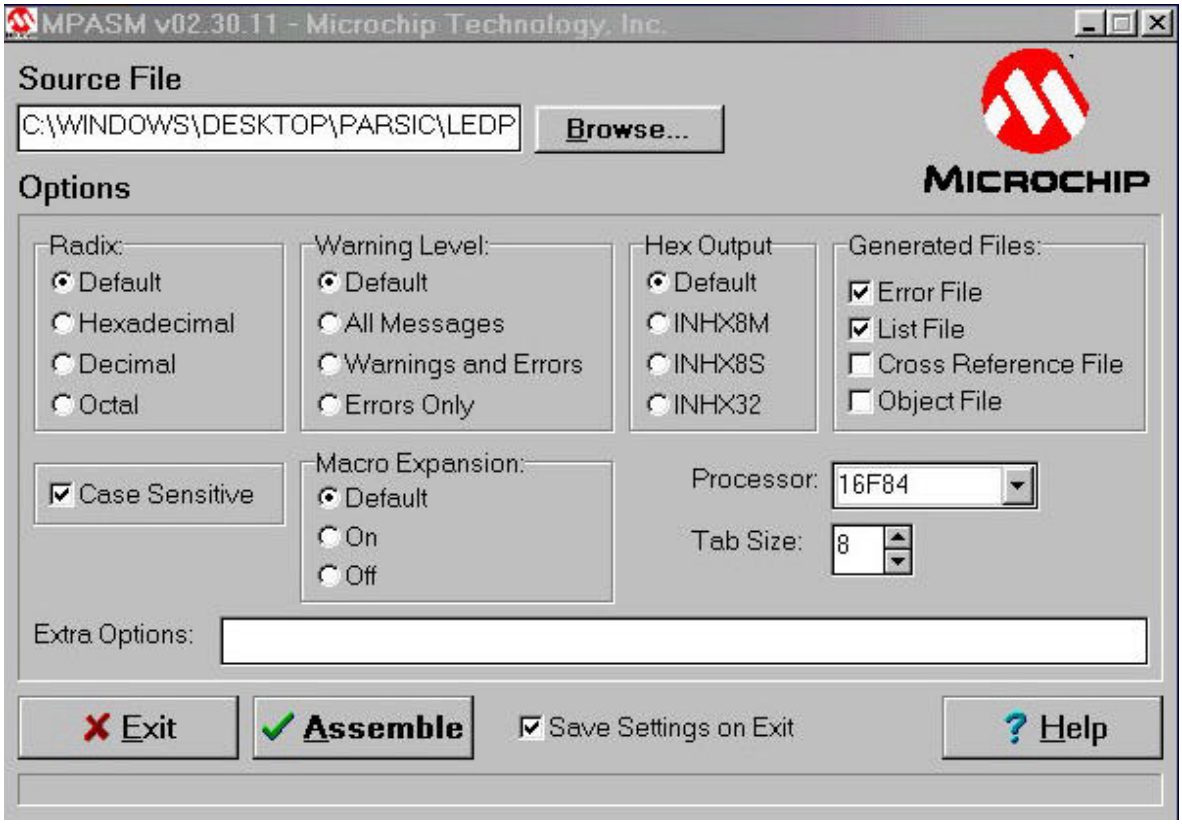
Si apre la stessa finestra di dialogo della figura sopra riportata. **nominare il file**, come spiegato in precedenza. Chiudendo l'operazione con **OK**, si vedra' attivare automaticamente l' **MPASM**. Tutti i file prodotti da **MPASM** saranno salvati nella **directory di Parsic** . Nella figura che segue, si possono osservare come sono stati prodotti i file del progetto **Ledpulse** e la maschera di presentazione di **MPASM**. **La funzione F10 rende automatica l'operazione.**

I file prodotti da Parsic ed MPASM



Questo e' l'elenco dei file prodotti durante il processo di conversione del file di testo Ledpulse.ASM
Se MPASM produce il codice di errore .ERR, il file. HEX (PICpro Document) non viene prodotto

La finestra di dialogo di MPASM



Per l'uso corretto di questo programma, seguire le istruzioni di Microchip.

Compilazione del codice sorgente.

I microcontrollori PIC non sono in grado di comprendere direttamente quello che l'utente scrive nel file di testo Assembler, ma necessita di istruzioni binarie o digitali in un particolare formato, detto linguaggio macchina. Prima che un programma Assembler possa funzionare su un microcontrollore PIC,deve essere tradotto dal formato sorgente al linguaggio macchina.Questa traduzione (che rappresenta il secondo stadio dello sviluppo del programma) è effettuato da un programma detto compilatore che, partendo dal sorgente, produce un nuovo file contenente le istruzioni in linguaggio macchina, che corrispondo ai comandi impartiti in Assembler.
Ogni compilatore richiede un comando particolare per la creazione del codice oggetto. Nel caso di Visual Parsic questa funzione è completamente automatica,. Se si utilizza un ambiente di sviluppo grafico,come Visual Parsic, effettuare la compilazione diventa estremamente facile. Al termine della compilazione,per mezzo dei comandi automatici (funzione F10), si ottiene la conversione dello schema elettrico di Parsic in una serie di file,ognuno con un'estensione diversa come ,ad esempio, il codice ASM generato in formato testo, il file (xxx.err) degli errori riscontrati nel file assembler, il file (file .hex) necessario alla programmazione del PIC , il file (.PJT) necessario per l'impiego dell'ICD debugger ,ecc. Visual Parsic permette una grande flessibilità di stesura degli schemi elettrici, dovuta dalla presenza, di una serie di blocchi funzioni,collegabili tra di loro in modo del tutto generale. Per formarsi un'idea complessiva del sistema di sviluppo Parsic ,tuttavia,conviene cominciare da uno degli esempi di configurazione circuitale già sviluppato.
Il codice sorgente Assembler, prodotto da Visual Parsic, è costituito da una serie di istruzioni e comandi impiegati per istruire il PIC in modo che esegua i compiti desiderati. Il primo passo del ciclo di sviluppo di una applicazione ,consiste nella costruzione di uno schema elettrico funzionale i cui " blocchi funzione" sono collegati tra loro a mezzo di linee di collegamento. Si opera per gerarchie funzionali dove,ogni sezione logica dello schema elettrico,svolge una compito specifico.
Si dovrà considerare come sviluppare uno schema elettrico, quali saranno le gerarchie circuitali da assegnare ad ogni insieme di blocchi funzionali, senza creare confusione nella stesura dei componenti sul foglio di lavoro,impiegando più possibile label terminali di linea,evitando quindi di intrecciare o sovrapporre le linee di collegamento tra loro.Si può sviluppare uno schema elettrico complesso,su più pagine circuitali,collegate tra loro a mezzo delle label terminali. L'impiego di targhe di commento, è strettamente necessaria. Molti programmatori principianti considerano i commento come un'inutile perdita di tempo.Il funzionamento di uno schema elettrico può essere molto chiaro se è specificato il funzionamento di un insieme di blocchi circuitali e,quando gli schemi diventano molto ampi e complicati, l'uso di target di identificazione dei blocchi è assolutamente consigliato. Specialmente se lo schema elettrico sarà riletto dopo molti mesi. Qualunque commento scritto non sarà considerato dal compilatore in fase di conversione.
Gli errori di compilazione dello schema elettrico sono comunicati per mezzo di messaggi che contengono una specifica dicitura oppure numero di codice. E' sempre possibile la stampa dello schema elettrico impiegando l'apposito comando.
I blocchi funzione hanno un nome. Il codice contenuto in ogni funzione (macro) è elaborato dopo che l'utilizzatore avrà collegato e/o specificato le terminazioni di collegamento. Un blocco funzione lasciato con i terminali non collegati o specificati erroneamente,in alcuni casi non permetterà la simulazione di funzionamento dello schema elettrico,facendo generare al compilatore un messaggio di errore.

Definizioni.

Variabili e costanti:

i programmi spesso lavorano con tipologie di dati differenti, e sovente devono archiviare i valori che gestiscono che possono essere sia numeri che caratteri. L'assembler prevede due modi di manipolare i valori,attraverso le variabili e attraverso le costanti. La variabile è una locazione di memoria impostata su un certo valore che può essere modificato durante l'esecuzione del programma. Al contrario una costante contiene un valore fisso che non deve essere modificato.
Nella sua architettura,il pic utilizza varie aree di memoria. La memoria RAM (register file)serve per archiviare le operazioni su cui lavora. La RAM è composta da circuiti di memoria che possono essere scritti e cancellati ogniqualvolta sia necessario. Il termine "volatile" riferito ad una memoria RAM, significa che il mantenimento dei dati termina con lo spegnimento del PIC. Ogni PIC dispone di un certo quantitativo di memoria RAM,la cui dimensione dipende dalle dimensioni del chip interno,quindi, dalla sigla del PIC scelto per l'applicazione . La memoria RAM del pic è organizzata in modo sequenziale ed ogni byte è immesso di seguito all'altro.. Gli indirizzi della memoria sono assegnati alle locazioni di memoria in ordine progressivo, cominciando dall'indirizzo 0. Date le sue caratteristiche la RAM del PIC (register file) è la memoria normalmente usata per memorizzare le variabili del programma,ovvero tutti quei valori il cui contenuto varia durante l'esecuzione.
Una variabile è una etichetta associata a una locazione di memoria. Utilizzare il nome della variabile equivale a riferirsi ai dati archiviati in tale posizione. Alcune locazioni della RAM (register file) sono usate per il controllo dell' hardware del PIC e non possono essere utilizzate per allocare i dati durante la fase di esecuzione del programma.Queste locazioni di memoria pertanto sono riservate per le funzioni speciali e non possono essere usate per altri scopi.
I nomi delle variabili,nel compilatore Parsic,sono assegnate automaticamente. L'utente può impiegare nomi diversi,specificando gli stessi con diciture alfanumeriche lunghe non oltre 31 caratteri.

Costanti . Analogamente alle variabili, le costanti rappresentano delle locazioni di memoria utilizzate dai programmi. A differenza delle variabili, però, i valori memorizzati nelle costanti non possono essere modificati durante l'esecuzione dei programmi .Una costante letterale è un valore digitato direttamente all'interno del codice sorgente nel punto in cui è richiesto. Normalmente queste costanti sono memorizzate in una particolare area di memoria dei PIC chiamata " Program Memory" ,che è una memoria speciale di tipo Flash, non cancellabile elettricamente e serve nei microPIC per tenere memorizzato il programma da eseguire.

Le costanti simboliche, rispetto a quelle letterali comportano dei vantaggi:

- ?? Possono essere richiamate ripetutamente per l'esecuzione di calcoli ;
- ?? Possono essere modificate,in fase di elaborazione, per ottenere una migliore precisione nell'esecuzione del calcolo.

Il modulo DAT di Visual Parsic può essere impiegato per definire una costante letterale.

Blocco funzionale:

per blocco funzionale ,dobbiamo intendere una funzione elementare o complessa, che fornisce una o più uscite digitali, in funzione di uno o più ingressi digitali o analogici e di eventuali parametri impostabili. Ogni blocco funzionale del programma può essere "specializzato" per svolgere funzioni più o meno complesse . E' il caso dei counter tipo ZV e ZR, dei multiplexer, dei moduli LCD, ecc.

Variabile Digitale:

una variabile logica binaria che assume i soli due valori 0 e 1 , corrispondenti ai valori logici Vero e Falso o viceversa. Le stato delle linee elettriche di collegamento virtuale,durante la fase di simulazione, assumono colori diversi in ragione dei valori digitali 0 e 1 prodotti dai moduli funzionali.

Variabile Analogica:

una variabile numerica che può assumere tutti i valori compresi tra 0% e 100%, con risoluzione pari a quella dell' AD converter del microprocessore utilizzato. I moduli DAT, nominati ADC, assumono il valore o 100% della linea analogica collegata all'ingresso del PIC.

La risoluzione interna delle funzioni logico-aritmetiche è impostabile a 8 oppure 16 bit.

Definizione dei blocchi circuitali di Visual Parsic :

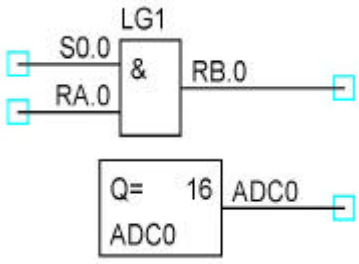
Identificativi di collegamento utilizzati da Parsic:

ad intervalli di tempo fisso, il sistema esegue una lettura di tutti gli ingressi/uscite digitali effettivamente collegati . Parsic assegna automaticamente un codice numerico ad ogni linea di collegamento. In particolari circostanze quali, ad esempio, nella definizione dei port di ingresso e di uscita dal microprocessore, l'operatore deve forzare l'identificazione delle linee in osservanza delle istruzioni riconosciute da Parsic. Le linee di collegamento elettrico virtuale, sono indicate con le seguenti etichette e la specifica delle stesse può essere automatica o manuale:

S. xx / Sx	specifica automatica: descrive il codice interno dei segnali assegnato dal programma. **La codifica Sx.x** e' orientata al bit, **Sx** al byte;	
Rx, GPx	specifica manuale	: l'operatore indica quale nome assegnare agli I/O del PIC;
ADCx	specifica manuale	: l'operatore descrive il nome assegnare agli input analogici del PIC.

Ogni collegamento e' distinto **SEMPRE** da una o più lettere seguita/e da uno o due numeri identificativi separati tra loro per mezzo di un punto : **S0.1 , S7.5, S8.2, SI.1 , UR.3, ADC0, GP4, RA.0, RC.5, ecc.**

Esempio di elencazione degli I/O :



Parsic riconosce automaticamente se un gate e' stato programmato come terminale di ingresso oppure di uscita. I terminali del processore che non sono impiegati, sono riconosciuti dal programma come port di ingresso. Nota : Microchip consiglia di vincolare a livello logico alto o basso i terminali non utilizzati del processore,collegando le terminazioni a resistenze di pull-up o pull-down.

Specifiche degli I/O dei pic.

Si consiglia sempre di leggere attentamente i data sheet dei pic utilizzati. Dalla lettura si può conoscere la posizione dei pin dei micro e la loro specifica.

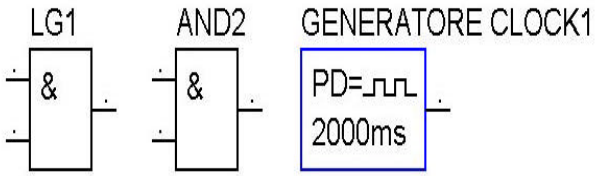
Corrispondenza degli ingressi digitali ad analogico:

RA.0 = ADC0 RA2 = ADC2
RA.0 = ADC1 RA.3 = ADC3 ecc.

Denominazione dei blocchi.

I blocchi logici circuitali sono nominati automaticamente dal software. E' possibile indicare con un nome diverso i blocchi circuitali, dando loro una denominazione diversa. Sono ammessi nomi con 30 caratteri, per ogni identificativo diverso dall'originale.

Esempio di denominazione dei blocchi circuitali:



Questo è l'elenco delle istruzioni assembler dei PIC. Queste istruzioni possono essere utilizzate nella scrittura dei file INCLUDE, impiegando l'apposito blocco funzionale di Parsic. Le stesse istruzioni non possono essere usate per nominare i blocchi funzionali.

ADDLW	SUBLW	MOVWF
ANDLW	XORLW	NOP
CALL	ADDWF	RLF
CLRWDT	ANDWF	RRF
GOTO	CLRF	SUBWF
IORLW	CLRW	SWAPF
MOVLW	COMF	XORWF
OPTION	DECF	BCF
RETFIE	DECFSZ	BSF
RETLW	INCF	BTFSC
RETURN	INCFSZ	BTFSS
SLEEP	IORWF	
TRIS	MOVF	

Operatori logici AND,NAND,OR, NOR, XOR, XNOR : 

Le porte logiche hanno due usi fondamentali : come elementi di decisione (o di logica) o come elementi di gating. I due diversi usi delle porte possono essere distinti nel seguente modo :

Porta (dispositivo logico) : circuito con due o più ingressi e un'uscita,la quale dipende da una combinazione logica dei segnali in ingresso. Le quattro porte base sono AND, OR,NAND e NOR.

Porta (dispositivo di gating) :circuito con due o più ingressi e un'uscita. Si può identificare un ingresso come ingresso dati, mentre gli altri sono ingressi di gating. Lo stato logico dell'ingresso di gating determina se il dato in ingresso deve apparire o meno in uscita.

Quindi, si può usare lo stesso dispositivo hardware per due diverse applicazioni. E' molto diffuso usare una porta per bloccare o trasmettere un dato. Il termine porta, in inglese gate, ha un certo numero di significati e quindi è meglio elencare tutte le possibili definizioni che se ne possono dare :

- ?? Circuito con due o più ingressi e un'uscita che dipende dalla combinazione degli stati logici in ingresso.Ci sono quattro porte base,dette AND,OR,NAND e NOR,ma la più usata è la porta AND.
- ?? Segnale usato per permettere il passaggio in un circuito di un altro segnale.Il segnale è detto, in molti casi, segnale di gating.
- ?? Uno degli elettrodi di un transistor FET.
- ?? Circuito in cui un segnale (di solito ad onda quadra o impulso) mette un altro segnale in on o off.

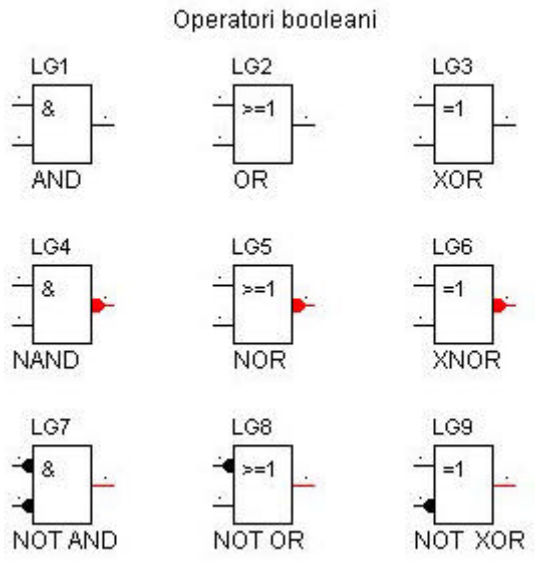
To gate . è un verbo e significa controllare il passaggio in un circuito digitale di un segnale digitale.
To enable e to strobe sono sinonimi. To enable significa permettere ad un circuito digitale di essere attivato mediante la sospensione di un segnale di inibizione ; to strobe significa attivare un circuito digitale.Precisiamo che le porte,funzioni elementari dell'elettronica digitale, possono essere usate in diversi modi, però indipendentemente da come una porta è usata, la tabella della verità fondamentale, per quella determinata porta, è sempre valida. Ne consegue che una tabella della verità è una caratteristica base di una specifica porta, e non è alterata in nessun caso dall'uso che si fa di quella porta.

Le tavole della verità delle funzioni logiche sono le seguenti :

	ingressi		uscite Sx.x	
OR-NOR :	a	b	or	nor
	0	0	0	1
	0	1	1	0
	1	0	1	0
	1	1	1	0
			and	nand
AND-NAND :	0	0	0	1
	0	1	0	1
	1	0	0	1
	1	1	1	0
			xor	xnor
XOR-XNOR	0	0	0	1
	0	1	1	0
	1	0	1	0
	1	1	0	1

La funzione NOT è liberamente collegabile ai gate di ingresso e/o uscita delle logiche,consentendo di realizzare funzioni complesse.

Esempio applicativo degli operatori booleani:



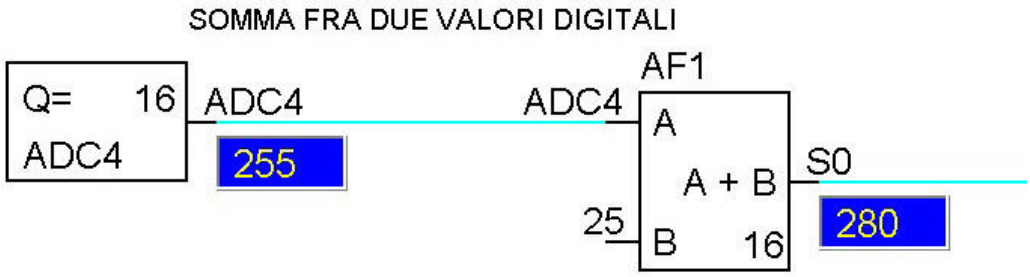
Blocco con operatore matematico.

Gli operatori matematici di Parsic effettuano le quattro operazioni aritmetiche. Questi operatori dovrebbero risultare abbastanza famigliare a tutti e non dovrebbero presentare alcun problema di utilizzo.I blocchi operazione sono quegli elementi che consentono di eseguire operazioni aritmetiche con risoluzione a 8 oppure16 bit. Il numero delle azioni è limitato dalla sola capacità di memoria dl Pic.
Si possono effettuare le operazioni aritmetiche di addizione,sottrazione,moltiplicazione e divisione.
Le operazioni possono riguardare diversi tipi di operandi quali costanti, variabili interne, valori associati a blocchi funzione e valori numerici. La struttura di una operazione aritmetica sarà costituita da un primo operando,da un secondo operando,da una destinazione. I limiti del valore assunto dagli operandi sono definiti, da 0 a 225, per le operazioni a 8 bit, da 0 a 65534 bit per le operazioni a 16 bit. Diversamente, per operazioni non congrue (superamento del limite) il sistema di calcolo andrà in overflow, determinando un errore e l'operazione non sarà significativa.

Blocco sommatore digitale :

Il blocco sommatore esegue la somma di due variabili e ne fornisce all' uscita il risultato :
S = a + b oppure S = k + a (k = costante)
La risoluzione può essere di 8 bit o 16 bit. Pertanto le variabili ammesse, nel primo caso, sono fino a 255 bit, nel secondo caso fino a 65535 bit.

Esempio di somma di una variabile analogica e di una costante digitale:

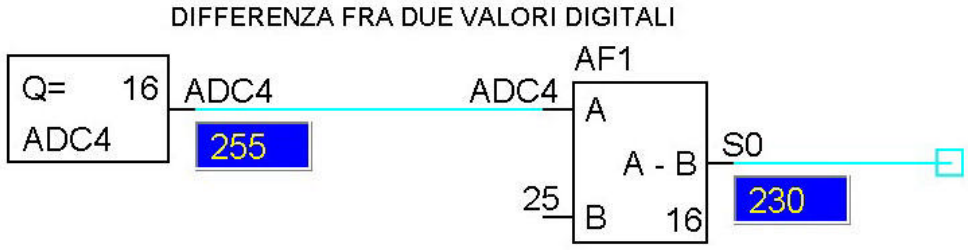


Blocco sottratore digitale :

Il blocco sottratore esegue la differenza di due variabili e ne fornisce all'uscita il risultato.
La funzione e' valida solo se a > b ,diversamente avremo un risultato non congruo :

S = 1 se a ? 1 b = 0
La risoluzione può essere di 8 o 16 bit.

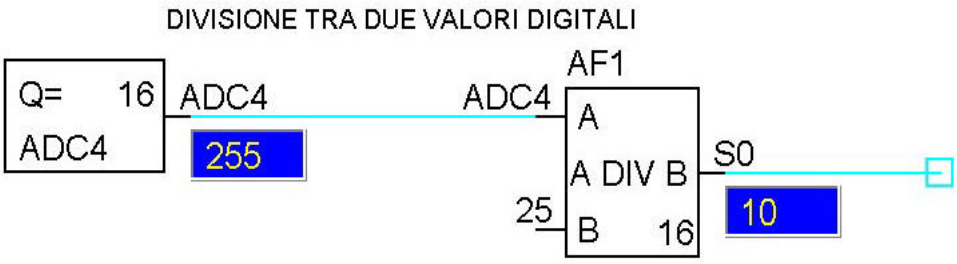
Esempio di differenza tra una variabile analogica ed una costante :

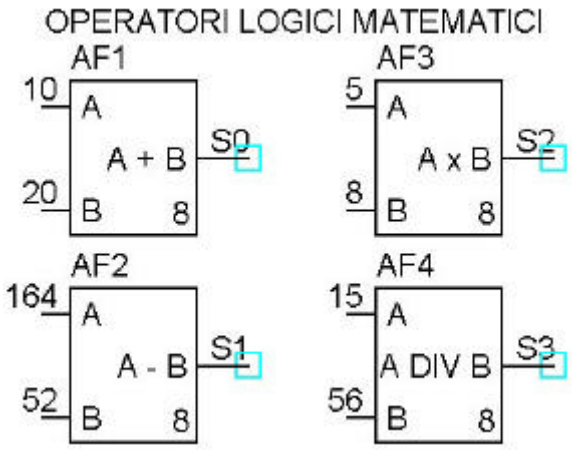


Blocco divisore digitale:

Esegue la divisione tra due variabili collegate ai suoi ingressi. L'operazione e' valida se b>0
Diversamente avremo :
S = 255 se a ? 0 b = 0
S = 0 se b > a
La risoluzione può essere di 8 o 16 bit.

Esempio :





Blocco moltiplicatore:
Esegue il prodotto di due variabili collegate ai suoi ingressi. Il risultato e' dato da S = a ? b
La risoluzione può essere 8 o 16 bit.

Esempio di moltiplicazione tra due valori digitali :

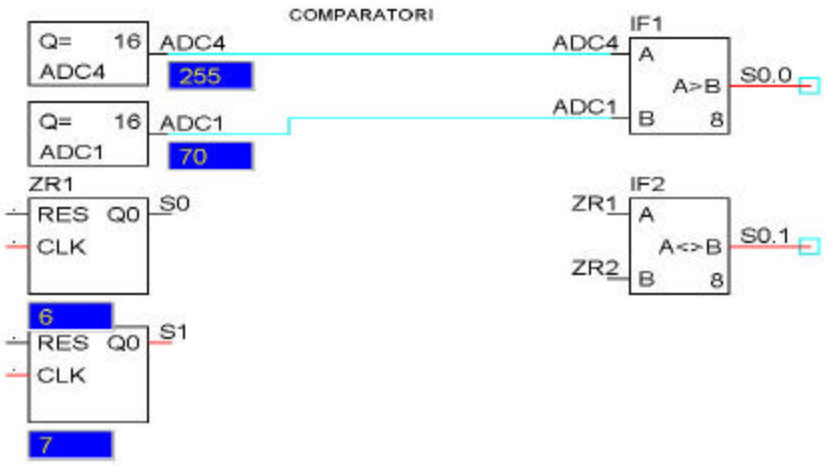
Operatori relazionali : comparatori.

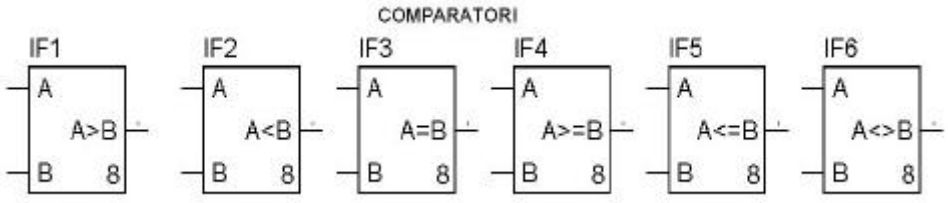
Gli operatori relazionali di Parsic, sono utilizzati per confrontare variabili e costanti e rispondere a domande come x (variabile) è più grande di 100 (costante) oppure y(variabile) è uguale a 30 (costante). Un'espressione contenente un operatore relazionale è valutata secondo un valore di verità,cioè vero (true) o falso (false).I sei operatori relazionali di Parsic sono elencati in tabella.
Il blocco relazionale permette il confronto tra due operandi, e attiva la sua uscita quando la comparazione è vera.

S = 1 se	a > b
S = 1 se	a < b
S = 1 se	a ? b
S = 1 se	a ? b
S = 1 se	a = b
S = 1 se	a<>b

Il primo e secondo operando sono costituiti da una qualsiasi valore immediato,proveniente da un blocco funzionale o da una costante numerica.

Esempio di collegamento di blocchi relazionali :



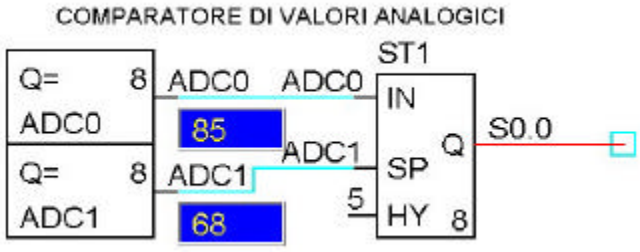


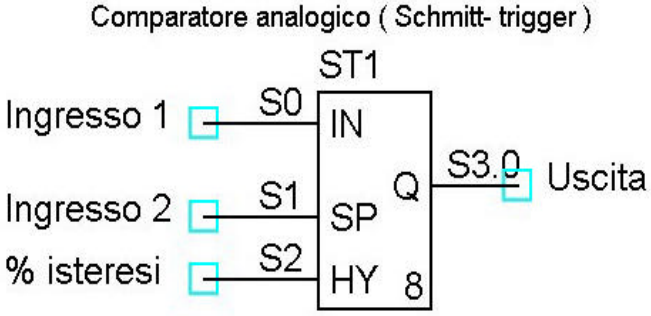
Comparatore ingresso analogico (Schmitt - trigger) :

Confronta due variabili e fornisce in uscita un livello logico **Vero o Falso** . Inoltre è prevista un'isteresi parametrizzabile che evita instabilità al circuito.Il valore dell'isteresi è sottratta all'ingresso IN, in base allo stato del comparatore prima di fare il confronto, comportandosi in tal modo come un'area di banda morta.Il trigger di Schmitt viene utilizzato principalmente per portare l'uscita a livello logico 1 quando sull'ingresso la variabile applicata supera il valore di soglia che è stata impostata da una costante .La differenza tra il valore della variabile in ingresso e il valore della costante si chiama isteresi. Il vantaggio che presenta l'uso del trigger di Schmitt è quello che l'uscita possa commutare in modo desiderato in presenza di variabili ben definite e che la stessa uscita possa ritornare allo stato iniziale dopo che la variabile è stabilizzata su valori al disotto della soglia di intervento. Per una corretta definizione dei valori di intervento dei comparatori è da tenere presente che il valore **HY** si sottrae a uno dei due ingressi. Pertanto, per cambiare di stato, l'ingresso **IN** deve modificarsi in modo da differire dall'altro ingresso, meno il valore dell'isteresi:

IN = variabile ingresso analogico (- HY)
SP = k (costante) analogica o digitale
HY = isteresi
Q = uscita blocco comparatore

Esempio di un possibile collegamento del blocco comparatore analogico (Schmitt- trigger)





Modulo DAT.
Variabili , Costanti e ADC

Nel precedente capitolo sono stati definiti i termini tecnici con i quali si definiscono le variabili. Il modulo **DAT** è un componente fondamentale nell'ambiente parsic ,perché ad esso possiamo attribuire più funzioni :

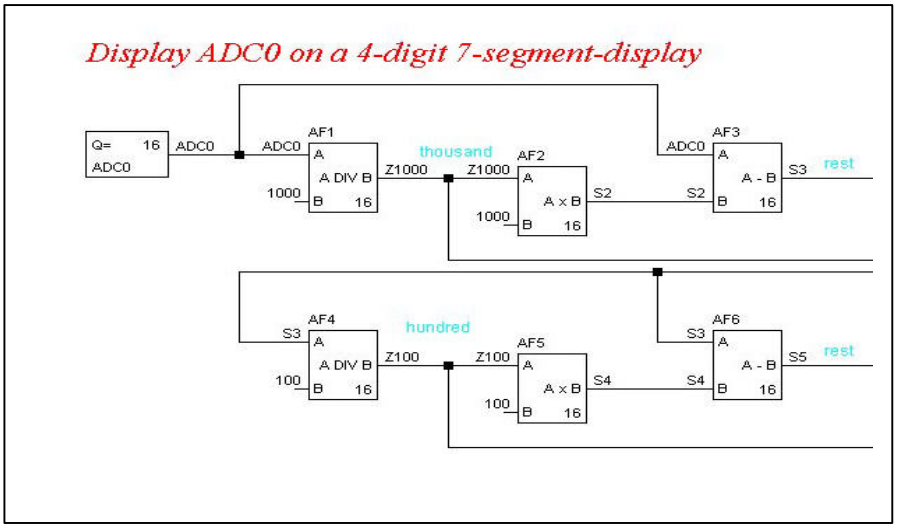
- ?? **è impiegato come ADC in associazione agli ingressi del PIC (ADC0,ADC1,ecc)**
- ?? **è impiegato come etichetta associabile ad una determinata variabile**
- ?? **è impiegato come etichetta associabile ad una determinata costante**

Gli **ingressi analogici** dei PIC sono costituiti da input il cui valore è acquisito in successione. Appena il valore analogico di un input è convertito dal blocco **A/D** , il valore numerico corrispondente è aggiornato nell'apposito registro di memoria, la cui variabile è associata al nome del modulo DAT che, per l'occasione, è nominato ADCx.
La risoluzione degli **ADC** nei PIC varia secondo il modello; per quelli con **10 bit** di risoluzione, la codifica del valore convertito va da **0 a 1023** bit. Nei PIC con risoluzione a **8 bit** la risoluzione dell'ADC va da **0 a 254** bit.Il valore analogico che si misura, sia esso espresso in gradi di temperatura,in corrente,in tensione,in qualsiasi altra grandezza che varia analogicamente, può essere trasformato in un valore decimale che è quello che si deve manipolare all'interno del programma .

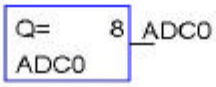
Possiamo destinare il funzionamento del modulo **DAT**, ad una o più variabili che saranno associate ad un file **Include,** scritto per la gestione di una funzione non implementata in Parsic. Ad esempio desideriamo impiegare un chip convertitore di temperatura tipo **DS1621**, collegato al bus **I2C**. Questo chip non è gestibile direttamente dal programma, pertanto sarà scritto un segmento di file **Assembler**, impiegando la **direttiva INCLUDE**. Detto file necessita di tre **soubroutine** collocate in tre sezioni distinte del listato **Assembler di Parsic**. Le tre soubroutine saranno nominate : D1621SUB.inc. D1621INI.inc DSMAIN.inc.
L'elaborazione restituisce 4 variabili denominate **IISADR , IIDAT, TEMP, TMPR** che sono rispettivamente : l'indirizzo del DS1621, **l'MSB e LSB** del valore della temperatura elaborato dal chip, ed il valore del **loop counter**. Queste quattro variabili saranno associate, **con lo stesso nome**, ai rispettivi moduli **DAT**, e resi disponibili per il completamento dello schema elettrico.

Nell'altro caso che esamineremo, il modulo **DAT** è impiegato in uno schema che prevede l'uso della **comunicazione seriale** tra PIC e PC. Il PIC è collegato in una scheda slave periferica. I dati provenienti dalla linea seriale, saranno collocati nelle seguenti variabili : **ADRx , VAR1, VAR2**. **ADRx** è l'indirizzo della periferica PIC, **VAR1 e VAR2** sono i nomi delle variabili aggiornate per mezzo della comunicazione seriale. ADRx è un valore costante ricavato,ad esempio dalla somma di due costanti in ingresso ad un modulo operatore matematico ed è un valore compreso tra **0 e 254**. VAR1 e VAR2 è una variabile il cui valore è compreso tra **0 e 65534**. Bisognerà predisporre i moduli DAT con risoluzione a 8 e 16 bit.

Esempio applicativo :



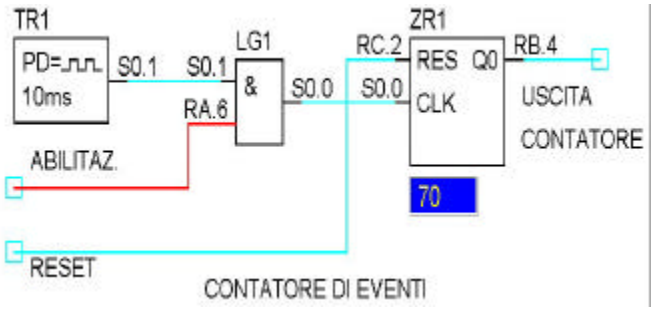
SELEZIONATORE INGRESSI ANALOGICI O COSTANTE ANALOGICA



Conteggio di eventi:

Il blocco contatore permette il conteggio di eventi. Il funzionamento è semplicemente quello di un contatore che incrementa di un passo a ogni impulso fornito sull'ingresso CLK. Il conteggio parte dal valore zero e termina al massimo valore ammesso,in dipendenza della risoluzione del contatore.(nei contatori a 8 bit la risoluzione massima è di 254 bit). Per resettare il blocco contatore occorre un impulso fornito sull'ingresso di reset.

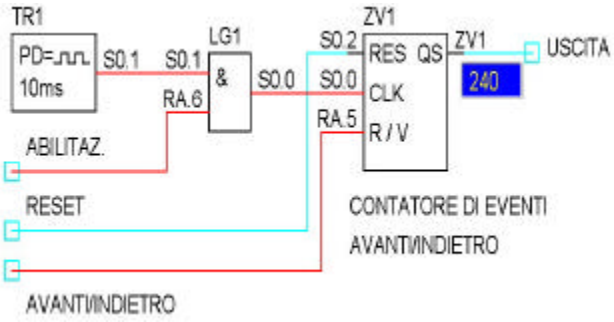
Esempio applicativo del blocco contatore 8 bit :



Blocco conteggio di eventi avanti/indietro:

Esempio applicativo del blocco contatore avanti - indietro :

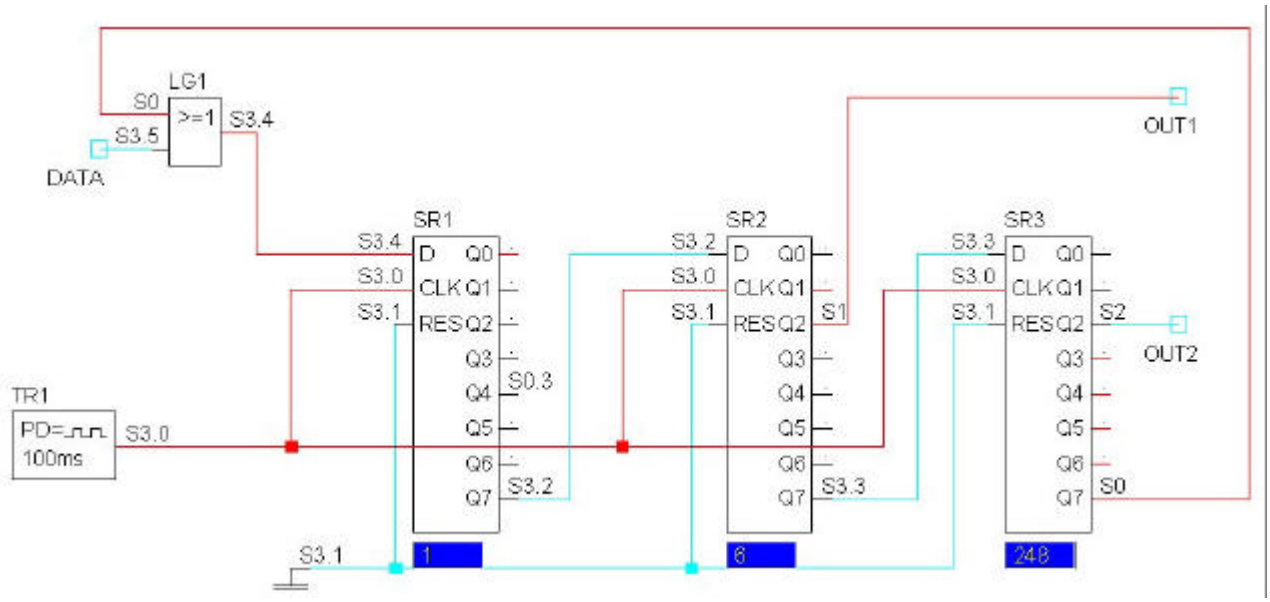
Il contatore reversibile ha lo stesso funzionamento del contatore visto in precedenza,con la sola differenza che il suo valore corrente può essere decrementato e incrementato,tramite gli appositi ingressi **UP e DOWN**. Il conteggio parte dal valore 0 ed evolve incrementando o decrementando,a seconda dell'ingresso che gli fornisce un segnale. L'escursione del contatore può essere selezionata con risoluzione a 8 oppure 16 bit. Nei sistemi di conteggio a preselezione,di tipo **UP/DOWN**, il contatore reversibile non può funzionare da solo e bisogna associare ad esso il modulo **LIMITER** .

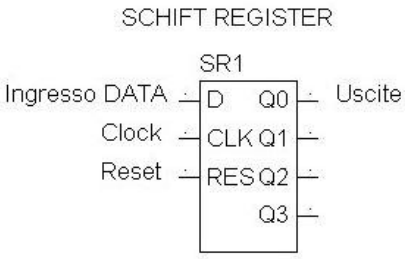


Blocco schift register :

Il blocco attiva lo scorrimento di dati all'interno di uno o più canali,con perdita dei bit che fuoriescono dal canale in seguito allo scorrimento dei dati. Lo **shit register** consente lo scorrimento dei dati all'interno del modulo designato. I dati entrano da sinistra e d escono da destra. I bit fuoriusciti vengono persi. Il modulo è controllato da tre ingressi,il primo è l'ingresso dati,il secondo è l'ingresso clock,il terzo è l'ingresso di reset.Ogni bit in uscita può essere impiegato per l'attivazione dei dispositivi esterni. Questo modulo trova piena applicazione nei circuiti di azionamento sincronizzato quali,ad esempio,nastri trasportatori con attivazione di bracci o pistoni meccanizzati,elettrovalvole,ecc.

Esempio applicativo:





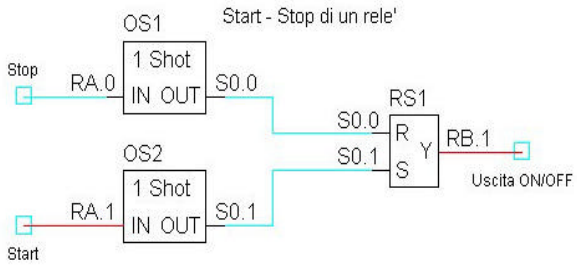
Flip-Flop set-reset :

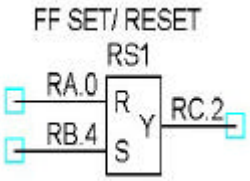
Un elemento di memoria è in genere un qualsiasi dispositivo in grado di conservare lo stato logico 1 oppure 0 di un bit,in modo tale che un singolo bit od un gruppo di bit possa essere indirizzato e successivamente riletto. La forma più comune in elettronica digitale è il bistabile, più comunemente conosciuto come **flip-flop**. Si tratta di un circuito la cui uscita possiede due stati stabili e che può passare da uno stato all'altro a seconda dei segnali di ingresso,ma che rimane stabile se non è presente in ingresso alcun segnale.Ciò differenzia l'elemento bistabile da un gate avente anch'esso due stati di uscita ma che richiede la permanenza di un segnale in ingresso per rimanere in un determinato stato.La caratteristica di possedere due stati stabili lo differenzia anche da un elemento monostabile,che invece ritorna in uno stato specifico,ed da un elemento astabile, che cambia continuamente da uno stato all'altro.In altre parole,il flip-flop è un termine che include qualunque tipo di elemento bistabile,o dispositivo a due stati,includendo :

?? **Flip-flop tipo D,JK,RS,RST,T**

Il **flip-flop** gestito da Visual Parsic è di tipo **RS**. In pratica è un circuito costituito da due NAND collegati in modo incrociato e che possiede due ingressi chiamati R e S. Il cambiamento di stato del FF si ottiene portando a livello logico 1 ora il terminale di S ora il terminale R.

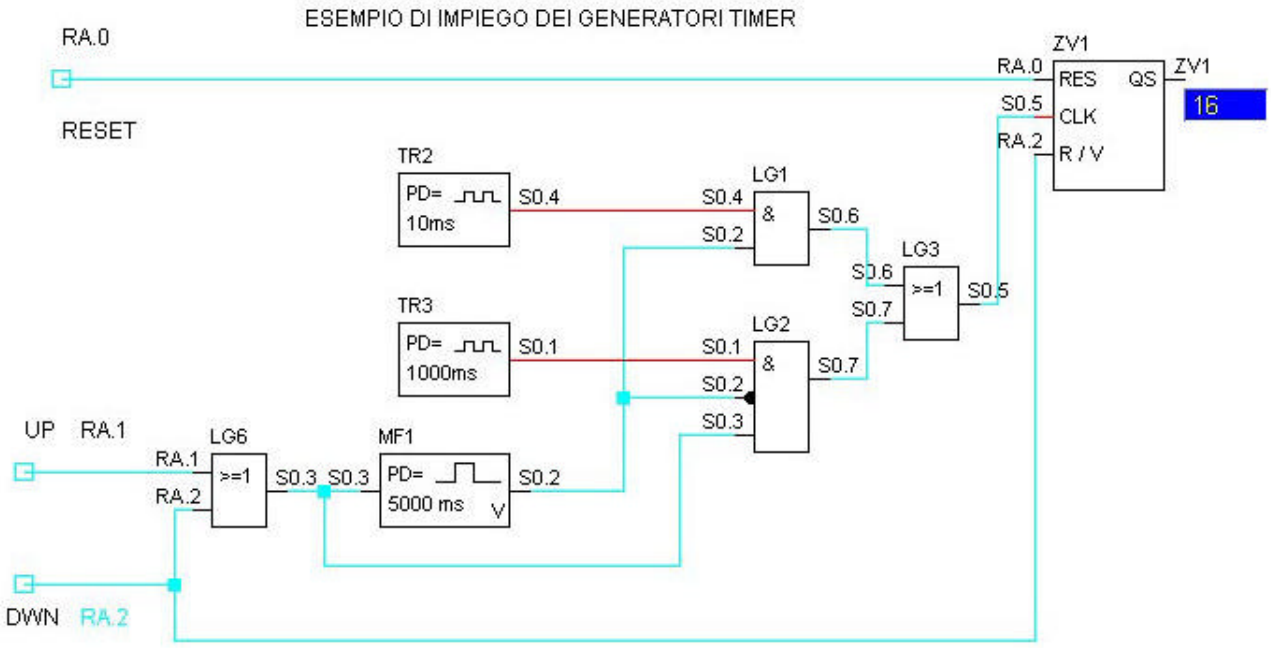
Esempio applicativo:

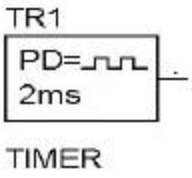




Timer o generatore base di tempi :

un generatore di impulsi controlla le temporizzazioni dei circuiti di un calcolatore rendendo regolare la velocità di elaborazione dei segnali. Esso serve a rendere sincrone tutte le operazioni del sistema digitale progettato.





Il modulo genera un segnale astabile il cui ciclo è determinato da un valore numerico impostato da tabella. Si può modificare tale valore facendo assumere al generatore il valore numerico prodotto da un blocco funzionale dello schema elettrico (contatore, sommatore, divisore, tabella, ecc) :

1- richiamare il modulo funzionale del generatore di clock;
2- assumere il valore di clock uguale a quello del blocco funzionale associato, ad esempio durata del periodo = ZR1.

In questo caso, il modulo genera una frequenza il cui periodo dipende da ZR1

Blocco circuito monostabile, su evento:

Il blocco monostabile consente di elaborare un impulso di durata regolabile,le cui caratteristiche sono configurabili selezionando la tabella del modulo. Si possono configurare monostabili su evento,retriggerabile,ad azione ritardata.

Una variabile digitale applicata all'ingresso del **monostabile** determina, per un periodo limitato di tempo, la variazione dell'uscita, **da livello basso a quello alto**. Le funzioni svolte da questo blocco sono le seguenti :

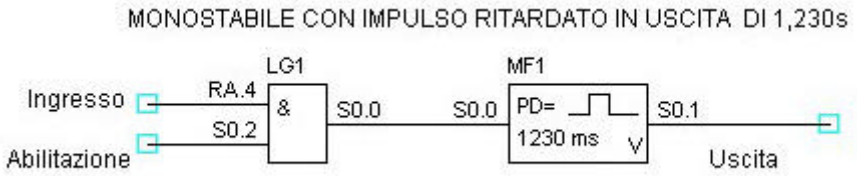
monostabile,su evento, retriggerabile:

l'applicazione di un impulso digitale in ingresso,di breve durata (one shot), determina lo stato di **cambiamento** logico dell'uscita **da livello logico basso a livello logico alto**. Tale livello logico **resta attivo** per tutta la durata della temporizzazione impostabile da **1 a 65535 ms**. Attivando il rettrigger, l'uscita rimane a livello logico alto, tra il ripetersi di un impulso e l' altro, nel dominio di tempo di durata dell'impulso stabilito .

Esempio di monostabile,retriggerabile:

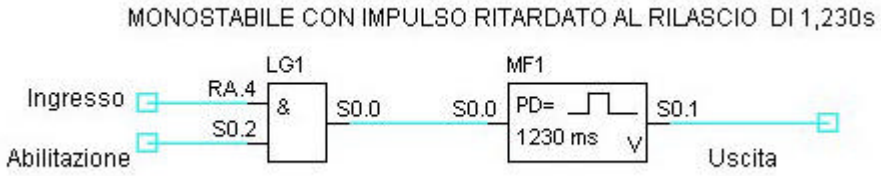
monostabile ,su evento,con attivazione ritardata

l'applicazione dell' impulso digitale in ingresso, determina lo stato di cambiamento digitale dell'uscita **da livello logico basso a livello logico alto**, solo **dopo un certo tempo**, determinato dall' impostazione della soglia di intervento che va' da **1 a 65535 ms**. **L'impulso** di ingresso **deve essere mantenuto** fino al cambiamento di stato dell'uscita. Togliendo l'impulso di ingresso l'uscita del monostabile torna a livello logico zero.



monostabile ,su evento,con attivazione slope

l'applicazione dell'impulso digitale in ingresso, di breve durata (one shot) , determina lo stato di cambiamento digitale dell'uscita **da livello basso a livello alto**. Tale livello **resta attivo** per tutta la durata della temporizzazione impostabile da **1 a 65535 ms**.



Monostabile di tipo one-shot :

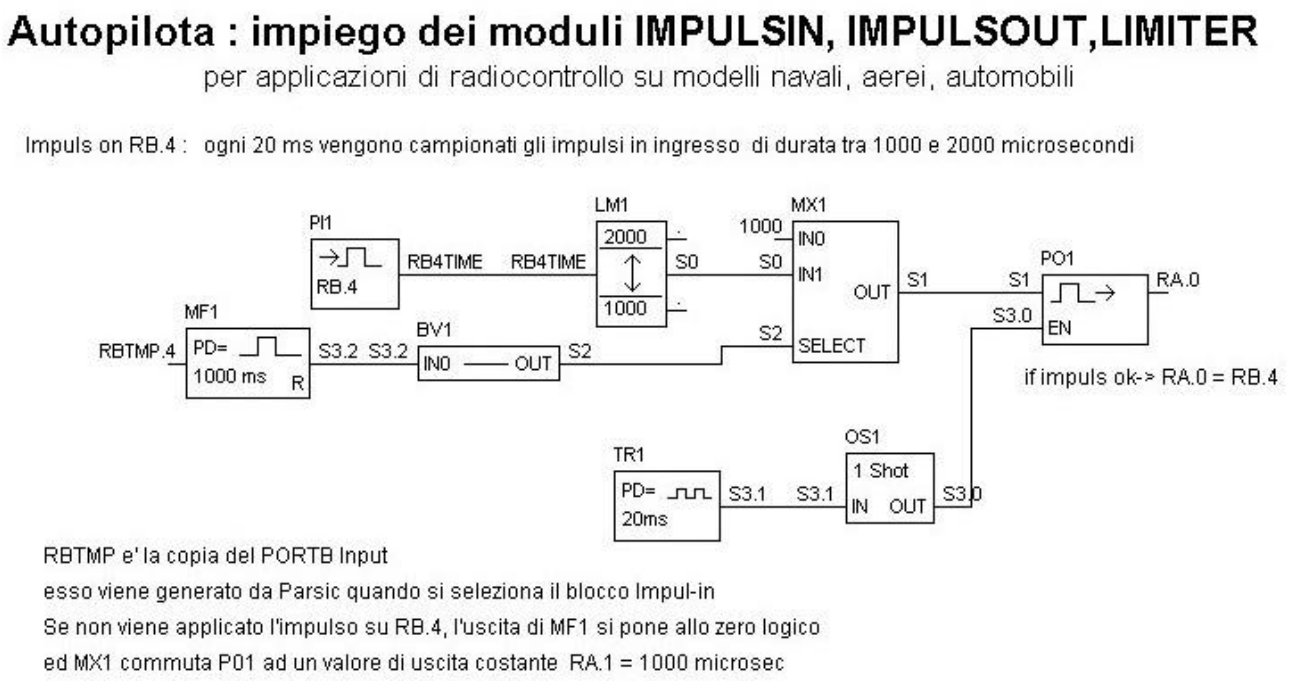
la funzione one shot, (un colpo), si ritrova in molti circuiti di automazione. Sono circuiti utili quando si desidera avere la certezza che il comando di attuazione sia uno ed uno solo. Classico l'impiego nei circuiti di conteggio comandati da semplici contatti a pulsante oppure in circuiti che fanno uso di fotocellule,ecc. La caratteristica del one-shot è quella di non produrre impulsi finché il contatto d'ingresso è attivato a 1 logico. Per ripetere l'impulso d'uscita il contatto dovrà portarsi prima allo 0 logico,poi ancora a livello alto.
Una variabile digitale, applicata all'ingresso del monostabile, provoca il **cambiamento** di stato **dell'uscita, per un breve periodo di tempo**, da livello logico basso a livello logico alto.
L'impulso di uscita e' **transitorio** , anche se l'ingresso continua ad essere mantenuto a livello alto.

MONOSTABILE TIPO ONE SHOT

La durata ON dell'impulso d'uscita del blocco monostabile, dipende dalla lunghezza del listato assembler. Più righe saranno contenute nel listato, più lungo sarà il tempo ON dell'impulso d'uscita.

Blocco condizionatore di impulsi, impulse-in. Interrupt

Questo modulo permette di utilizzare gli ingressi interrupt dei micro,favorendo la priorità di esecuzione di alcuni comandi digitali,rispetto ad altri.
Impiegando un quarzo da 4 MHz la risoluzione del periodo di intervento dell'interrupt è di 1 µs mente a 20 MHz è di ? 20 nanosecondi . L'ingresso del blocco condizionatore permette la rapida misura degli impulsi di ingresso compresi tra 50 e 65550 imp/sec. Il tempo di durata dell'impulso e' calcolato sul fronte positivo di salita dell'impulso .Sul fronte di discesa dell'impulso ha termine la misura che è memorizzata in una variabile denominata **RBxTIME ed RBxTIME_HI**.
Portando il cursore del mouse sul blocco impulsin potrete selezionare, nella finestra di dialogo, se la misura deve essere attuata sul fronte di salita o di discesa dell'impulso. Inoltre sara' possibile selezionare quale degli ingressi interrupt **PortB,** impiegare nel circuito.
Parsic genera il codice **RBTMP.x** , che e' la copia dell'impulso applicato al **PortB** impiegato. Questo codice è utilizzato per attivare i circuiti di campionamento e conversione del segnale all'interno del circuito in progetto. Nell'esempio che segue, è mostrato un circuito applicativo dove, oltre al condizionatore di impulsi (siglato PI1) ,sono utilizzati anche i blocchi logici limiter (LM1) ed impuls-out (P01) .



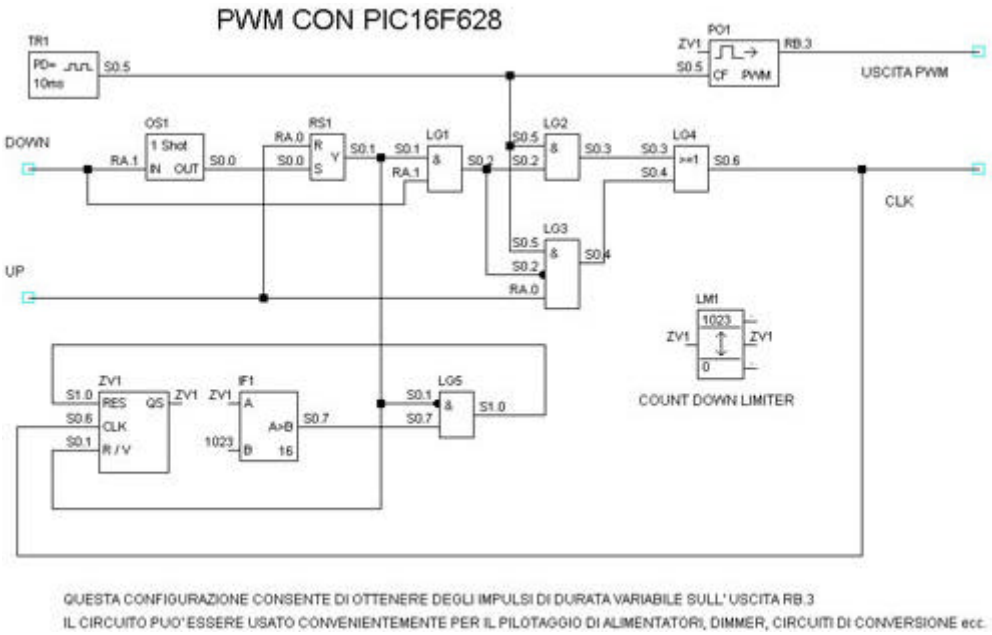
Blocco condizionatore di impulsi, impuls-out, generatore PWM e PFM.

Il blocco funzionale assume il doppio funzionamento di generatore di frequenza e generatore PWM. Quando è predisposto come generatore di frequenza,all'uscita del modulo sarà presente una frequenza di il cui valore sarà determinato dalla variabile,compresa tra 50 e 65550, collegata dall'ingresso del modulo. L'ingresso EN abilita l'uscita del modulo se è applicato al suo terminale un livello logico alto.
La risoluzione è di 1 µs se il quarzo impiegato è di 4 MHz, di 20 ns se il quarzo impiegato è di 20 MHz.

Generatore di impulsi PWM.

Questa funzione non e' implementabile su tutti i modelli di Pic.
Utilizzando il modulo come generatore **PWM** si ottiene al terminale di uscita, una frequenza il cui duty-cycle dipende dal valore numerico della variabile digitale applicata al terminale di ingresso. Tale valore deve essere compreso tra 0 e 1023 bit.
La frequenza di uscita dell'impulso e' costante **4KHz** con clock di 4MHz (20 KHz a 20 MHz).
Il **Duty-Cycle- Register** è aggiornato in base all'attivazione dell'ingresso Enable del modulo .
Con opportune modifiche circuitali esterne, ed adoperando un amplificatore operazionale, l'uscita PWM può essere convertita in DAC (Digital Analog Converter).

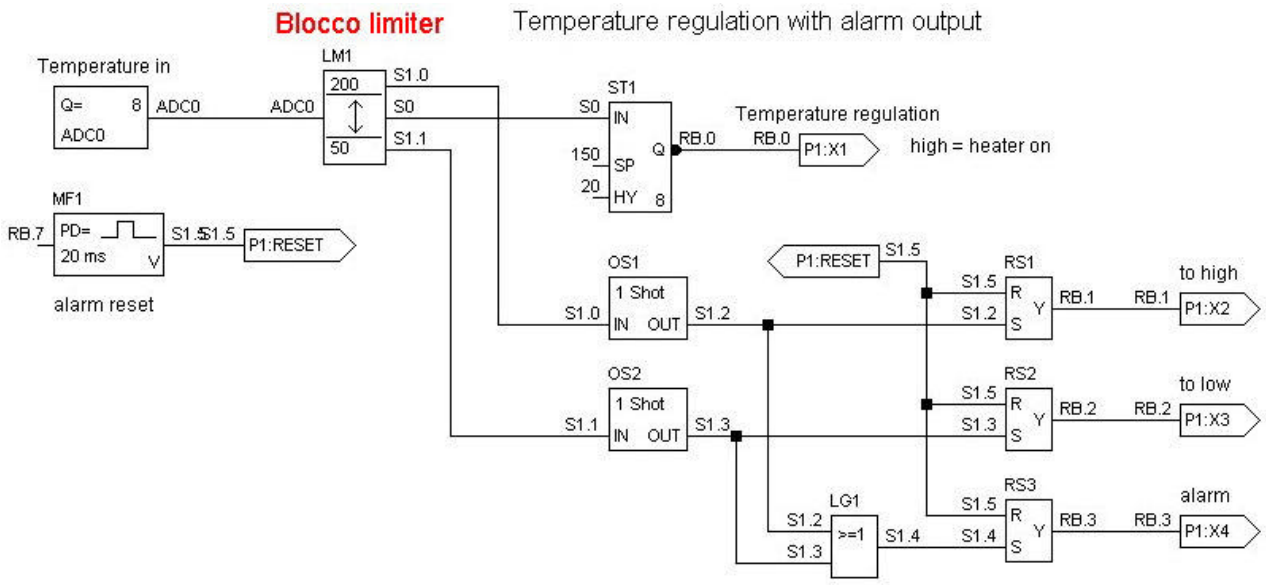
Esempio applicativo del generatore PWM

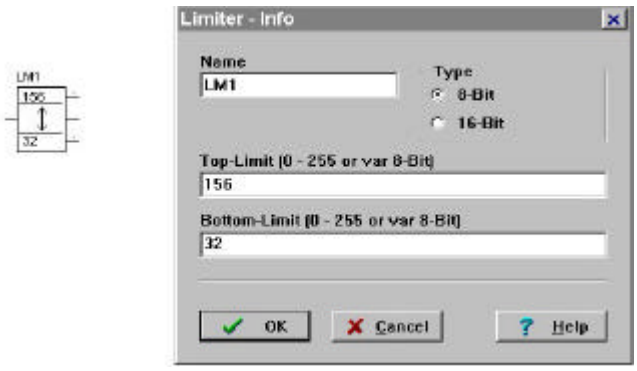


Limiter

Questo blocco funzionale controlla il **contenuto di una variabile** e stabilisce **due livelli di allarme** corrispondenti al valore **limite superiore ed inferiore** della variabile stessa.
Se il contenuto della variabile di ingresso e' maggiore del valore di set impostato (top limit) ,l'uscita corrispondente si portera' a livello logico alto.
Se il contenuto della variabile di ingresso e' inferiore al valore di set impostato (bottom limit) , l'uscita corrispondente si portera' a livello logico alto.
Il blocco funzionale limiter si comporta similmente ad un **filtro passa banda**: il valore numerico della variabile digitale, in uscita dal blocco, è contenuto all'interno dei valori di set (top e bottom limit) impostati, i valori inferiori e superiori al set saranno tagliati e non saranno considerati nell'elaborazione della variabile.
L'uso delle uscite digitali di allarme sono del tutto opzionali. Se esse non sono impiegate Parsic non le utilizzerà e non saranno considerate nel file sorgente.
E' possibile stabilire il funzionamento del blocco con definizione a **8** oppure a **16 bit**.

Esempio di impiego del blocco limiter:



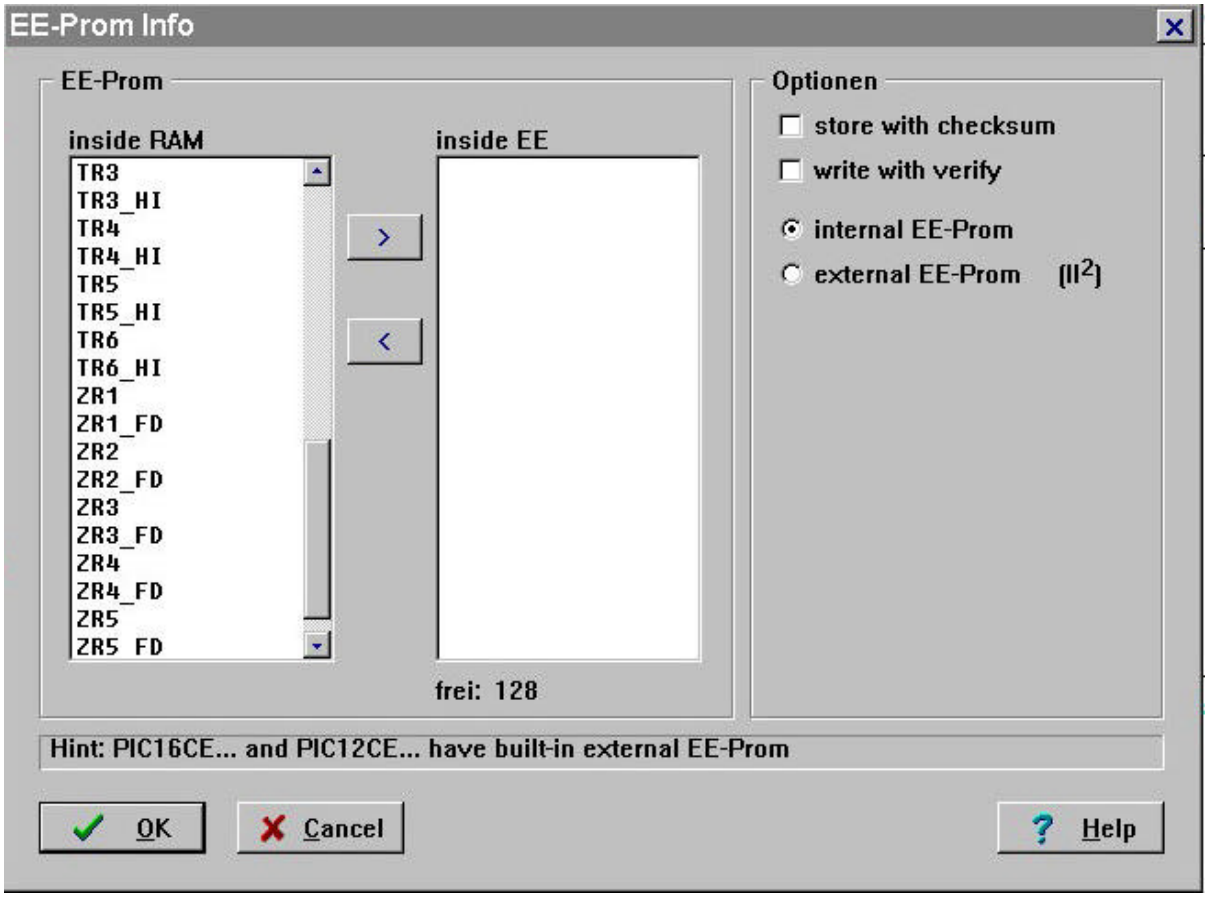


Lo stesso blocco funzionale è impiegato per altri scopi ,quale ad esempio la limitazione di escursione di un counter UP/DOWN : vedi esempi applicativi

EE

Gestione EEPROM

Una memoria dati è una memoria a semiconduttori dalla quale i dati possono essere ripetutamente letti o scritti. La **EEPROM** e' una particolare area di memoria e può essere ripetutamente programmata e quindi cancellata mediante l'uso di speciali tecniche di programmazione. Le modalità di accesso alla memoria **EEPROM** seguono speciali procedure onde evitare perdite di dati in condizione di funzionamento anomalo. Con il blocco **EEPROM (EE)** si possono memorizzare i contenuti dei dati sia nell'area di memoria EE integrata nel Pic, sia in quella esterna. I dati contenuti nella memoria **EE** sono richiamati ad ogni reset del mcp. Affinchè i dati possano essere scritti nella memoria **EE,** è necessario impiegare la tabella **EE-PROM INFO**. Questa tabella si attiva con il tasto destro del mouse, quando il puntatore si trova posizionato sul blocco funzionale **EExx.** Essa permette di spostare i dati dall'area **RAM** del pic a quella **EEPROM,** con semplici operazioni di tipo tabellare. La figura seguente, mostra la tavola EE-Prom-info che appare sullo schermo quando è attivata:



Nel riquadro di sinistra della tabella sono elencati le variabili dei blocchi logici facenti parte del circuito in progetto. Queste variabili, saranno codificati in una sequenza corretta di codici operativi (opcode) e memorizzati all'interno della memoria del pic. Tale memoria di programma è composta, secondo il tipo di microprocessore, da un certo numero di locazioni, ognuna delle quali contiene un opcode .

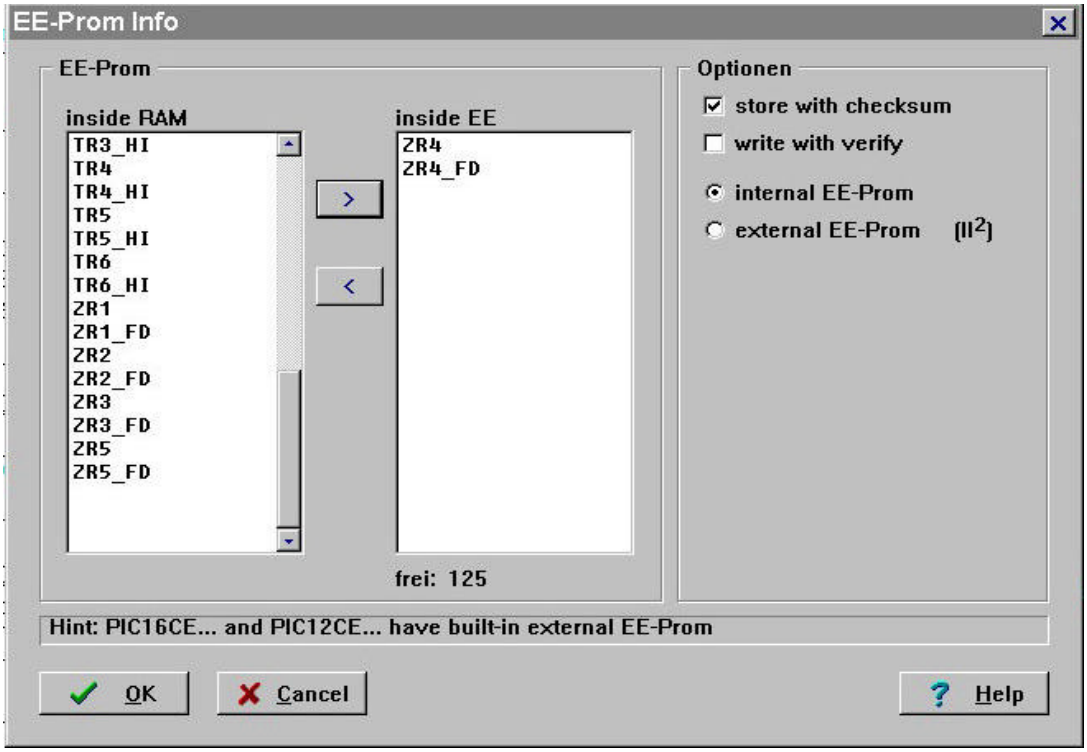
EE

Gestione della EEprom interna.

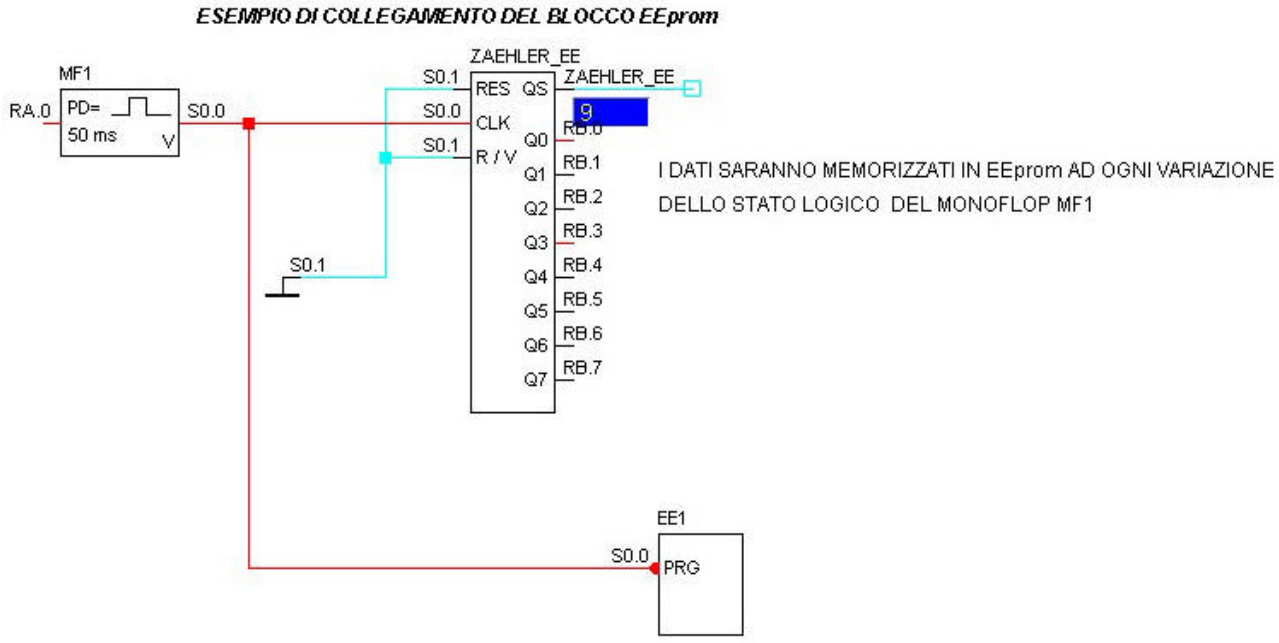
Esempio:

vogliamo trasferire il valore della variabile ZR4 in EE.

Le variabili ZR4 ed ZR4_FD sono collocate nella tabella di sinistra. Posizionarsi con il cursore del mouse su una delle due variabili e selezionarla (tasto sinistro del mouse). Ora portarsi sul pulsante di trasferimento, posto al centro della tabella, e spostate nella tabella di destra la variabile selezionata azionando il pulsante > : la variabile selezionata sarà trasferita nell'area EEprom. Si ripete l'operazione con la seconda variabile e così via, fino a trasferire in EEprom tutte le variabili di nostro interesse. Ovviamente si può operare il procedimento inverso , nel caso di trasferimenti non più necessari. **Si consiglia di attivare la funzione Cecksum.**



EE



Gestione della EEprom esterna.

Per collegare la Eeprom esterna al Pic, si adopera il bus I2C, (Inter-Integrated Circuit), adoperando lo standard Philips. Nella specifica dello standard I2C la velocità massima di trasferimento dei dati è di 100Kbps. Parsic rispetta il protocollo di comunicazione I2C . Di seguito Specifichiamo alcune terminologie proprie del bus I2C :

Transmitter	e' il dispositivo che invia i dati attraverso il Bus di collegamento;
Receiver	e' il dispositivo che riceve i dati dal Bus di collegamento;
Master	e' il dispositivo che inizia il trasferimento dei dati,genera il segnale di Clock, e termina il trasferimento dei dati;
Slave	e' il dispositivo indirizzabile attraverso il Master ;
Multi-master	sono più dispositivi Master che condividono lo stesso Bus di collegamento ed il cui sistema di controllo fa si che non si verifichino conflittualità nel sistema di comunicazione;
Arbitration	e' una procedura che consente ad un solo dispositivo master di controllare il bus di collegamento e fa si che non si verifichino conflittualità nel sistema di comunicazione;
Synchronization	procedura che consente di sincronizzare il segnale di clock a più dispositivi collegati tra loro.

Nota :

Parsic consente di operare con più dispositivi I2C collegati sullo stesso bus di comunicazione, però si consiglia di operare **sempre** con soli **due** dispositivi per volta, cioè con **un** dispositivo **Master** ed **un** dispositivo **Slave.** Nello schema elettrico funzionale del sistema distingueremo :

il circuito **Master** : (Pic) trasmette e Slave riceve; il circuito **Slave** : (EEprom) trasmette e Master riceve.

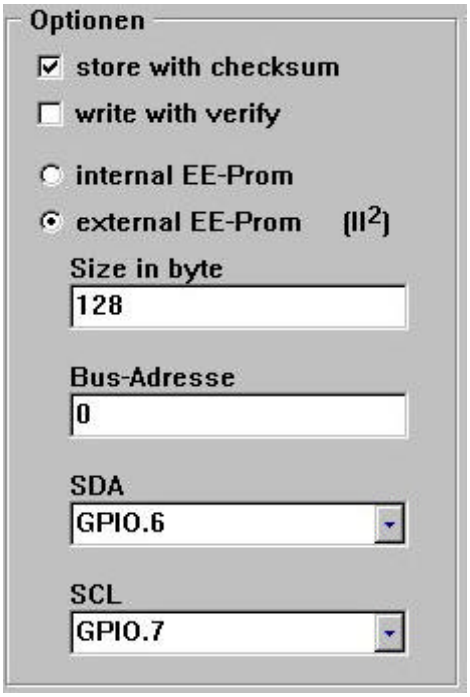
In ogni caso sarà sempre il **Master** a generare il segnale di clock.

Lo stadio di uscita del Clock si chiama **SCL** e quello dei dati **SDA**. Questi stadi sono costituiti da transistor configurati **open collector** oppure **open drain**. Una resistenza esterna di pull-up deve essere necessariamente collegata alle porte di comunicazione.

Denominazioni dei port di comunicazione.

Bisogna distinguere la denominazione delle porte di comunicazione secondo il tipo di Pic utilizzato. Per i Pic della serie **12CExxx** , Parsic nomina i port in questo modo:

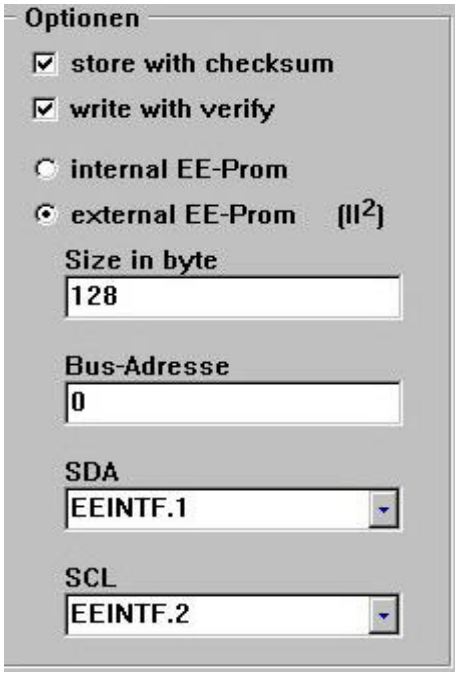
SDA = GPIO.6
SCL = GPIO.7



EE

Per i Pic della serie **16CExx**,**16Fxx** Parsic nomina i port in quest'altro modo:

SDA = EEINTF.6
SCL = EEINTF.



Salvataggio con cecksum.

Salvando in eeprom i dati ed utilizzando questa opzione, è generato il codice di cecksum di un byte. Questo codice è impiegato per confrontare il corretto valore dei dati memorizzati in eeprom. Se il dato memorizzato non e' corrispondente al cecksum, esso è azzerato.

Check after writing (verify)

Utilizzando questa opzione, il dato trasferito in EEprom è controllato nella sua integrità. Se questo dato non è scritto correttamente, il processo di trasferimento viene ripetuto.

Slave address

Le EEprom , sono identificate attraverso un indirizzo di collegamento. Quest'indirizzo è specificato nella casella Bus-Address con la denominazione A0, A1 e A2. Per default questa denominazione e' 0 (zero).

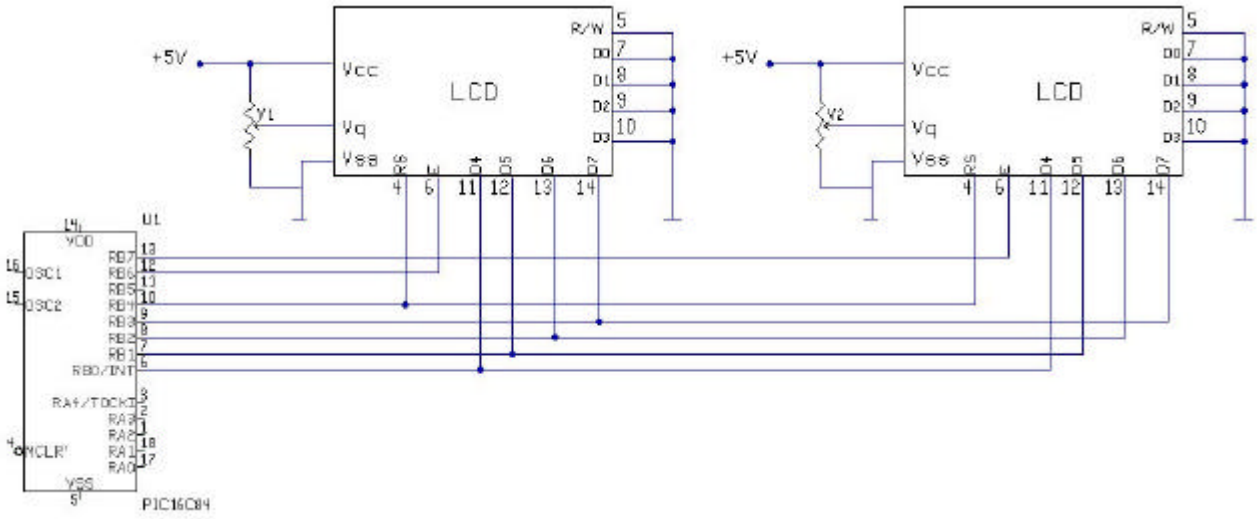
N.B. non tutti i PIC della serie 16Cxx supportano la gestione della EEprom esterna.

Gestione display LCD alfanumerici.

I display LCD più comuni dispongono di un'interfaccia ideata da **Hitachi** che nel tempo e' diventata uno standard industriale utilizzato nella costruzione dei display LCD alfanumerici. Questo tipo di interfaccia prevede che il display sia collegato al micro tramite un bus di **4 o 8 linee dati** , più due linee di controllo.

Parsic impiega , appunto, lo standard Hitachi HD44780 , per la gestione dei moduli alfanumerici LCD.

Le linee di pilotaggio dei moduli sono ridotte a sei, con evidente risparmio delle risorse I/O dei Pic. Il programma consente di collegare, all'uscita dello stesso dispositivo Pic,fino a **tre moduli LCD**. Le linee utilizzate per il pilotaggio, sono quelle relative ai **PortB** (default) ma, secondo le necessità del progetto, si possono impiegare port diversi (A,B,C,D). Sono impiegati i port **Rx.0 ..... Rx.4** per i **dati** (D) ed i port **Rx.5...Rx.7** (E) per le **linee di controllo**. Lo schema seguente mostra il collegamento di due display LCD che utilizzano le linee dello stesso port:

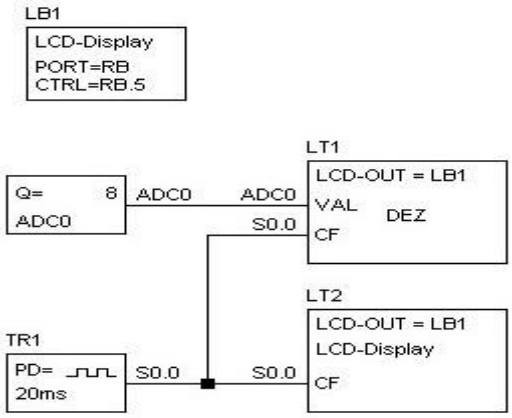


Dato che il tempo necessario alla corretta visualizzazione dei caratteri può variare secondo il tipo di LCD impiegato, è possibile modificare il tempo di trasferimento dei dati dal microprocessore al modulo LCD , impostando tre diversi tempi di propagazione :

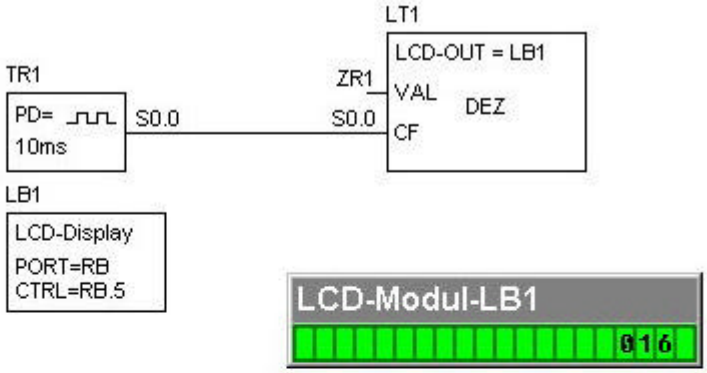
x 1 = 50 ? **sec;**
x 2 = 100 ? **sec;**
x 3 = 150 ? **sec;**

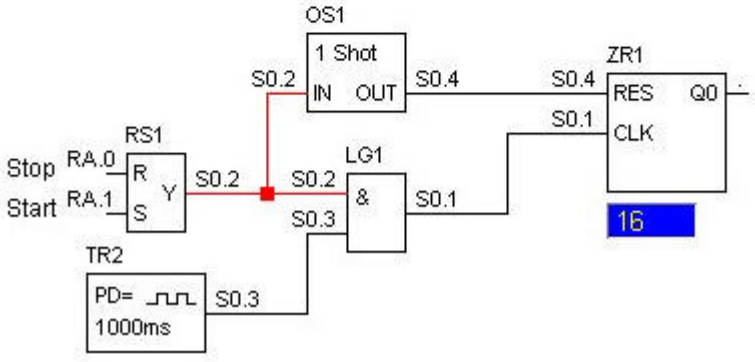
Prima di collegare il display al Pic, si consiglia di leggere con attenzione **il data sheet** del display LCD che si vuole adoperare. Le terminazioni dei display LCD di attuale produzione sono equivalenti, fra marche diverse di produttori, e seguono lo stesso standard di numerazione. Capita che alcuni display, prodotti negli anni 90 ,ancora in circolazione sul mercato, dispongano di piedinature predisposte diversamente dallo standard attuale. Ricordiamo inoltre che i display LCD **40x2** dispongono di due ingressi **Enable** : il primo serve ad abilitare le prime due righe ,il secondo le altre due.

Lo schema elettrico mostra il collegamento dei blocchi funzionali LCD.



Per visualizzare in continuità i messaggi sul display LCD, è necessario attivare il terminale CF con un generatore di clock,oppure con un impulso generato in sincronismo con il messaggio che si desidera evidenziare .

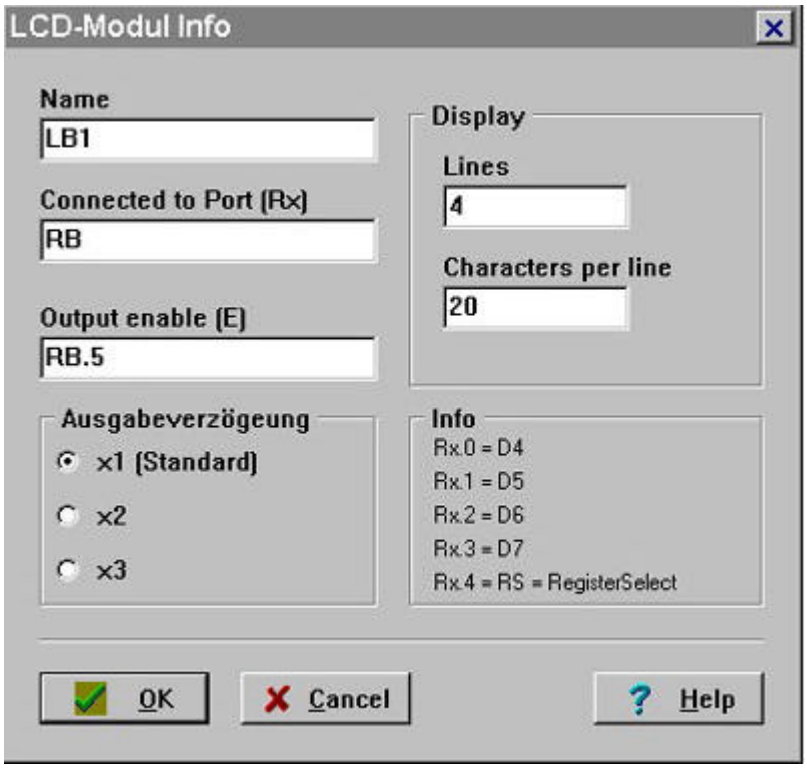




Inserimendo del testo nei display LCD

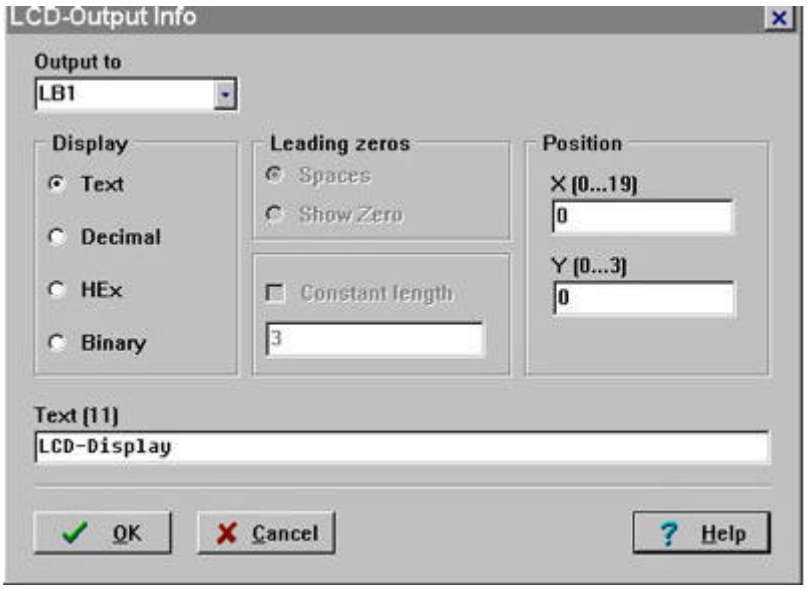
Modulo base.

Per rendere funzionante un display LCD, e' necessario posizionare nello schema elettrico il **modulo base** che definisce le caratteristiche elettriche del display LCD. Nello schema elettrico precedente si distingue nettamente il modulo base **LB1** dal display **LT1**. In ogni schema elettrico e' necessario **un solo modulo base** ed **nn moduli LTx**, tanti quanto sono le frasi che vogliamo gestire.



Nel riquadro **Name** specificare il codice del modulo LCD (LB1,LB2,LB3, ecc).
Specificare su quale Port collegare il display, **Connected to Port (Rx)**, e quale Pin sarà specializzato per pilotare il terminale enable , **Output enable (E).** Si possono impiegare display LCD standard disponibili in commercio specificando nel riquadro Display le linee ed il numero di caratteri.
Nel riquadro **Ausgabeverzoogeung** , si possono selezionare 3 diversi tempi di scrittura dei caratteri sull'LCD. Per LCD di vecchia generazione si consiglia di utilizzare l'opzione x3.

Modulo di testo.



Per ogni rigo di testo, bisogna posizionare nello schema elettrico **un modulo LT** , selezionandolo, dalla Toolbar. Una volta in posizionato il modulo attivate con il tasto destro del mouse la finestra di dialogo.
Specificate il nome del modulo che state trattando **Output to..,** in quanto lo stesso bus potrebbe gestire più di un display. Si possono impiegare caratteri in formato testo, decimale, esadecimale, o binario. I caratteri sono descritti nel riquadro **Text** e occuperanno lo spazio necessario alla loro rappresentazione. Si decide in quale posizione X (matrice) e rigo Y si desidera iniziare a scrivere il testo. Prestare attenzione quando si tratterà di gestire messaggi commutati,ovverro diciture collocate nello stesso rigo e stessa posizione. (per esempio :ON-OFF, APERTO-CHIUSO ecc.)
Le diciture devono coincidere fra loro, occupando le stesse posizioni. Eventuali frasi più corte rispetto ad altre più lunghe, saranno compensate immettendo degli spazi vuoti. In altri termini, se la prima frase occupa 12 spazi, anche la seconda frase dovrà occupare gli stessi spazi; se più corta si dovranno aggiungere n spazi fino a raggiungere la stessa lunghezza della prima (e viceversa). Si possono inserire caratteri speciali specificando il codice del carattere preceduto e chiuso dal segno grafico #.

Esempio: #$E2# , #226# ,#226,S#

Rappresentazioni numeriche decimali.

Per visualizzare dati numerici ,ci sono tre modalità di rappresentazione :

lunghezza variabile delle cifre **Spaces** : i numeri sono rappresentati in crescendo,con shift a sinistra della cifra, nessuno zero iniziale;

rappresentazione degli zeri **Show zero** : la cifra è rappresentata da una serie di zeri iniziali che sono sostituiti dal valore numerico corrispondente ,mano a mano che procede il conteggio;

lunghezza costante **Constant lenght** : è rappresentato un valore numerico di lunghezza costante. La rappresentazione numerica non può subire incrementi oltre la cifra impostata.(max 5 cifre)

La rappresentazione dei dati numerici decimali, a lunghezza costante, è utile quando si vogliono rappresentare valori derivati, ad esempio, da una conversione analogico-digitale.
Ad esempio, desideriamo che del valore di conversione equivalente a 1223, sia rappresentato solo il valore 23: si spunterà la casella **Constant lenght** e si inserirà la costante 2, nell'apposito riquadro.

Blocco funzionale Sleep mode

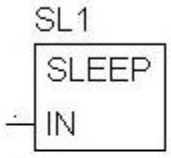
L'istruzione **SLEEP** è impiegata per mettere i Pic in **Power Down Mode** e ridurre di conseguenza la corrente assorbita da Pic che si stabilizzerà a pochi ?A, ovvero **1000 volte in meno**, circa, il valore normalmente assorbito dal microprocessore.
Parsic provvede ad inserire l'istruzione **SLEEP** in un determinato punto del file sorgente, spegnendo tutti i circuiti interni, tranne quelli necessari a mantenere attivo lo stato delle porte I/O ed a rilevare le condizioni di risveglio del processore.
Per ridurre il consumo energetico in questo stato, questi circuiti devono essere progettati in modo da limitare il loro assorbimento nelle condizioni di **Power Down**. Si rimanda l'utente alla lettura delle Application Note di Microchip.
Si consiglia di collegare al positivo **Vdd** o al negativo **Vss** di alimentazione tutte le linee I/O non utilizzate. Quando le linee di ingresso **sono tutte ad alto livello**, è attivato lo **SLEEP MODE** ed il programma è posto in condizione di **STOP**, **l'oscillatore e' posto ad off**.

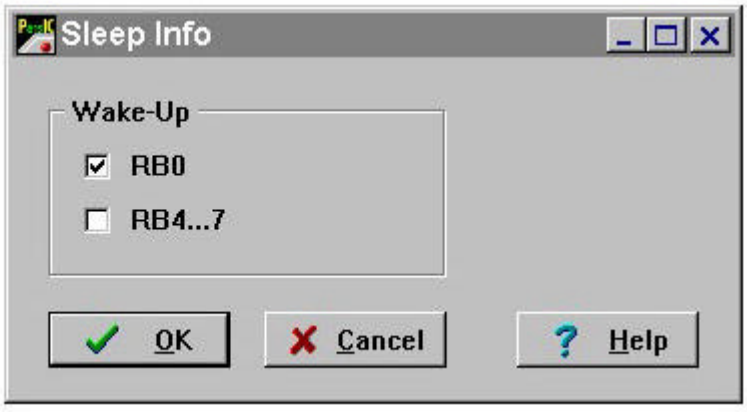
Per ripristinare le funzioni del microprocessore ci sono due possibilità:

la prima :	applicare un impulso positivo al port **RB.0**;
la seconda :	applicare una variazione di livello ad uno dei port **RB.4, RB.5, RB.6, RB.7.**

Restrizioni dell'uso della funzione di Sleep mode.

Non e' possibile abilitare la funzione **Watch-Dog Timer** se è utilizzato il **blocco funzione Sleep** . Questa funzione non e' implementata nei microcontrollori della serie **12CExxx.**





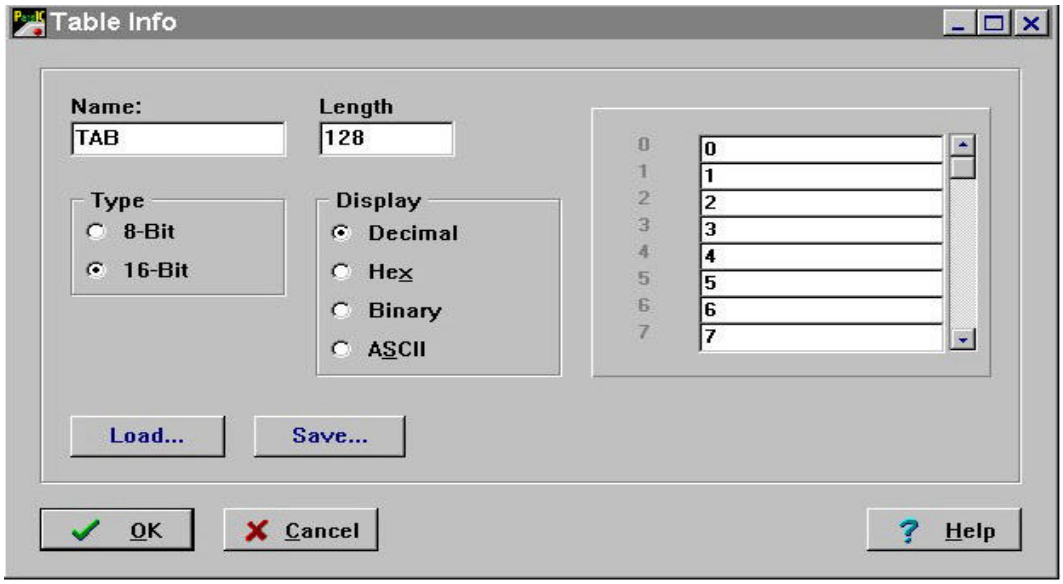
Blocchi funzionali Table e Call (subroutines).
Istruzioni di transcodifica.

Un valore numerico può essere espresso in una word con diversi codici,cioè in svariati sistemi di numerazione; la codificazione può essere fatta inizialmente con un sistema di numerazione,mentre può successivamente essere convertita in un codice diverso : questa operazione si chiama transcodifica. Si potranno convertire valori da formato binario in formato BCD, oppure da formato BCD in formato Hex,oppure da formato binario a formato decimale,ecc. I blocchi Table e Call,sono preposti al funzionamento di tale transcodifica.

Table.

Il blocco Table,contiene determinati valori,scritti in apposite righe numerate, facenti parte di una tabella. Il numero massimo di righe a disposizione dell'operatore sono definibile in base alla risoluzione del dato che si vuole convertire. Per dati con risoluzione 8 bit,le righe disponibili sono 256,per dati con risoluzione a 16 bit ,le righe disponibili sono 128.
I valori sono scritti secondo una gerarchia di transcodifica,decisa all'atto dell'impostazione del progetto. Tale dati sono strettamente collegati al canale del codice sorgente,proveniente,ad esempio da un modulo funzionale contatore oppure,come nell'esempio che segue,dall'uscita di un canale multiplexer. Il risultato della conversione è disponibile all'uscita del blocco Call.
Nel riquadro si riepiloga il funzionamento del blocco funzionale Table :

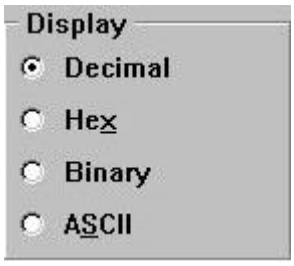
Valori a 8 bit	sono catalogati in una tabella contenente **256** locazioni (0...255) specificati in numerazione crescente da 1 a 256, il cui valore e' compreso tra **0 e 255** ;
Valori a 16 bit	sono catalogati in una tabella contenente **128** locazioni (1....128) specificate in numerazione crescente da **0 a 127**, il cui valore e' compreso tra 0 e 65535. La sequenza dei valori è compresa in una scala contenente il low byte ed high byte .



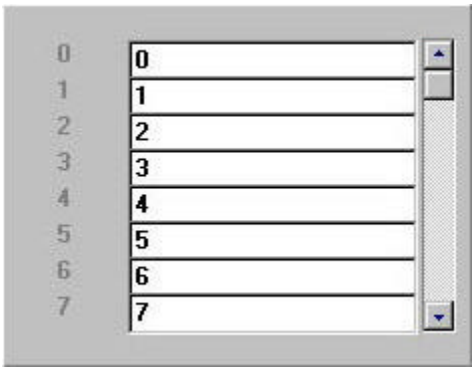
Table

dare un nome identificativo alla tavola e definire la sua lunghezza, cioè il numero delle variabili che essa deve contenere.

Definire il tipo di dato se a 8 oppure 16 bit



Definizione dei dati in tabella
Decimale = numeri compresi tra 0 e 9;
Hex = valori numerici e lettere, preceduti dal simbolo $ es. $7F
Bin = valori digitali 0 - 1. Il bit piu' significativo viene posto a sinistra.
Asc = codici ASC , preceduti dal segno '

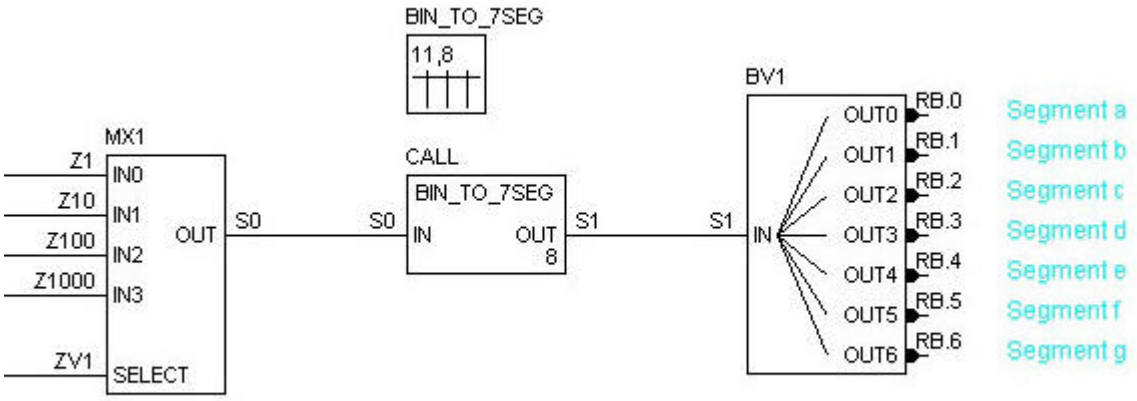


Indipendentemente dal tipo di rappresentazione che si e' scelto nel riquadro display, inserire i dati partendo dal rigo 0 della tabella.

Il contenuto della tabella può essere salvato in formato txt. con il comando **Save...**
Il contenuto di una tabella, precedentemente salvato, viene ricollocato in tabella in formato txt. I valori sono riposti con lo stesso ordine in cui sono stati salvati. Le linee precedentemente non utilizzate sono ignorate.

Call.

Il blocco funzionale **Call** serve per richiamare i dati collocati in tabella.
I dati contenuti in tabella, sono richiamati dalla **Call**, per mezzo di un valore numerico proveniente da un qualunque blocco funzionale del programma.



In quest'esempio notiamo che l'uscita del modulo **MX1** è collegato all'ingresso del modulo **CALL**
Il modulo multiplexer MX1 è ,a sua volta ,collegato ai blocchi circuitali denominati Z1…Z1000
Operando la selezione degli ingressi Zxx attraverso ZV1, in uscita al modulo MX1 avremo una serie di valori numerici compresi tra 0 e 255. Questo valore numerico attiverà l'uscita del dato contenuto nella tavola di conversione,come da esempio seguente:

selezione	ingresso binario	uscita binaria	indirizzo rigo tavola	uscita modulo Call
ZV1=0	MX1 IN0 =123	MX1 OUT =123	Rigo Tavola = 123	Call OUT= 26
ZV1=0	MX1 IN0 = 45	MX1 OUT = 45	Rigo Tavola = 45	Call OUT= 82

ZV1=0	MX1 IN0 =237	MX1 OUT =237	Rigo Tavola = 237	Call OUT= 12
ZV1=0	MX1 IN0 = 65	MX1 OUT =65	Rigo Tavola = 65	Call OUT= 127
ZV1=0	MX1 IN0 = 93	MX1 OUT =93	Rigo Tavola = 93	Call OUT= 66

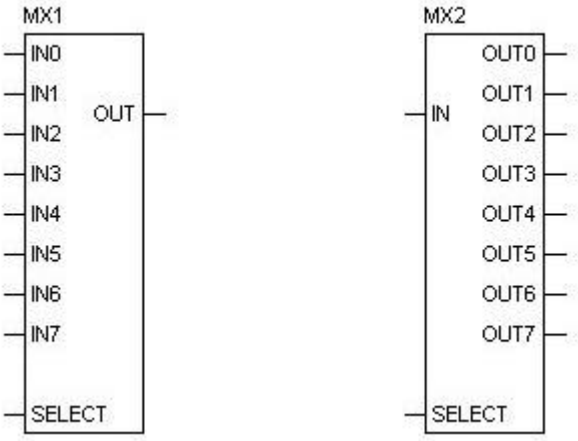
Commento :

Il valore binario123, in ingresso al modulo **MX1/Z1** è selezionato per mezzo di **ZV1** : andrà a indirizzare il rigo 123 della tabella,dove era stato posto il dato 26.
Questo dato è richiamato dal modulo **CALL** che lo invierà in uscita al modulo **decodificatore BV1**.
La sequenza si ripete all'infinito ed è valida per qualunque tipo di selezione si operi su **MX1.**

L'impiego della tabella di conversione è utile in quei processi che richiedono,ad esempio, una correzione dati negli AD converter, oppure come nel caso dell'esempio, nella conversione di un dato binario, in un codice 7 segmenti dei display led.

Multiplexer

Un multiplexer è un dispositivo digitale che può selezionare uno tra più ingressi/uscite e trasferire lo stato logico di tale ingresso/uscita sull'unica uscita/ingresso. Questo dispositivo agisce come "unidirectional single-pole position switch", che passa l'informazione di un ingresso/uscita verso l'uscita/ingresso



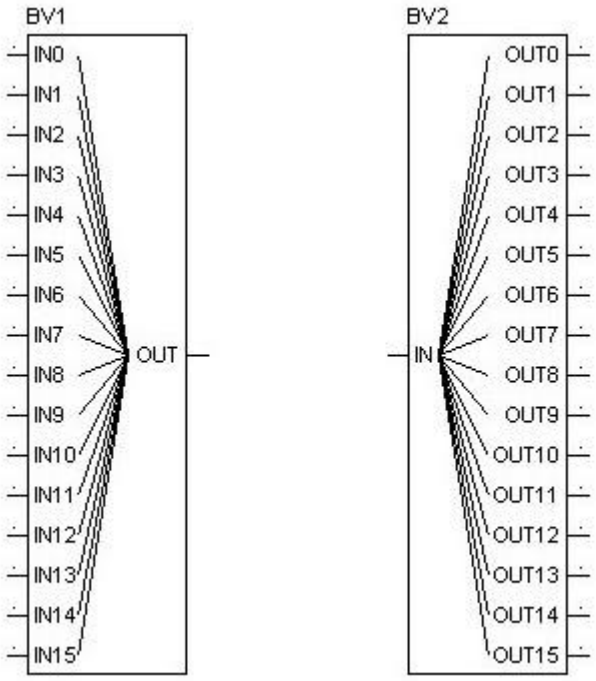
Nel blocco di sinistra, **multiplexer**, le linee di ingresso sono commutate in uscita, ad una ad una, per mezzo della variabile di selezione applicato al terminale Select. Tale variabile ,proveniente ad esempio da un contatore ZV avrà un'escursione numerica compresa tra 0 e 7.
Nel blocco di destra , **de-multiplexer**, le linea di ingresso e' commutata alle uscite, utilizzando ancora il terminale Select. La selezione avviene sempre per una linea d'uscita alla volta.
Se la selezione impostata al terminale Select è più alta del numero di ingressi/uscite ammessi, essa viene ignorata dal programma. Gli ingressi del multiplexer andranno collegati ad una sorgente digitale il cui valore binario è di 8 oppure16 bit. Agli stessi ingressi ,è possibile collegare una costante digitale,scrivendo tale valore direttamente su uno dei gate del module .

Terminale INx	ingresso della variabile o costante digitale ;
Terminale OUT	uscita ;
Select	selezione del gate IN/OUT .

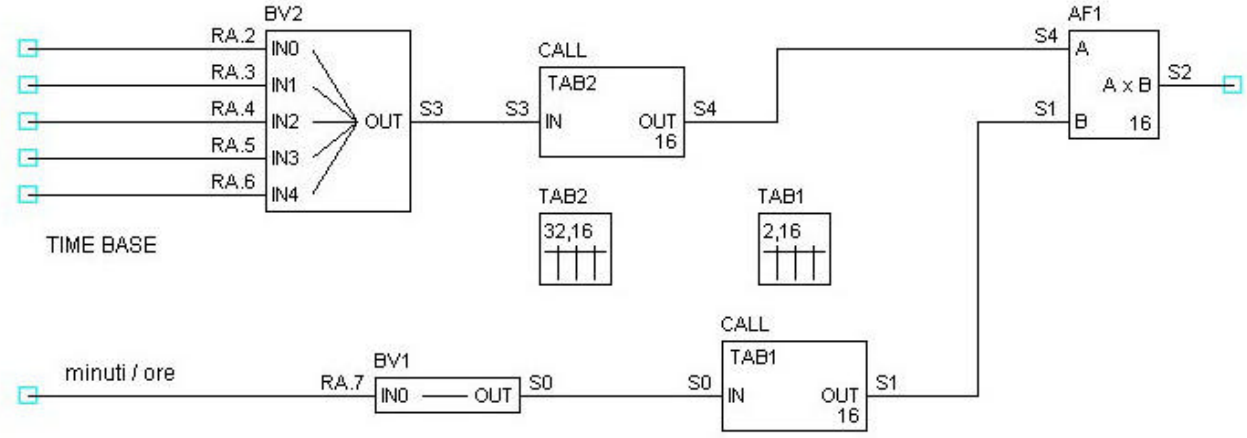
Blocco funzionale de-codificatore Encoder /Decoder

Sebbene la maggior parte dei circuiti digitali operi con due stati logici,alcuni individui preferiscono lavorare con l'aritmetica digitale,cioè rappresentando tutti i numeri con gli interi e le potenze in base 10. Considerando che diventa vantaggioso codificare un numero decimale in forma binaria, il blocco funzionale descritto in questo paragrafo svolge un'importante azione ai fini della semplificazione degli schemi elettrici e del raggruppamento dei segnali digitali.

Questo blocco funzionale, svolge l'importante funzione di **codifica** o **decodifica** delle variabili binarie direttamente rappresentate con numerazione decimale. Abbiamo visto precedentemente, nell'esempio del modulo CALL, l'impiego della decodifica utile a pilotare un display led a 7 segmenti.
Uno dei codici più usati per codificare i numeri decimali è il codice 8 4 2 1,detto anche binario decimale codificato BCD. E' chiaro che il binario codificato è un codice molto semplice di quattro bit che inizia da 0000 e termina a 1001: ogni decimale è codificato con quattro bit. Esistono altri codici più complessi ,come l'esadecimale, l'alfanumerico,l'ottale, ecc.
I moduli Encoder (BV1) sono di tipo esadecimale e l'attivazione di ogni gate di ingresso determina il corrispondente valore binario di uscita. Attivando più ingressi, otterremo in uscita la somma dei valori binari corrispondenti. La rappresentazione del valore binario, in uscita dal modulo, è di tipo decimale (0…65534). Il modulo Decoder (BV2) svolge analoghe funzioni al precedente,solo che a differenza del primo, l'unico valore numerico d'ingresso è decodificato sulle porte di uscita.
Ad esempio il valore numerico 2084 in ingresso al modulo decodifica BV2 determina l'attivazione delle uscite OUT 2/5/11. Gli ingressi IN 2/5/11 attivati al modulo codificatore BV1 produrrebbero in uscita al modulo una variabile di valore numerico decimale 2048.



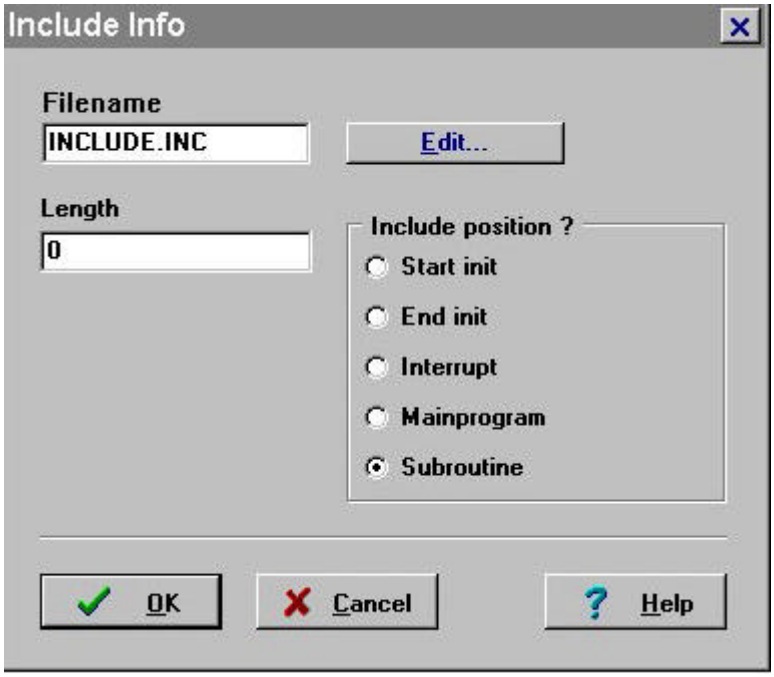
L'esempio circuitale che segue mostra l'impiego di due circuiti di codifica, utilizzati per selezionare il cliclo di un temporizzatore:



Il codifica BV2 produce in uscita un valore numerico dipendente dalla somma dei gate Inxx attivati.
La codifica BV1 può assumere solo due valori 0 e 1. Le uscite delle decodifiche BV1 e BV2 selezionano i dati delle tabelle TAB1 e TAB2 . Il prodotto dell'operatore matematico AF1 determina il set del temporizzatore.

Include 

La direttiva INCLUDE, ordina al compilatore di inserire il contenuto di un file di inclusione nel programma in fase di compilazione. Un file include è un file che contiene informazioni aggiuntive, scritte in un secondo file.Il compilatore si limiterà a sostituire la linea contenente la direttiva INCLUDE con il contenuto dei file indicati ed effettuare la compilazione come se fosse anch'esso parte del source elaborato da Parsic. I file di inclusione avranno tutti l'estensione .INC. Inoltre gli stessi saranno collocati in una sezione ben definita del sorgente Parsic,specificando in fase di scrittura dove andrà collocato il file include : Include position ?
Procedendo, l'utente inserirà nel sorgente Parsic una subroutine specializzata per una determinata funzione.



Nel riquadro 

scegliere in quale posizione del file sorgente si desidera inserire il file Include;

Nel riquadro 

specificare il nome del file, scritto in codice assembler. Il file và salvato con l'estensione .INC

Utilizzare l'opzione 

per scrivere il file Assembler, in formato testo ;

Nel riquadro 

inserire le dimensioni del file.

Esempi di programmazione assembler con l'impiego del modulo include

PIC16F628 .
Predisposizione del funzionamento dei comparatori in modo 4. (vedi data sheet PIC16F628)

Filename = COMP4
Lenght = 4
Include position = fine programma

Testo ASM:

```
; switch to comparator modus 4
; see datasheet for other modes

movlw 4                 ; load a 4 into work
banksel  cmcom          ; switch to bank cmcom
movwf   cmcom           ; write
```

Esempio di programmazione di una matrice di tasti 4x4 con PIC16F877

Testo ASM :

```
Filename  = KEYINT.INC
Lenght    =  4
Include position = interrupt
---------------------------------------
; Change used port here
; in this example PORTB is used

#define MyPort PORTB

;---------------------------------------
KPDELAY EQU 255-80      ; ~20ms
;---------------------------------------

        btfss intcon,t0if       ; only if tmr0 interrupt
        goto notimerint

;-------------------------------------------------------

        movlw H'80'   ; first row

kp1
        banksel ktemp1          ; store
        movwf ktemp1

        banksel myport          ; set row
        movwf myport
        nop
        nop
        movfw myport            ; read
        andlw H'0F'
        btfss status,zero       ; key pressed ?
        goto kp3                ; yes

        bcf status,carry        ; no
        banksel ktemp1
        rrf ktemp1,f    ; next row
        movfw ktemp1
        btfss ktemp1,3; all done ?
        goto kp1        ; no

;-------------------------------------------------------

kp2
        banksel key     ; no key pressed
        clrf key
        banksel ktemp2          ; clear previous
        clrf ktemp2
```

```
        goto kp1
;-------------------------------------------------------
kp3
        banksel myport          ; a key is pressed !
        movfw myport            ; same as previous ?
        banksel ktemp2
        subwf ktemp2,w
        btfss status,zero
        goto kp4        ; no

        banksel ktemp3          ; yes, - is timer zero ?
        movf ktemp3,f
        btfsc status,zero
        goto kp10       ; yes

        incfsz ktemp3,f         ; no - count up to zero
        goto kp10

        banksel ktemp2          ; timer is zero
        movfw ktemp2            ; copy ktemp2 to key
        banksel key
        movwf key
        goto kp10
;-------------------------------------------------------

kp4
        banksel myport          ; copy pressed key to ktemp2
        movfw myport
        banksel ktemp2
        movwf ktemp2
        movlw kpdelay           ; and set timer
        banksel ktemp3
        movwf ktemp3
        banksel key
        clrf key        ; clear current pressed key

;-------------------------------------------------------

kp10
        banksel myport          ; always set direction for used port
        clrf myport
        movlw H'0F'
        bsf status,rp0
        movwf myport
        banksel myport

notimerint
```

Nota: inserendo l'include nella posizione **Interrupt**, e' necessario che il file contenga un ciclo di temporizzazione . Diversamente il file include sarebbe ignorato dal programma.

Comunicazione seriale. **UART Base**

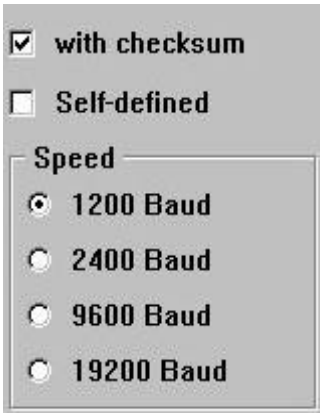
Lo standard **RS232** definisce una serie di specifiche per la trasmissione seriale dei dati tra due dispositivi denominati **DTE** (data terminal equipment) e **DCE** (data communication equipment). Il **DCE** e' un dispositivo che si occupa di gestire una comunicazioni dati mentre il **DTE** e' un dispositivo che si occupa di generare o ricevere dati. In pratica l '**RS232** e' stata creata per connettere tra loro un terminale dati ,con un modem per la trasmissione a distanza dei dati generati.
Per la connessione tra due computer sono necessari cinque dispositivi di comunicazione: un computer **DTE**, un modem **DCE**, una linea di collegamento, ancora un'altro modem **DCE** ed infine un computer **DTE**.
Per collegare tra loro due computer (microprocessori) si può utilizzare una linea senza interporre alcun modem. Questo tipo di connessione si chiama **Null Modem** e si realizza con un cavo elettrico,invertente, in grado di scambiare i dati tra i due computer (microprocessori).
La comunicazione che Parsic e' in grado di gestire e' di tipo **RS232** asincrona, cioè un tipo di trasmissione che non prevede affatto che la comunicazione tra due dispositivi diversi sia di tipo sincronizzata.
La trasmissione e la ricezione tra due dispositivi, in comunicazione tra loro, avviene per mezzo di due fili ed un collegamento comune riferito a massa. Il formato di trasmissione e' del tipo **8N1**:

8 bit data, nessuna parità, nessun bit stop

Non tutti i microprocessori PIC sono dotati di porta di comunicazione seriale. Se, per errore, si seleziona un dispositivo non adatto all'impiego, è generato un segnale di errore.

Baud rate.
La velocità di trasmissione **Baud-rate** è riferita al valore della frequenza di clock che corrisponde a quella dell'oscillatore al quarzo impiegato, cioè 4MHz. Si raccomanda di utilizzare sempre un oscillatore al quarzo per le comunicazioni seriali. Portando il cursore del mouse sul blocco UART , cliccando con il tasto destro, si apre la finestra di dialogo UART-Info.

Baud-rate. Uart Base
Predisposizione del modulo di comunicazione seriale.



La velocità di trasmissione è impostata nel riquadro Speed spuntando una delle velocità già preimpostate.
Se è impostata l'opzione Self-defined, il Baud -rate è definito con un semplice calcolo matematico.

Se BRGH è mantenuto a 0 (casella BRGH non selezionata), per il calcolo del Boud-rate procedere in questo modo:

X = e' il valore da immettere in SPBRG ed e' compreso tra 0 e 225;
Fosc = e' la frequenza esterna dell'oscillatore (4 MHz)
Baud-rate = Fosc/(64(X + 1)

oppure (Fosc/64) / Baud-rate -1

Se BRGH è mantenuto ad 1 (casella BRGH selezionata) per il calcolo del Baud-rate procedere in questo modo:

Baud rate = Fosc/(16(X+1))

oppure (Fosc/16) / baud-rate -1

Esempio:

Baud-rate 19200, frequenza esterna = 4 MHz, BRGH=1
SPRG = (4000000 / 16) / 19200 -1
SPRG = 250000 / 19200 -1
SPRG = 13,0208 -1
SPRG = ~ 12
Il valore SPRG da inserire e' 12

TX - RX Pin. Uart Base

Si può definire quale pin sarà impiegato come terminale TX. Se è usato il terminale hardware,definito dal produttore Microchip, questi viene denominato:

RB.2 per i dispositivi PIC a 18 pin;
RC.6 per i dispositivi PIC a 20 e 40 pin.

Per il terminale RX valgono le stesse regole di cui sopra ,distinguendo diversamente i pin:

RB.1 per i dispositivi PIC a 18 pin;
RC.7 per i dispositivi PIC a 20 e 40 pin.

Standard RS485. Uart Base

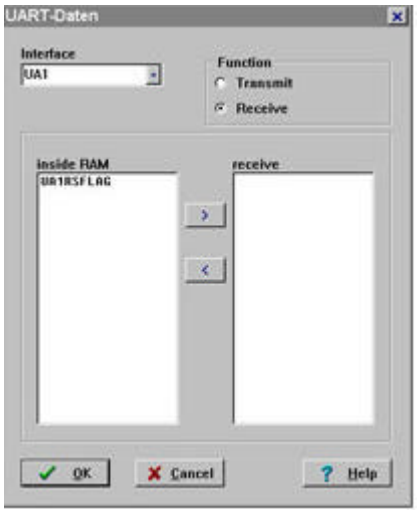
Negli ambienti industriali ed in piccole reti di microcontrollori è molto usato lo standard RS485.
Lo standard RS485 è uno standard half-duplex: la trasmissione dei dati è bidirezionale ma non contemporanea, occorre quindi che si indichi in modo esplicito se si vuole realizzare una ricezione o una trasmissione. L'utilizzo di un dispositivo hardware per la selezione della direzione ha il vantaggio di non richiedere cambiamenti software di comunicazione rispetto ad un normale cavo seriale null-modem, con le uniche avvertenze di disattivare il controllo hardware del flusso dei dati e di evitare di trasmettere e ricevere contemporaneamente.

Selezionando lo standard RS485, nel riquadro **Half-duplex** **DIR-PIN** ,si dovrà specificare quale pin di uscita al PIC si desidera utilizzare per il controllo Rx/Tx della porta seriale. Questo terminale è normalmente a livello basso durante la ricezione del segnale, commuta a livello alto durante la trasmissione e resta a tale livello fino al termine. (vedi schema elettrico allegato).

Checksum. Uart Base

Attivando quest'opzione è generato il cecksum (1 byte) al termine di ogni periodo di trasmissione.

Ricezione :
codice di checksum è comparato, durante il processo di ricezione del segnale, con il valore di checksum calcolato dal microprocessore. Se la comparazione è valida, i dati sono trasferiti nella variabile indicata a destra della tabella sottostante (riceive...):



Nel caso sia di interesse,il calcolo del codice di cecksum e' facilmente ricavabile dalla seguenti formule:

1. CS = 170 (HEX = $AA)
2. CS = CS + 1st byte
3. CS = CS + 2nd byte
4. ECC.

CS= checksum

Esempio:

Il contatore ZV5 presenta un valore di conteggio = 1000.
Nel riquadro di sinistra,del modulo di ricezione, vedremo indicati:

ZV5 (che rappresenta il low-byte del contatore, che al momento contiene il valore 232 ($E8))
ZV5_HI (che contiene l'high-byte del contatore = 3)
il seguente calcolo mostra come ricavare il contenuto del contatore ZV5 (3 x 256 + 232 = 1000)
il valore di checksum sarà determinato:

CS = 170 + 232 + 3 = 405 405 - 256 = 149 rappresenta il valore del (8) bit piu' basso trasmesso.

La sequenza di trasferimento conterrà i seguenti valori <232> <3> <149>

UART data

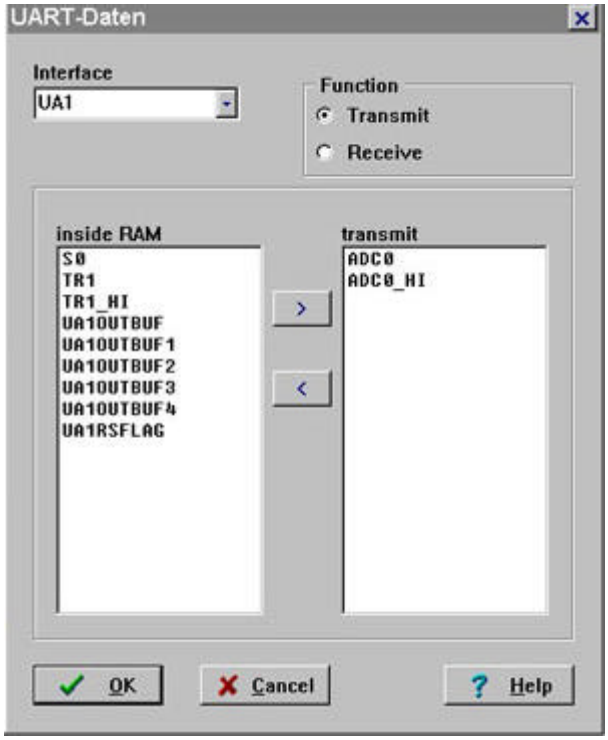
Utilizzando questo modulo e' possibile ricevere o trasmettere i dati sulla linea seriale. Il modulo si presenta con due terminazioni :

il terminale ACT e' il pin di ingresso/uscita dei dati;
il terminale Enable è il pin che abilita la ricezione/trasmissione dei dati.

Per tutto il periodo di trasmissione, il terminale ACT-Out rimane a livello logico alto.Durante la fase di ricezione, il terminale ACT-Out rimane a livello logico alto fino alla corretta ricezione del dato.
Utilizzando l' ACT si può determinare quale dei moduli, trasmittente-ricevente, dovrà restare attivo durante il trasferimento dei dati.

Trasferimento dei dati.

Le variabili del programma sono contenute in una lista, specificata nell'UART-data.

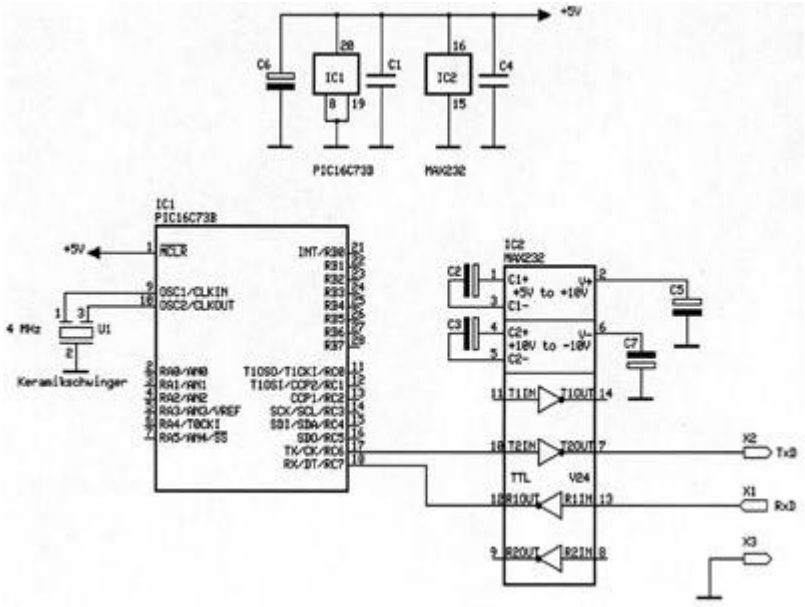


Le valori che si vogliono trasmettere saranno trasferiti nella tabella di destra, selezionando con il puntatore del mouse le variabili contenute nel riquadro Inside Ram, agendo poi, sul pulsante > .
La trasmissione dei dati avverrà nello stesso ordine in cui essi sono trasferiti nel riquadro Transmit.
La trasmissione si attiva quando è applicato un impulso positivo al terminale ENable :Al buffer di uscita saranno inviati i dati selezionati, più il checksum. E' sufficiente un solo impulso per attivare la trasmissione e, durante la stessa, altri impulsi di abilitazione non saranno considerati fino al termine dell'operazione.

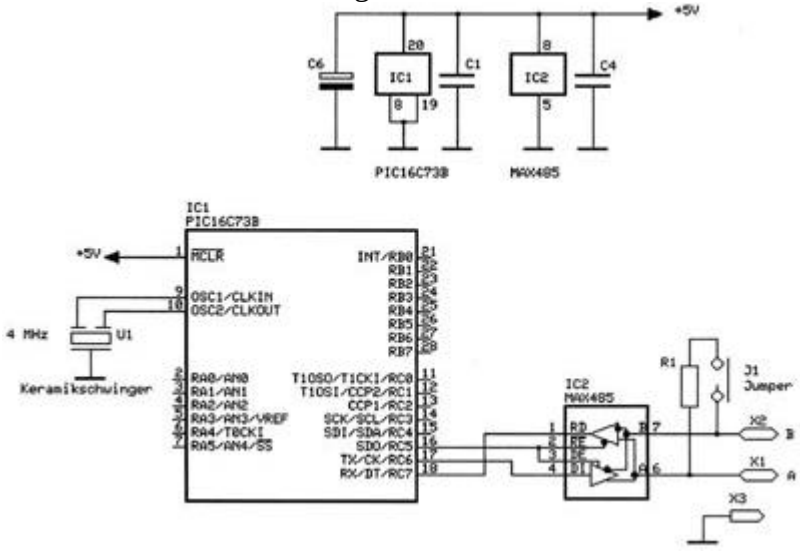
Ricezione dei dati Uart Data.

I dati che si vogliono ricevere saranno trasferiti nel riquadro Receive, con le stesse modalità specificate sopra. La ricezione dei dati ha inizio applicando al terminale ENable un impulso positivo e, durante la fase di ricezione, eventuali altri impulsi applicati al terminale ENable saranno ignorati. Se durante la ricezione si manifesta un errore nei dati o nel checksum, la ricezione è interrotta ed il terminale ACT-Out si porta a livello logico basso.

Schema elettrico di un collegamento seriale RS232

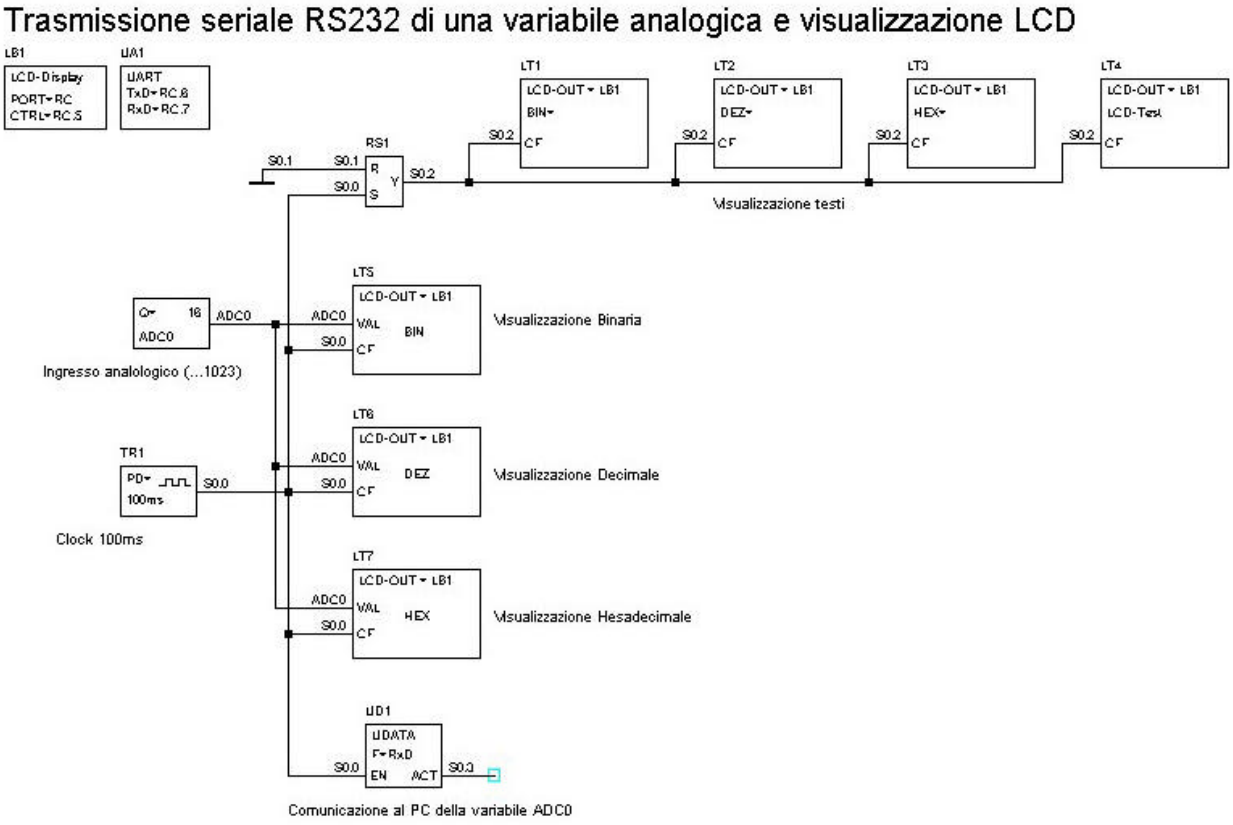


Schema elettrico di un collegamento seriale RS485



Esempio di una trasmissione RS232 .

Il PIC è utilizzato per trasmettere la variabile analogica ADC0 ad un PC.
Si noterà, nello schema funzionale, che il generatore di clock è utilizzato contemporaneamente per il refresh dei display LCD e per attivare la trasmissione seriale RS232.



L'editor di testo. 

Durante la costruzione dello schema funzionale si consiglia il frequente uso di quest'editor. Consente di aggiungere commenti e segnalazioni lungo il tracciato dello schema. Inoltre sarà di valido aiuto quando saranno ripresi schemi archiviati da tempo di cui non si ricordano più i particolari di progetto. Queste tracce non occupano spazio nella memoria del PIC pertanto è utile impiegarle a proprio piacimento. E' sufficiente posizionare l'etichetta vicino al componente o sezione di impianto che si desidera commentare.

I potenziali di riferimento. 

Utilizzando questa funzione si possono collegare ai terminali d'ingresso, dei blocchi funzionali, i riferimenti di potenziali **GND** e **Vcc**.
Questi riferimenti non sono altro che lo **0 (zero)** oppure **1 (uno)** logico digitale. Servono, ad esempio, a forzare un livello logico su una porta di ingresso di un operatore booleano, a mantenere lo zero logico al terminale di reset di un contatore,ecc.
Dopo aver selezionato il riferimento di potenziale, portatevi con il puntatore del mouse su di esso,operate sul tasto destro del mouse e,quando compare la finestra di dialogo,definite l'orientamento da dare all'oggetto e la sua polarità.

Indice

schmitt-trigger
DAT-ADC
Counter

schift-register
flip-flop
generatore di clock
monostabile su evento

monostabile one-shot
interrupt
PWM-PFM-DAC
Limiter o filtro digitale
EEprom

LCD. Come si collega un display alfanumerico

sleep mode
transcodifica

multiplexer
de-codifica deimale-binaria-decimale
la direttiva INCLUDE

UART

editor e potenziali pull-up/pull-down