Versione N° 1.0a

LaurTec

Controllo di 16 carichi attraverso la seriale del PC

Autore : *Mauro Laurenti*

email: info.laurtec@gmail.com

ID: PJ7005-IT

INFORMATIVA

Come prescritto dall'art. 1, comma 1, della legge 21 maggio 2004 n.128, l'autore avvisa di aver assolto, per la seguente opera dell'ingegno, a tutti gli obblighi della legge 22 Aprile del 1941 n. 633, sulla tutela del diritto d'autore.

AVVERTENZE

I progetti presentati non hanno la certificazione CE, quindi non possono essere utilizzati per scopi commerciali nella Comunità Economica Europea.

Chiunque decida di far uso delle nozioni riportate nella seguente opera o decida di realizzare i circuiti proposti, è tenuto pertanto a prestare la massima attenzione in osservanza alle normative in vigore sulla sicurezza.

L'autore declina ogni responsabilità per eventuali danni causati a persone, animali o cose derivante dall'utilizzo diretto o indiretto del materiale, dei dispositivi o del software presentati nella seguente opera.

Si fa inoltre presente che quanto riportato viene fornito cosi com'è, a solo scopo didattico e formativo, senza garanzia alcuna della sua correttezza.

L'autore ringrazia anticipatamente per la segnalazione di ogni errore.

Introduzione

Quando ci si rende conto delle potenzialità del computer si comincia subito a pensare ad un modo per poter controllare lampade e quant'altro direttamente dal PC. Per mezzo di questo progetto si mostra come sia possibile programmare un PIC affinché possa, per mezzo della scheda Freedom e delle schede di espansione con Relay e MOS di potenza, controllare sino a 16 carichi facendo uso della porta seriale.

Analisi del progetto

Il programma di gestione è stato scritto in C18[1] per il PIC18F4580 montato sulla scheda Freedom. Questa scheda è utilizzata come supporto principale ma nulla vieta di utilizzare una scheda personale, montata anche su mille fori, su cui montare solo l'hardware strettamente necessario per l'applicazione; in Figura 1 è riportato il montaggio del sistema ottenuto collegando direttamente le schede di espansione contenente 4 Relay e le schede con 4 MOS di potenza. La configurazione presentata gestisce in particolare 8 carichi poiché sono presenti solo 2 schede.

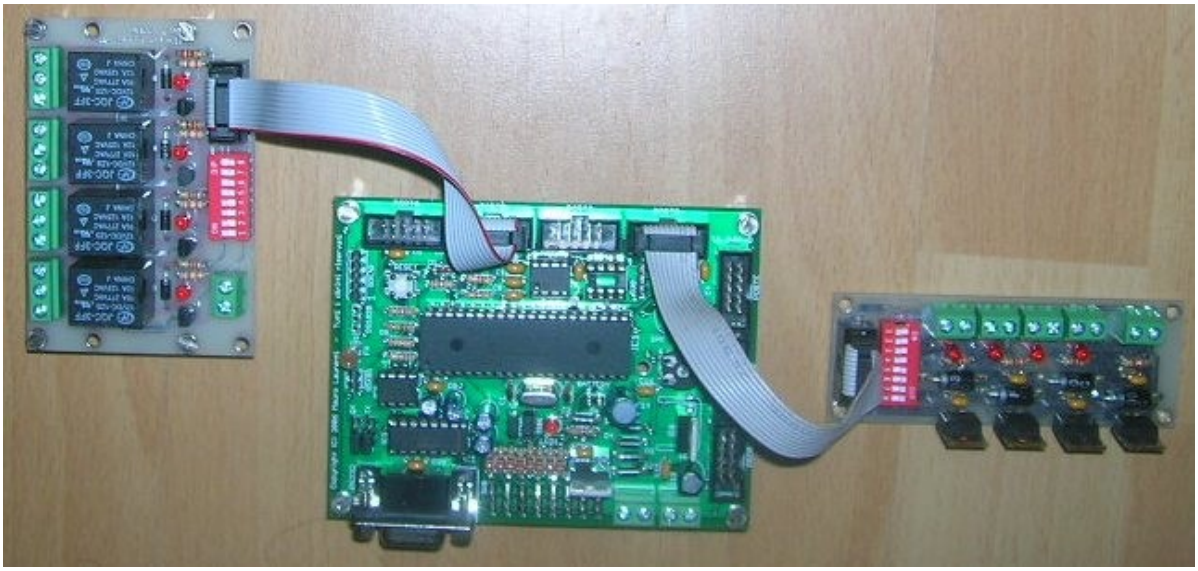

Figura 1: Configurazione 8 carichi

Le schede possono essere collegate alla PORTB e PORTD del PIC in una qualunque combinazione fino al massimo di 16 carichi ovvero un totale di 4 schede, questo significa che per ogni porta del PIC possono essere collegate due schede. Sulla modalità d'utilizzo delle schede di espansione si rimanda ai relativi articoli scaricabili dal sito www.LaurTec.com. L'utilizzo o meno di una scheda piuttosto che un'altra dipenderà semplicemente dalle esigenze, dunque nulla vieta di collegare 4 schede dello stesso tipo.

Il programma sorgente sotto riportato[2] è scritto facendo riferimento al software utilizzato per il progetto del Robot Domotino. In particolare dopo l'inizializzazione del PIC inizia un ciclo infinito in attesa d'interruzioni da parte della porta seriale. Ogni volta che viene ricevuto un dato questo viene posto nella variabile *data*, che rappresenta un array di caratteri, fino al raggiungimento di 5 byte che rappresenta la lunghezza del comando per mezzo del quale è possibile controllare le schede di espansione collegate alla scheda.

I comandi che il sistema riconosce sono due, il primo comando è caratterizzato dalla lettera iniziale

[1] Per ulteriori informazioni su come programmare in C18 si rimanda al Tutorial "C18 step by step".

[2] Per una migliore identazione si faccia riferimento direttamente al file sorgente main.c.

T che sta per Trasmissione. I quattro byte che seguono il carattere T rappresentano rispettivamente il valore in esadecimale dei byte che devono essere scritti sulla PORTB e PORTD, partendo dal valore più significativo della PORTB e terminando con il valore meno significativo della PORTD.

Il secondo comando è invece individuato dalla lettera R che sta per Ricezione. I quattro byte successivi possono assumere un qualunque valore. Il PIC, alla ricezione di questo comando, risponde inviando il valore in esadecimale della PORTB e PORTD partendo dal valore più significativo della PORTB e terminando con il valore meno significativo della PORTD.

Vediamo in dettaglio i due comandi:

Comando di Trasmissione

T	B_H	B_L	D_H	D_L

dove :

B_H : rappresenta il valore esadecimale della parte più significativa di PORTB
B_L: rappresenta il valore esadecimale della parte meno significativa di PORTB
D_H : rappresenta il valore esadecimale della parte più significativa di PORTD
D_L : rappresenta il valore esadecimale della parte meno significativa di PORTD

Es.
Per accendere RB1 e RD3 il comando è: T0204

Comando di Ricezione

R	X	X	X	X

X può assumere qualunque valore diverso dal carattere nullo

Es.
Per richiedere lo stato del sistema il comando è: R0000
Il sistema risponderà con il valore della PORTB e PORTD. Dopo il comando dell'esempio precedente il sistema risponderà 0204

Come nel sistema Domotino affinché un comando venga riconosciuto è necessario che vengano inviati 5 byte. Questi devono essere inviati entro circa 4 secondi. Questo permette di filtrare eventuali disturbi derivanti da un ambiente rumoroso. Come verrà spiegato nel paragrafo GUI Controller è presente un interfaccia grafica ad hoc per mezzo della quale è possibile inviare comandi manualmente o interagire con il sistema direttamente con un semplice pannello di controllo, dimenticandosi dei comandi.

Programma Sorgente

```
1 /*****************************************************
2 *****************************************************
3
4 Autore : Mauro Laurenti
5 Versione : 1.0
6 Data : 02/10/2006
7 CopyRight 2006
8
9 Controllo di 16 Carichi con la seriale del PC
10
```

```
Visita www.LaurTec.com per ulteriori informazioni

*******************************************************
*******************************************************/

// variabili globali

unsigned char valPORTB = 0; //variabili associate alle porte di uscita
unsigned char valPORTD = 0;

//includo le librerie

#include <p18f4580.h>
#include <usart.h>
#include <timers.h>

#pragma config OSC = HS    // 20Mhz
#pragma config WDT = OFF   // disattivo il watchdog timer
#pragma config LVP = OFF   // disattivo la programmazione LVP
#pragma config PBADEN = OFF // disabilito gli ingressi analogigi sulla PORTB


//funzione per il ritardo
void Delay (int time)
{  int i;
   int j;

   for (j=0; j<time;j++)
         for (i=0; i<12000; i++);
}

//funzione per generare un beep con il cicalino
void beep ()
{  int i;

   PORTEbits.RE1 = 1;
   Delay (2);
   PORTEbits.RE1 = 0;
}



void Low_Int_Event (void); // prototipo di funzione


#pragma code low_vector=0x18

void low_interrupt (void)
{
   _asm GOTO Low_Int_Event _endasm //imposta il salto per la gestione

}

#pragma code

#pragma interrupt Low_Int_Event

void Low_Int_Event (void)
{ int i; // variabile generica
static unsigned char counter = 0; //conatore byte ricevuti per il comando
static unsigned char ResetUSART = 0; // Flag per controllare il Time out in ricezione
static unsigned char data [5]; // variabile che conterrà i dati ricevuti
```

```
73 static unsigned char num =0; // variabile per il conteggio degli interrupt
74 unsigned char tamp1 = 0; // variabili per la conversione carattere numero
75 unsigned char tamp2 = 0;
76 unsigned char tamp3 = 0;
77
78
79 if (PIR1bits.RCIF == 1 ) // Controllo che l'interrupt sia stato generato dall'USART
80 {
81    data[counter] = ReadUSART(); // leggo il dato dal buffer di ricezione
82
83    counter++; //incremento il numero di byte ricevuti
84
85 if (counter > 4) //quando ricevo 5 byte decodifico il comando ricevuto
86 {
87    for (i=0;i<5; i++)
88    {
89    WriteUSART(data[i]); // ritrasmetto il dato ricevuto
90    while (BusyUSART()); // attendo che il dato venga trasmesso
91    }
92    WriteUSART(13); // invio carattere ritorno a capo
93
94    if (data[0] == 'T') //riporto in uscita i valori letti
95    {
96          if (data[1]-48 > 9)
97                tamp1 = data[1]-55;     // ho una lettera
98          else
99                tamp1 = data[1]-48;
100
101         if (data[2]-48 > 9)
102               tamp2 = data[2]-55;     // ho una lettera
103         else
104               tamp2 = data[2]-48;
105
106         tamp3 = tamp1 << 4;
107         tamp3 = tamp3 | tamp2;
108
109         PORTB = tamp3;
110         valPORTB = tamp3;
111
112
113         if (data[3]-48 > 9)
114               tamp1 = data[3]-55;     // ho una lettera
115         else
116               tamp1 = data[3]-48;
117
118         if (data[4]-48 > 9)
119               tamp2 = data[4]-55;     // ho una lettera
120         else
121               tamp2 = data[4]-48;
122
123         tamp3 = tamp1 << 4;
124         tamp3 = tamp3 | tamp2;
125
126         PORTD = tamp3;
127         valPORTD = tamp3;
128   }
129
130   if (data[0] == 'R') //Lettura PORTB e PORTD
131   {
132 //leggo il valore di PORTB e lo converto in esadecimale
133
134   tamp1 = (valPORTB & 0b11110000);
```

```
      tamp1 = tamp1 >>4);
135   if (tamp1>9)
136         tamp2 = tamp1 + 55; //ho una lettera
137   else
138         tamp2 = tamp1 + 48;
139
140   WriteUSART(tamp2); // trasmetto i 4 bit più significativi del dato letto
141   while (BusyUSART()); // attendo che il dato venga trasmesso
142
143   tamp1 = (valPORTB & 0b00001111);
144
145   if (tamp1>9)
146         tamp2 = tamp1+ 55; //ho una lettera
147   else
148         tamp2 = tamp1 + 48;
149
150   WriteUSART(tamp2); // trasmetto i 4 bit meno significativi del dato letto
151   while (BusyUSART()); // attendo che il dato venga trasmesso
152
153 //leggo il valore di PORTD e lo converto in esadecimale
154
155   tamp1 = (valPORTD & 0b11110000);
      tamp1 = tamp1 >>4);
156   if (tamp1>9)
157         tamp2 = tamp1 + 55; //ho una lettera
158   else
159         tamp2 = tamp1 + 48;
160
161   WriteUSART(tamp2); // trasmetto i 4 bit più significativi del dato letto
162   while (BusyUSART()); // attendo che il dato venga trasmesso
163
164   tamp1 = (valPORTD & 0b00001111);
165
166   if (tamp1>9)
167         tamp2 = tamp1 + 55; //ho una lettera
168   else
169         tamp2 = tamp1 + 48;
170
171   WriteUSART(tamp2); // trasmetto i 4 bit meno significativi del dato letto
172   while (BusyUSART()); // attendo che il dato venga trasmesso
173
174   WriteUSART(13); // invio carattere ritorno a capo
175 }
176
177 counter = 0;
178 }
179 PIR1bits.RCIF = 0;
180 }
181
182
183 if (INTCONbits.TMR0IF == 1 ) // Controllo che l'interrupt sia stato generato da TMR0
184 {
185   num++;
186
187 if (num == 200)
188 {
189   if (counter > 0) // se non ricevo altri byte enro 4 secondi riazzero il
contatore byte
190   {
191         ResetUSART++;
192         if (ResetUSART = 100)
193         {
```

```
194         counter = 0;
195         ResetUSART = 0;
196         beep ();
197         }
198   }
199 }
200 INTCONbits.TMR0IF = 0;
201 }
202 }
203
204
205 void main (void)
206 {
207 TRISA = 0xFF;       // inizializzazione PORTA
208
209 TRISB = 0b00000000; // inizializzazione PORTB
210 PORTB = 0x00 ;
211
212 TRISC = 0b10100000; // inizializzazione PORTC
213 PORTC = 0x00;
214
215 TRISD = 0b00000000; // inizializzazione PORTD
216 PORTD = 0x00;
217
218 TRISE = 0x00;       // inizializzazione PORTE
219 PORTE = 0x00;
220
221 //inizializzo le varie periferiche interne al PIC
222
223 OpenTimer0 (TIMER_INT_ON & T0_SOURCE_INT & T0_16BIT ); //contatore uso generale
224
225 // Configura l'USART
226 // 8 bit
227 // 19200 bit/s
228 // 1 bit stop
229 // 0 bit parita'
230
231 OpenUSART( USART_TX_INT_OFF &
232 USART_RX_INT_ON &
233 USART_ASYNCH_MODE &
234 USART_EIGHT_BIT &
235 USART_CONT_RX &
236 USART_BRGH_HIGH,
237 64 );
238
239 INTCONbits.PEIE = 1 ;     // abilito interrupt per periferiche
240 INTCONbits.GIE = 1;       // abilito l'interrupt globale
241
242 beep ();      //segnalo l'attivazione del sistema
243
244 while (1);    //ciclo infinito in attesa d'interrupt
245 }
```

Integrazione di sistema

La scheda Freedom deve essere impostata per lavorare con la porta seriale RS232 come riportato nella documentazione della scheda. Il quarzo deve essere di 20 MHz e il PIC deve essere il PIC18F4580[3].

Le schede di espansione devono essere collegate sulle PORTB e PORTD impostando in maniera opportuna gli indirizzi delle singole schede come riportato nei relativi articoli delle schede di espansione. Le schede di espansione dovranno essere alimentate in maniera opportuna a seconda delle esigenze e rispettando i limiti delle schede stesse.

Se si dovesse far uso di un numero inferiore di schede di espansione, il sistema considererà comunque che tutte le schede sono connesse ma ovviamente si controllerà un numero inferiore di carichi.

[3] Facendo uso di un PIC differente il codice sorgente deve essere oppurtunatamente ricompilato.

GUI Controller

Per mezzo del Programma 16_Carichi, la cui schermata principale è riportata in Figura 2 è possibile inviare i comandi alla scheda di controllo facendo uso direttamente del Pannello di Controllo virtuale. In questo modo è possibile dimenticarsi della struttura stessa dei comandi e controllare il tutto in maniera più veloce.

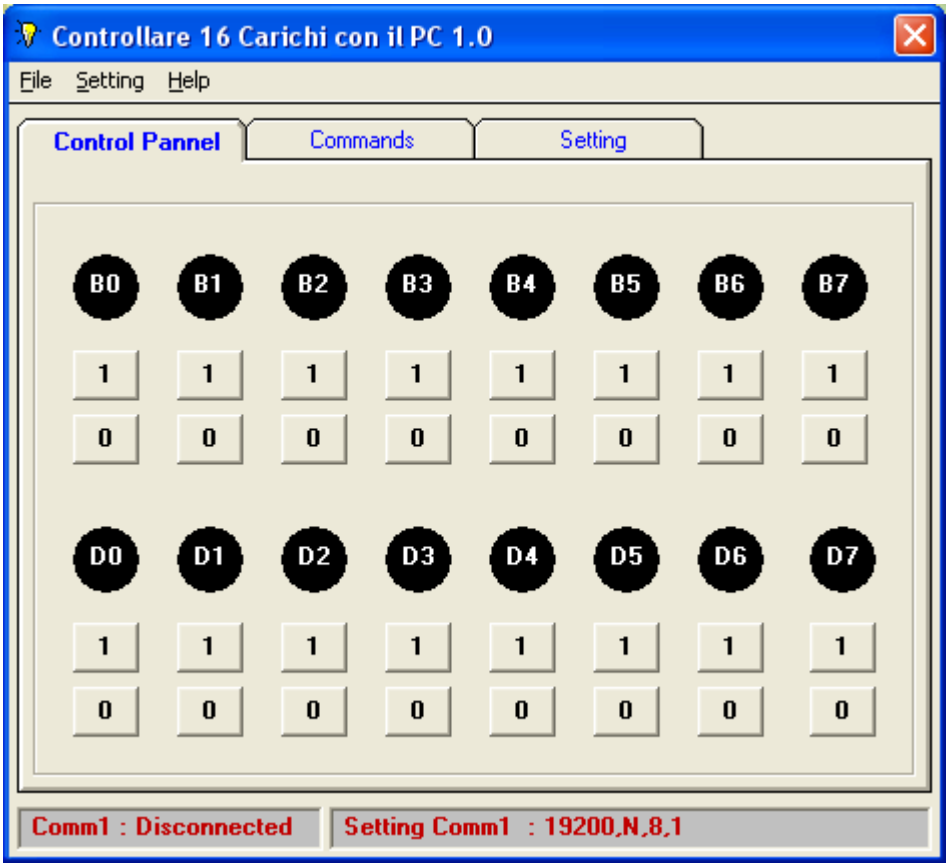

Figura 2: Schermata principale

Per chi volesse mantenere un controllo più diretto del sistema è possibile inviare anche comandi manuali facendo uso del Tab Commands. Vediamo in maggior dettaglio i vari Tab.

Per poter far uso della scheda di controllo è necessario che questa sia stata collegata al Computer prima di accenderlo. Dopo aver eseguito il Programma 16_Carichi è necessario impostare la porta seriale RS232 del PC facendo uso dei Parametri presenti nel Tab Setting (per la comprensione di tali parametri si rimanda alle spiegazioni che seguiranno[4]). Le impostazioni della porta di connessione e il suo stato sono riportate in rosso in basso alla finestra del programma stesso. Per poter iniziare la comunicazione tra il programma e la scheda è necessario aprire la porta di comunicazione[5] per mezzo della voce *Connect* del menù *Setting*.

Una volta che si attiva la comunicazione il programma invia lo stato dei carichi che è stato eventualmente impostato in precedenza facendo uso dei pulsanti di attivazione. Come possibile vedere dalla Figura 2 ogni carico connesso alla scheda è rappresentato da un cerchio nero. In particolare

[4] Per ulteriori informazioni sulla porta seriale si rimanda al Tutorial "Protocollo RS232" che è possibile scaricare dal sito www.LaurTec.com.

[5] Si raccomanda di aprire la porta di connessione solo se si è certi che la scheda è stata connessa alla porta seriale e le impostazioni sono corrette.

all'interno del cerchio è presente il nome del bit della porta rappresentato dal cerchio stesso. Per esempio PORTB0 è rappresentato da B0. Quando il cerchio è di colore nero vuol dire che il carico è spento mentre se il colore del cerchio è rosso vuol dire che il carico è stato attivato[6]. In Figura 3 è riportato un esempio di cerchi di colore rosso che individuano carichi attivati.

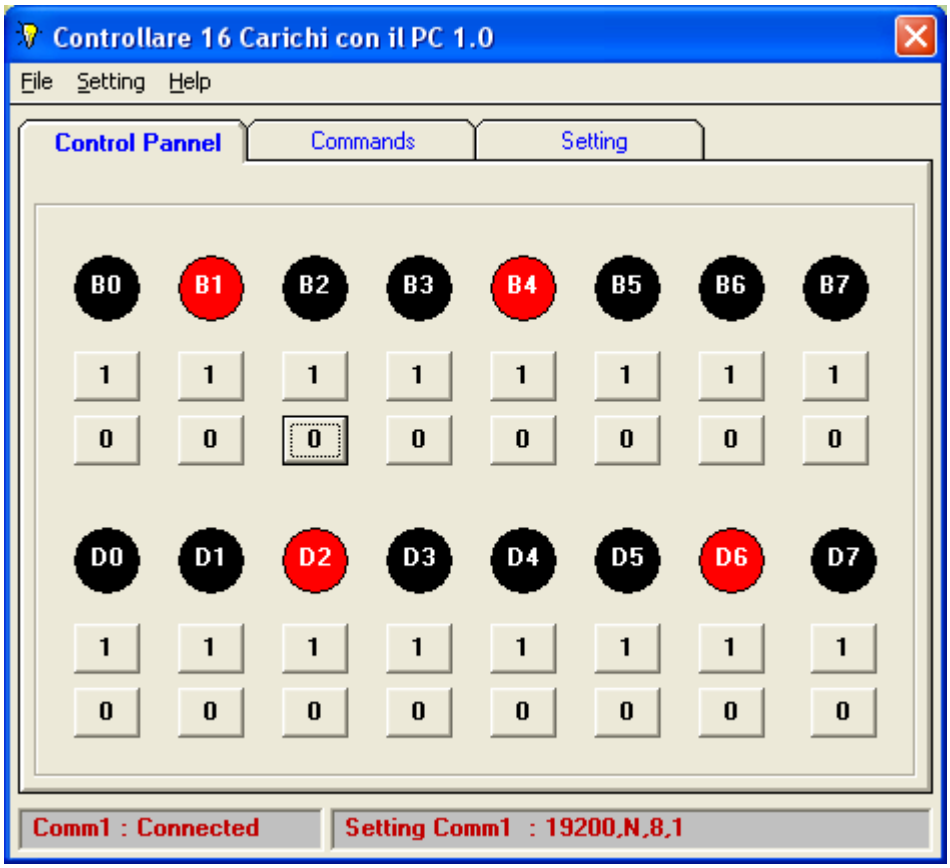

Figura 3: Colore rosso dei carichi attivi

Per attivare un carico bisogna premere il pulsante 1 posizionato sotto il cerchio che individua il carico stesso. Con la pressione del pulsante 1 viene inviato il comando relativo allo stato di tutti i carichi in modo da aggiornare lo stato della scheda. Per spegnere un carico precedentemente attivato è necessario premere il pulsante 0 presente sotto il carico che si vuole disattivare. Con la pressione del pulsante 0 viene inviato il comando relativo allo stato di tutti i carichi in modo da aggiornare lo stato della scheda. Quanto appena descritto per attivare e disattivare i carichi può essere effettuato anche quando la connessione della porta non è attiva. In questo caso il sistema e il programma non sono allineati. L'aggiornamento del sistema avverrà solamente all'attivazione della porta seriale. Per mezzo della voce *Turn OFF All* del menù *Setting* è possibile spegnere tutti i carichi con un solo Click; questa operazione corrisponde all'invio del comando T0000. Questo comando viene automaticamente inviato al PIC ogni volta che si chiude il programma. Qualora si dovesse perdere il controllo della scheda si raccomanda di premere il pulsante di Reset di Freedom in modo tale che il Firmware possa provvedere a spegnere tutti i carichi in maniera indipendente dal programma 16_Carichi.exe.

Si fa osservare che se le schede di espansione non dovessero essere 4, i bit della o delle porte non collegati saranno comunque delle uscite il cui valore sarà descritto dai cerchi colorati del pannello di

[6] Quando la connessione è ancora disattiva il colore Rosso non significa che il carico è effettivamente acceso ma soltanto che all'attivazione della porta seriale verrà inviato il comando per attivarlo. Si fa presente inoltre che il programma non controlla lo stato effettivo dei carichi, dunque se un comando non viene eseguito propriamente lo stato effettivo dei carichi e i colori dei cerchi potrebbe non coincidere.

controllo. Ciononostante dal momento che nessun carico è effettivamente collegato, un eventuale cerchio rosso avrà il solo significato che il bit della porta corrispondente vale 1.

Qualora si volesse controllare in maniera più diretta il sistema, è sempre possibile inviare i comandi manualmente. In questo modo è per esempio possibile controllare lo stato dei carichi per mezzo del comando R0000[7].

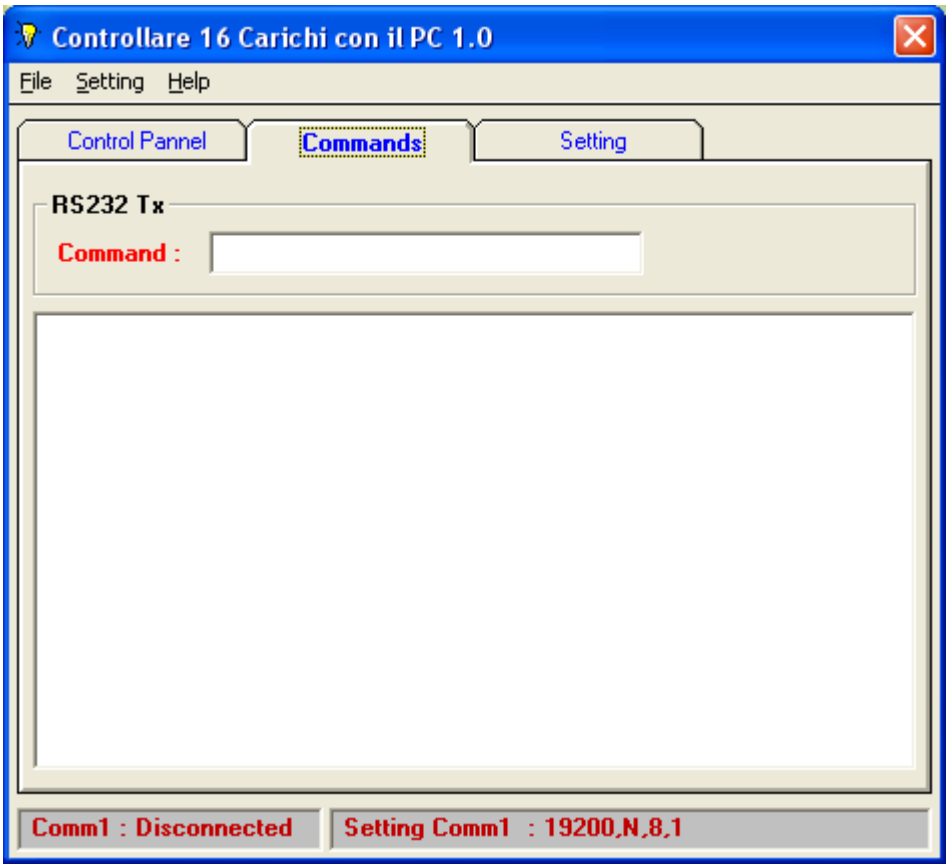

Figura 4: Tab per l'invio di Comandi

Il comando che si vuole inviare deve essere scritto nella casella di testo più piccola e per inviarlo è necessario premere invio. Nella casella di testo più grande sono riportati i dati ricevuti dal computer.
I comandi ricevuti da Freedom vengono ritrasmessi e dal momento che nella casella di testo più grande è riportato il testo ricevuto è possibile verificare se Freedom ha ricevuto il comando correttamente. Su questa casella di testo vengono riportati anche i comandi che Freedom ritrasmette quando si sta facendo uso del Tab Control Pannel.

Per mezzo del Tab Setting è possibile cambiare la porta seriale e le sue impostazioni. Una volta che il programma viene chiuso le impostazioni vengono salvate nel file Setting contenuto della directory dove è installato 16_Carichi.exe. Questo significa che se si collega il sistema sempre alla stessa porta seriale non ci si dovrà preoccupare molto delle impostazioni della porta seriale anche se una sbirciata all'angolo destro è sempre consigliabile al fine d'essere certi che le impostazioni siano corrette. Il Tab Setting è riportato in Figura 5.

[7] Nonostante sia possibile verificare lo stato dei carichi per mezzo dell'istruzione R0000 si ricorda che non è presente nessuna tecnica per la correzione di eventuali errori di comunicazione tra il PC e il sistema Freedom. Questo significa che per aumentare l'affidabilità del risultato è bene inviarne più di un comando. Se l'ambiente non è rumoroso il risultato del comando R0000 è in generale affidabile.

E' possibile osservare che è possibile cambiare tutte le impostazioni della porta seriale del PC. Questa possibilità discende dal fatto che è possibile utilizzare questo programma anche in altre applicazioni in cui siano richieste altre impostazioni. Le configurazioni della porta[8] per poter comunicare con il PIC devono essere come in Figura 5. Unico parametro che può cambiare è la variabile Delay Time che può assumere un valore compreso tra 1 e 2000.

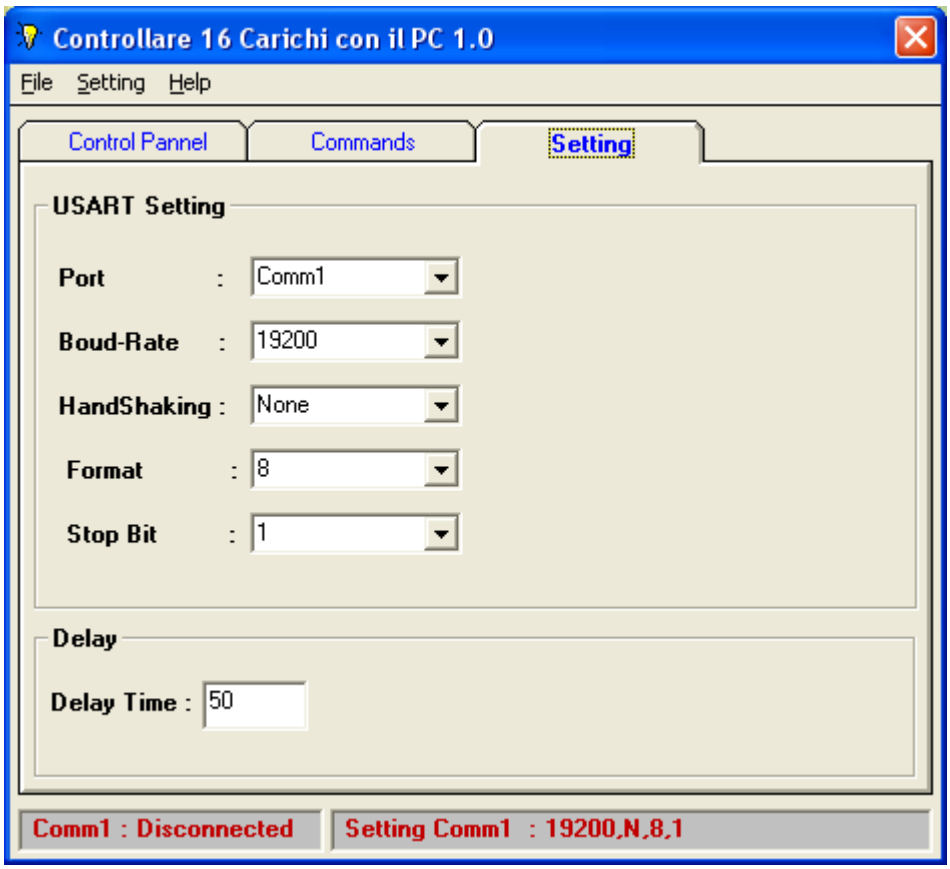

Figura 5: Schermata per l'impostazione della Porta Seriale del PC

Il valore può cambiare da computer a computer e deve essere scelto in maniera da far funzionare correttamente il programma. In particolare valori troppo piccoli possono causare il blocco del programma stesso o del sistema. Se si nota che i dati ricevuti dal sistema non riempiono in maniera opportuna le caselle di testo del programma 16_Carichi vuol dire che bisogna aumentare il valore di Delay Time. Valori alti hanno come effetti collaterali solo quello di rallentare la trasmissione e non quello di causare errori. Questo parametro non è usato nella versione 1.0 del programma 16_Carichi

Bibliografia

www.LaurTec.com : sito di elettronica dove poter scaricare gli altri articoli menzionati, aggiornamenti e progetti.

[8] Ogni volta che viene cambiato un parametro della porta, ad eccezione del Delay, la connessione viene interrotta ed è necessario riattivarla se si è certi che le impostazioni siano corrette. Qualora le impostazioni non siano supportate dal sistema la connessione non verrà attivata e si avrà una segnalazione d'errore.