programma di comunicazione

LINK-64 1.2

di Renato Giussani

Spesso si sente dire che per collegarsi via telefono a banche dati e servizi di ogni tipo gestiti via computer basta possedere un modem, oltre s'intende ad un computer.

Nulla di più falso! E ne sanno qualcosa i numerosi utenti di Commodore 64 che hanno dovuto imparare a loro spese quanto sia difficile riuscire a «partire» con l'hobby della telematica.

Io ero uno di quelli, ed ora che collegarsi ad MC-LINK è diventato solo un problema di linee occupate, posso finalmente raccontarvi come ho fatto.

Anzitutto ricordo agli interessati che desiderino chiarirsi meglio le idee su cosa è la telematica, che possono andare a leggere (o ri-leggere) i numerosi articoli di Corrado Giustozzi sull'argomento.

Ad uso e consumo dei più pigri ricorderò brevemente che esistono dei servizi di vario tipo, accessibili via computer, previo collegamento al telefono attraverso un modem, che consentono un rapido scambio di dati ed informazioni fra gli utenti ed il sistema, ed anche fra gli utenti stessi.

Uno di questi è il nostro MC-LINK, collegandosi al quale ci si può fare assegnare una casella postale (a cui gli altri utenti del sistema possono inviare messaggi privati) e si può partecipare in modo interattivo a conferenze sui più svariati argomenti.

Il modem

Sorvolando sui problemi burocratici e legali inerenti il possesso e l'uso di un modem, possiamo intanto convincerci che in sua assenza MC-LINK non può essere contattato se non «a voce» telefonando personalmente a Bo Arnklit!

Ecco che la Via Crucis dell'utente Commodore è già iniziata. Infatti, se è vero che di modem il mercato ormai abbonda, qual'è il più adatto e meno dispendioso? Senza voler entrare in particolari che esulano dagli scopi di questo articolo, diremo che esistono modem «Standard» e modem Commodore; quelli che abbiamo chiamato Standard sono i modem collegabili direttamente a qualsiasi computer dotato di connettore standard RS-232, gli altri hanno invece un connettore da inserire nel «User Port» (già, maschile, dato che «port» in inglese vuole dire «porto», di arrivo e di partenza dei segnali, e non già porta).

Con il C-64 si possono usare entrambi i tipi di modem, solo che l'uso di quello standard passa attraverso l'acquisto, o la costruzione, di una apposita scheda capace di trasformare i segnali dai ±12 volt in arrivo dal modem in ±5 volt e viceversa quelli a ±5 volt forniti dal Commodore in ±12 volt. Le magiche «Interfacce RS-232» che proclamano di effettuare una conversione di standard (quando non di protocollo) fra User-Port ed RS-232 hanno proprio questo scopo e rendono teoricamente (e praticamente) possibile il collegamento al C-64 di un modem standard.

Ricordate quindi che l'integrato responsabile della gestione dei segnali relativi al protocollo RS-232, nel C-64 c'è già ed è accessibile attraverso i contatti del User-Port. Ciò vuol dire che, un modem costruito per dialogare con il suddetto integrato al livello richiesto di ±5 volt funzionerà senza problemi e, grazie anche alla possibilità di trarre la sua alimentazione direttamente dal computer, costerà parecchio di meno.

Da qui il mio consiglio di acquistare la «interfaccia RS-232» per il Commodore solo se possedete già un modem «standard» o se pensate che vi vorrete divertire in seguito anche a fare esperimenti di collegamento diretto e scambio dati fra il C-64 ed un computer dotato di RS-232 (es.: C-64 con IBM-PC).

In ogni altro caso, ovvero per collegarsi ad MC-LINK, basterà un modem «dedicato» al C-64 capace di comunicare a 300 baud.

Ora che abbiamo acquistato il modem ed abbiamo effettuato il collegamento computer-modem-telefono, potremmo pensare a chiamare uno dei numeri che il Giustozzi tanto gentilmente ci ha fornito, MC-LINK compreso (06/4510211).

Poniamo i commutatori del modem su «Full» e «Originate» e componiamo il numero; dopo due o tre squilli dall'altra parte ci perviene un fischio molto «digitale» e noi, senza perdere un colpo, commutiamo il modem su «Data», ma... non succede niente.

Certo, la scena appena descritta non è del tutto verosimile, ma quanti di voi hanno *sempre* saputo che per «Comunicare» serve un «Programma di Comunicazione»?

L'interfaccia RS-232

Cominciando dal principio, possiamo immaginare di avere due computer dotati di connettori RS-232 sul loro pannello posteriore collegati attraverso l'apposito cavo.

I due computer hanno ora la potenzialità per scambiarsi dati, ma manca ancora la «volontà» di farlo realmente. Questa situazione non è molto diversa da quella in cui fra i due computer al posto del cavo poniamo due modem ed una linea telefonica; i modem servono solo a trasformare i segnali in uscita dai computer, che non potrebbero essere trasmessi come tali attraverso la linea telefonica, in altri capaci

Questo programma è disponibile su disco presso la redazione. Vedere l'elenco dei programmi disponibili e le istruzioni per l'acquisto a pag. 204.

Listato programma Link-64 1.2.

```
10 rem *** link64-2 r.giussani '86 ***:1956
20 close2:close3:close4:close15:clr:1303
30 open15,8,15,"i0":770
40 poke53280,0:poke53281,0:1101
50 printchr$(147)chr$(14)chr$(158)"LINK-64          ":print:2958
60 print"(C)1986 by Technimedia":print:print"R.Giussani":print:print:3822
70 print"Baud      = 300":print"Parola    = 8 bit":print"Stop bit = 1":3619
80 print"Parita'  = None":print"Modem     = 0":print"Mode      = Modem":print:4214
90 print"Variazioni    (s/n)":1620
100 a$="":geta$:ifa$<>"s"anda$<>"n"then100:2662
110 ifa$="n"then220:989
120 b=6:print:input"Baud       (300/600) ";a$:ifleft$(a$,1)="6"thenb=7:3547
130 p$="None":input"Parola         (8/7) ";a$:ifa$="7"thenb=b+32:c=96:p$="Even":5
056
140 print"Parita' (None/Even) : ";p$:2296
150 print"Stop bit           : 1":1603
160 open3,2,3,chr$(b)+chr$(c):1466
170 hm=16478:hb=int(hm/256):lb=hm-hb*256:rem non usare mai hm,hb,lb nel progr.:5
450
180 poke56,hb:poke52,hb:poke55,lb:poke51,lb:rem top basic 16478:3272
190 input"Mode  (Modem/Local) ";md$:ifleft$(md$,1)="l"thenmd=1:goto 250:4256
200 input"Modem         (0/1) ";mo$:ifmo$="1"thenpp=16:2987
210 goto250:498
220 open3,2,3,chr$(6)+chr$(0):rem 300 baud:8 bit:1 stop bit:par.none:3918
230 hm=16478:hb=int(hm/256):lb=hm-hb*256:rem non usare mai hm,hb,lb nel progr.:5
510
240 poke56,hb:poke52,hb:poke55,lb:poke51,lb:rem top basic 16478:3332
250 printchr$(14)chr$(147)chr$(158)"Attendere prego":b=16478:rem buffer 24k:4387
260 dimo%(255),i%(255):rv$=chr$(18):nr$=chr$(146):gi$=chr$(158):su$=chr$(145):43
03
270 rem ***  64<>ascii  ***:1159
280 forx=0to255:788
290 o%(x)=x:i%(x)=x:next:1377
300 forx=65to90:o%(x)=x+32:i%(x)=x+128:next:2812
310 forx=193to218:o%(x)=x-128:next:2049
320 forx=97to122:i%(x)=x-32:next:1948
330 o%(20)=8:604
340 i%(8)=20:608
350 c=0:r=24:gosub1580:1213
360 printrv$"Premi <"chr$(95)"> per menu' comandi          "nr$;:3440
370 gosub1620:printsu$"Per iniziare premi un tasto":2969
380 geta$:ifa$=""then380:1253
390 printchr$(147)"Collegamento attivato":2462
400 rem ***  routine input-output  ***:2224
410 geto$:ifo$=""then440:1308
420 ifo$=chr$(95)then550:1308
430 b%=asc(o$):o%=o%(b%):o$=chr$(o%):print#3,o$;:2728
440 get#3,i$:ifi$=""then410:1453
450 ifi$=chr$(19)thengosub1400:1512
460 ifi$=chr$(1)thenfb%=1:print" ":printgi$"Buffer aperto":3358
470 ifi$=chr$(26)andfb%=1thenfb%=0:bp=bp-1:print" ":printgi$"Buffer chiuso":4745
480 a%=asc(i$):i%=i%(a%):i$=chr$(i%):2267
490 ifa%=13thenprint" ";:1233
500 ifi%=34theni$=chr$(39):1617
510 printchr$(159)i$rv$" "nr$chr$(157);:1952
520 iffb%andbp<=40447-bthenpokeb+bp,i%:bp=bp+1:3097
530 ifbp=40448-bthenfb%=0:bp=bp-1:print:printgi$"Buffer chiuso":4025
540 goto410:316
550 rem ***  menu  ***:904
560 c=0:r=24:gosub1580:poke198,0:1631
570 printgi$rv$"B)uffer  D)isco  T)rasmiss.  R)icez.   "nr$;:3799
580 getk$:ifk$<>"b"andk$<>"d"andk$<>"t"andk$<>"r"then580:3828
590 ifk$="b"then640:963
600 ifk$="r"ork$="t"then700:1603
610 printsu$:printrv$"S)ave  L)oad  D)irectory  "chr$(95)")quit       "nr$;:4224
620 getk$:ifk$<>"s"andk$<>"l"andk$<>"d"andk$<>chr$(95)then620:4112
630 goto700:408
640 printsu$:printrv$"A)pri  C)hiudi  V)uota  S)tampa  "chr$(95)")quit"nr$;:4570
650 getk$:ifk$<>"a"andk$<>"c"andk$<>"v"andk$<>"s"andk$<>chr$(95)then650:4929
660 ifk$<>"s"then700:1225
670 printsu$:printrv$"V)ideo  S)tampante  "chr$(95)")quit            "nr$;:4019
680 getk$:ifk$<>"v"andk$<>"s"andk$<>chr$(95)then680:3410
690 gosub1640:gosub1620:print:goto1450:1532
700 gosub1640:gosub1620:print:1145
710 ifk$=chr$(95)then800:1337
720 ifk$="a"thenfb%=1:print"Buffer aperto":2678
730 ifk$="d"thengosub1660:1297
740 ifk$="c"thenfb%=0:print"Buffer chiuso":2699
750 ifk$="v"thenfb%=0:bp=0:pokeb,0:print"Buffer vuoto":3479
760 ifk$="s"thenprint"Saving Buffer":gosub810:2755
770 ifk$="l"thenprint"Loading Buffer":gosub900:2557
780 ifk$="t"thengosub1050:1101
790 ifk$="r"thengosub1290:1115
800 poke198,0:goto410:784
810 rem ***  save buffer  ***:1377
820 nf$="":input"Nome del file ";nf$:ifnf$=""thenreturn:2951
830 input"Seq. o Prg. (S/P) ";ft$:2083
840 open4,8,4,"@0:"+nf$+","+ft$+",w":2129
850 fori=0tobp:823
860 print#4,chr$(peek(b+i));:1266
870 if(i+1and127)=0ori=0thengosub1000:ifethenreturn:2723
880 next:245
890 close4:print:print"Salvataggio effettuato":return:2764
900 rem ***  load buffer  ***:1452
910 nf$="":input"Nome del file ";nf$:ifnf$=""thenreturn:3041
920 input"Seq. o Prg. (S/P) ";ft$:2173
930 open4,8,4,"0:"+nf$+","+ft$+",r":2150
940 i=0:474
950 get#4,a$:ifa$=""thena$=chr$(0):1896
960 s=st:if(i+1and127)=0ori=0thengosub1000:ifethenreturn:3299
970 pokeb+i,asc(a$):i=i+1:1690
```

(continua a pagina 208)

di pervenire senza (o con poche) alterazioni all'altro capo, dove l'altro modem si incaricherà della trasformazione inversa.

Torniamo quindi per semplicità al banale cavo e vediamo come si possono fare scambiare dati e messaggi fra i nostri due utenti.

Immaginando di voler spedire da un computer all'altro la parola «ciao», dovremo inviare i quattro caratteri (byte) «c», «i», «a», «o» all'integrato responsabile di spedirli sul cavo con le modalità previste dall'apposito standard RS-232.

Questo integrato è il 6526 che nel C-64 è collegato al User-Port e che viene chiamato CIA 2 (Complex Interface Adaptor n. 2). In realtà il CIA 2 va a cercare i caratteri da spedire in un apposita area di memoria (buffer di trasmissione) dalla quale li preleva in sequenza e li trasmette. Il Basic 2.0 del C-64 prevede i comandi adatti a stampare caratteri nel buffer di trasmissione, dopo avere adeguatamente aperto un canale RS 232 di comunicazione sul quale vanno inviati con le semplici istruzioni PRINT# .

Contestualmente all'apertura del canale RS 232 con l'apposito OPEN, si devono comunicare al CIA 2 anche i parametri secondo i quali si vuole che avvenga lo scambio di dati sul canale RS-232 (tra parentesi i valori di MC-LINK):

- Baud: velocità di trasmissione in bit/s (300).
- Word lenght: lunghezza parola (8).
- Stop bit: bit i stop (1)
- Parity: parità (None)
- Duplex: bidirezionale o unidirezionali (Full).

Effettuata, da programma, la operazione di apertura del canale (ad es. per MC-LINK: OPEN 3,2,3, chr$(6)+chr$(0)) non ci resta che comandare un PRINT#3, «ciao» ed avremo effettuato il trasferimento.

Dall'altra parte della linea il secondo computer dovrà preoccuparsi anch'esso di attivare la sua interfaccia, per poi leggere dal canale aperto i caratteri in arrivo e depositarli al sicuro da qualche parte, ad esempio in memoria. Nel caso che anche il ricevente fosse un C-64, dopo avere effettuato la stessa OPEN del canale si potrà andare a prelevare dal «buffer di ricezione» i caratteri arrivati e ricostruire la parola con una sequenza di get#3, i$:parola$ = parola$ + i$.

Note per la copiatura dei listati per il 64

Nel numero 44 (settembre 85) è stato pubblicato un programma di Checksum per aiutare i lettori nella copiatura dei listati per il Commodore 64 pubblicati sulla rivista.
Il funzionamento è il seguente:
— copiate il programma Checksum del numero 44 e salvatelo su disco o cassetta;
— per la successiva copiatura di un listato (con Checksum), caricate (dal vostro disco o dal vostro nastro) il programma di Checksum e fatelo partire; a questo punto potete copiare le varie linee del listato, compresi i due punti ed il numero che trovate alla

(segue da pagina 207)

```
980 ifs<>0thenclose4:bf%=0:bp=1-1:print:print"Caricamento effettuato":return:476
7
990 goto950:520
1000 rem ***  lettura errore disco  ***:2232
1010 ifi>0thenprint".";:1175
1020 input#15,e:602
1030 ifethenprint:print"Errore disco":close4:2080
1040 return:162
1050 rem ***  trasmissione  buffer  ***:2052
1060 input"Preparare (s/n) ";a$:1556
1070 ifa$<>"s"anda$<>"n"thenprintsu$su$:goto1060:2697
1080 ifa$="s"thenprint"Scrivi e premi "chr$(95)" per terminare":gosub2400:3984
1090 input"Protocollo trasm.: A)scii  X)modem";ft$:3209
1100 ifft$<>"a"andft$<>"x"thenprintsu$su$:goto1090:2900
1110 input"Tutto ok (s/n) ";ok$:1623
1120 if ok$<>"s"andok$<>"n"thenprintsu$su$:goto1050:2956
1130 ifok$="n"then return:1104
1140 print"Trasmissione in corso":print:2214
1150 ifft$="x"then2130:rem trasmissione x-modem:2855
1160 i=0:439
1170 o%=o%(peek(b+i)):1225
1180 o$=chr$(o%):849
1190 get#3,i$:ifi$=chr$(19)thengosub1400:1945
1200 if(i+1and127)=0thenprint".";:1740
1210 bl=abs(peek(669)-peek(670)):1816
1220 ifmdthen1240:850
1230 if(peek(56577)and16)=ppthenprint:print"Persa portante":i$=chr$(24):goto2250
:4473
1240 ifbl>128then1210:1196
1250 print#3,o$;:651
1260 i=i+1:ifi<=bpthen1170:1924
1270 print#3,chr$(26);:940
1280 print:print"Buffer trasmesso":return:1944
1290 rem ***  ricezione buffer  ***:1724
1300 input"Protocollo ricez.: A)scii  X)modem ";ft$:3186
1310 ifft$<>"a"andft$<>"x"thenprintsu$su$:goto1300:2849
1320 input"Tutto ok (s/n) ";ok$:1578
1330 if ok$<>"s"andok$<>"n"thenprintsu$su$:goto1320:2911
1340 ifok$="n"then return:1059
1350 print"Ricezione in corso":print:1920
1360 ifft$="a"thenfb%=1:return:1492
1370 goto1750:rem ricezione x-modem:1885
1380 ifbp=0thenprint:print"Ricezione abortita":return:2853
1390 print:print"Ricezione terminata":return:2264
1400 rem ***  x-on / x-off  ***:1433
1410 t=ti:554
1420 get#3,i$:if(ti-t)>600thenreturn:1871
1430 ifi$<>chr$(17)then1420:1509
1440 return:307
1450 rem ***  print buffer  ***:1601
1460 ifk$="s"thenopen4,4,7:cmd4:1604
1470 fori=0tobp:933
1480 geta$:ifa$=chr$(95)then1560:1704
1490 a%=peek(b+i):1079
1500 ifft$="p"orft$="x"thena%=a%and127:2458
1510 ifk$="v"anda%=34thena%=39:1930
1520 ifa%<32anda%<>13anda%<>10thena%=32:2777
1530 a$=chr$(a%):916
1540 ifk$="v"thenprintchr$(159)a$;:next:print:printgi$"Fine buffer":3261
1550 ifk$="s"thenprint#4,a$;:next:1362
1560 ifk$="s"thenprint#4:close4:1250
1570 poke198,0:goto410:789
1580 rem ***  print-at  ***:1196
1590 p1=peek(210):p9=peek(209):p2=peek(211):p4=peek(214):3175
1600 a=1024+r*40:ra=int(a/256):rb=a-ra*256:3172
1610 poke210,ra:poke209,rb:poke211,c:poke214,r:return:2279
1620 rem ***  restore-at  ***:1387
1630 poke210,p1:poke209,p9:poke211,p2:poke214,p4:return:2383
1640 rem ***  cancel-line  ***:1428
1650 printsu$:print"                                                    ";:return:2263
1660 rem ***  directory  ***:1378
1670 open2,8,0,"$0":y=6:printchr$(157)" ":1864
1680 i=0:gosub1000:ife<>0thenreturn:1820
1690 fort=1toy:get#2,xx$:next:ify=6thenprint" 0"rv$;:goto1710:3246
1700 get#2,c$,d$:printasc(c$+chr$(0))+256*asc(d$+chr$(0));:3241
1710 get#2,c$:ifstthen1740:1308
1720 ifc$<>""thenprintc$;:goto1710:1734
```

fine di ciascuna riga. Alla pressione del return, se la linea è stata copiata bene si può passare a copiare la successiva, altrimenti il programma di Checksum vi lascerà "inchiodati" sulla linea mal copiata obbligandovi a correggere l'errore prima di proseguire.

A quanto detto nel numero 44 riguardo al programma Checksum in questione, aggiungiamo che la routine di Checksum in LM si avvia con SYS 52480 mentre, in caso di arresto con Run-Stop/Restore, il restart si effettua con SYS 53072.

Attenzione: chi non vuole usare il Checksum, NON DEVE copiare i due punti e il numero alla fine delle righe, pena la segnalazione di "syntax error" da parte del computer.

```
1730 print" ":y=2:ifst=0goto1690:1763
1740 close2:print:return:831
1750 rem ***  ricezione x-modem  ***:1990
1760 bp=0:bl=1:fr=0:n=0:bl=40704:t=0:2801
1770 n=n+1:get#3,i$:ifn<128then1770:2234
1780 n=0:554
1790 iffr=0thenprint#3,chr$(21);:1629
1800 iffr=0andt<>0thenprint"R";:1852
1810 iffr<>0thenprint#3,chr$(6);:print".";:fr=0:2347
1820 n=n+1:ifmdthen1840:1302
1830 if(peek(56577)and16)=ppthenprint:print"Persa portante":goto2080:3582
1840 ifn=10thenprint:goto2080:1264
1850 t=ti:484
1860 get#3,i$:475
1870 ifi$=chr$(4)then2050:1209
1880 ifi$=chr$(1)then1910:1220
1890 if(ti-t)<600then1860:rem 10s:1853
1900 goto1790:461
1910 t=ti:p=peek(667):1298
1920 if(peek(667)=p)and((ti-t)<30)then1920:rem 0.5s:2912
1930 ifpeek(667)<>pthen1910:1528
1940 n=0:ck=0:885
1950 br=peek(bl+peek(668)):get#3,i$:1948
1960 bc=255-peek(bl+peek(668)):get#3,i$:2270
1970 if(br<>bc)or((br<>bl)and(br<>(bl-1)))then2080:3598
1980 ifbr=(bl-1)thenfr=1:goto1780:2052
1990 i%=peek(bl+peek(668)):get#3,i$:n=n+1:2561
2000 pokeb+bp,i%:bp=bp+1:ck=(ck+i%)and255:2861
2010 ifn<128then1990:1154
2020 cr=peek(bl+peek(668)):get#3,i$:2019
2030 ifcr=ckthenbl=(bl+1)and255:fr=1:goto1780:2953
2040 bp=bp-128:fr=0:goto1780:1890
2050 print#3,chr$(6);:650
2060 bp=bp-1:710
2070 goto1380:371
2080 rem ***  abbandono ricezione  ***:1951
2090 print#3,chr$(24);:bp=0:1168
2100 n=0:364
2110 n=n+1:get#3,i$:ifn<128then2110:2053
2120 goto1380:421
2130 rem ***  trasmissione x-modem  ***:2109
2140 bl=1:fr=0:n=0:op=0:n1=0:t1=600:2640
2150 n=n+1:get#3,i$:ifn<128then2150:2097
2160 n=0:424
2170 t=ti:ifmdthen2190:1262
2180 if(peek(56577)and16)=ppthenprint:print"Persa portante":i$=chr$(24):goto2250
:4403
2190 get#3,i$:ifi$=""and(ti-t)<t1then2190:rem 10s/1s:3045
2200 n=n+1:ifn=10thenprint:print"Trasmissione abortita":return:3749
2210 ifi$=""then2170:1033
2220 ifi$=chr$(21)andfr=1thenn1=n1-128:print"R";:3012
2230 ifi$=chr$(21)then2300:1359
2240 ifi$=chr$(6)then2280:1331
2250 ifi$=chr$(24)thenprint:print"Trasmissione abortita":return:3504
2260 iffrthenprint"NAK";:n1=n1-128:goto2300:2768
2270 goto2170:569
2280 ifop>=(b+bp)thenprint#3,chr$(4):print".":print"Buffer trasmesso":return:421
3
2290 bl=bl+1:print".";:1315
2300 bl$=chr$(bl):bc$=chr$(255-bl):ck=0:t1=60:3089
2310 print#3,chr$(1);bl$;bc$;:1115
2320 op=b+n1                                  :1813
2330 ifop>(b+bp)theno%=0:goto2350:1879
2340 o%=peek(op):773
2350 o$=chr$(o%):print#3,o$;:n1=n1+1:1932
2360 ck=ck+o%)and255:1225
2370 if(n1and127)<>0then2320:1521
2380 print#3,chr$(ck);:fr=1:1250
2390 goto2160:433
2400 rem *** editor ***:1051
2410 ifbp>0thenbp=bp+1:1481
2420 printrv$" "nr$chr$(157);:1274
2430 geta$:ifa$=""then2430:1309
2440 ifa$=chr$(95)thenbp=bp-1:print:return:2221
2450 ifbp<=40447-bthenpokeb+bp,asc(a$):printa$;:bp=bp+1:3535
2460 ifbp=40448-bthenbp=bp-1:print:printgi$"Buffer chiuso":return:3913
2470 goto2430:513
```

Alla fine dell'operazione, effettuando sul computer ricevente un PRINT parola$ vedremo sullo schermo il nostro «ciao», arrivato sano e salvo.

Il programma di comunicazione

Normalmente, nelle comunicazioni fra computer si richiede di effettuare qualcosa di molto più complesso del semplice invio di una parola, che nel nostro esempio abbiamo fra l'altro inviato ad un computer già pronto a riceverla.

Questo è il motivo per cui, per rispondere al meglio a tutte le esigenze che possono nascere durante un collegamento fra computer, vengono di norma utilizzati opportuni programmi di comunicazione. Questi sono capaci di gestire la apertura e chiusura del collegamento, variarne i parametri, interagire con le periferiche e trattare i caratteri scambiati secondo opportuni protocolli di comunicazione in grado di consentire l'invio e la ricezione di dati nelle condizioni di velocità e/o affidabilità richieste.

Per consentire un facile scambio di messaggi e programmi fra i due computer, che avevamo collegato direttamente via cavo, occorre dunque dotare entrambi dei rispettivi programmi di comunicazione ed effettuare i trasferimenti secondo le modalità ed utilizzando i comandi specifici dei programmi usati.

Tornando all'ipotetico utente che aveva collegato computer, modem e telefono, e che in assenza del programma non era riuscito a collegarsi ad MC-LINK, abbiamo visto che alla risposta (fischio) del modem di MC aveva prontamente commutato il modem da «Tel» a «Data» ed in assenza del programma di comunicazione non era successo nulla.

Ora, con il programma in esecuzione, sul suo schermo comparirebbe la classica scritta di intestazione di MC-LINK seguita dalla richiesta del suo numero di codice, in assenza del quale dovrebbe digitare un punto interrogativo seguito dal <RETURN> per rispondere alla procedura di primo collegamento.

È bene che fin d'ora vi abituate ad un concetto molto importante con il quale vi troverete sempre a che fare durante le vostre sperimentazioni telematiche:

l'operatore di un computer collegato ad un sistema remoto lavora contemporaneamente con due programmi su due sistemi.

A - Il programma di comunicazione

installato sul vostro computer accetta ed esegue sia i comandi specifici necessari per gestire il collegamento sia una serie di operazioni «accessorie» di utilità (es.: mostrare la directory del disco presente nel drive o caricare il file da spedire).

B - Contemporaneamente, almeno mentre siete in «Modo Terminale», ovvero state inviando caratteri al sistema remoto, sarete in grado di dare comandi al programma che «gira» su quest'ultimo.

Ad esempio, un utente Commodore, usando il suo Vip Terminal o LINK-64, potrà inviare al nostro computer «remoto» i comandi necessari per «pilotare a distanza» il programma di MC-LINK scritto da Bo Arnklit in Turbo Pascal e che attualmente gira su un PC-XT.

Il LINK-64

La decisione di scrivere un programma di comunicazione originale per Commodore 64 risale ad alcuni mesi fa, quando Bo ed io stavamo sperimentando il collegamento Commodore-IBM per effettuare il trasferimento di un mio programma Commodore (CROSS-64) al Toshiba TI-1100, sul quale avrei in seguito lavorato alla versione per IBM (CROSS-PC).

Risolto abbastanza rapidamente il problema contingente, ci rendemmo conto che, nonostante il numero elevato di programmi di terminale disponibili (peraltro noti per lo più solo agli addetti ai lavori) il C-64 era abbastanza abbandonato a se stesso sul fronte dei programmi di comunicazione «veri», con possibilità di gestione di una working area, del drive, di comunicazione X-Modem e via discorrendo. L'unico programma di comunicazione veramente degno di questo nome che conoscevamo era il Vip-Terminal della Softlaw Corp. (132 Aero Camino, Goleta, CA 93117 USA - Tel.: 805/968-4364), ma la sua notevole lunghezza nonché le difficoltà che si incontrerebbero ad effettuare qualsiasi intervento di «personalizzazione» non contribuivano certo a considerare il problema completamente risolto; fra l'altro, la prima release del Vip non prevedeva l'X-Modem, necessario per effettuare lo scambio di programmi Commodore. Il programma LINK-64 che presentiamo non tenta nemmeno da lontano di fare concorrenza al sofisticatissimo Vip (in particolare perché nella versione odierna non prevede ancora la trasmissione e ricezione «diretta» da disco), ma gode in cambio di notevoli vantaggi pratici, ormai ben noti ai numerosissimi utenti che, dopo averlo «scaricato» da MC-LINK con il Vip, hanno deciso di eleggerlo a loro mezzo di comunicazione preferito per l'uso quotidiano.

Il listato che presentiamo non pretende di non essere migliorabile, ed anzi la sua stesura completamente in Basic è stata voluta proprio per facilitare qualsiasi intervento di modifica. Il rovescio della medaglia è una intrinseca lentezza che viene risolta solo con la compilazione con il Pet-Speed, *assolutamente indispensabile.*

Chi avrà la pazienda di digitarlo e non possieda il Pet-Speed (altri compilatori non vanno bene) potrà comunque utilizzarne la opzione di ricezione X-modem per scaricare da MC-LINK la versione già compilata.

Per i meno volenterosi vi è poi la possibilità di acquistare il LINK-64 dalla Technimedia già registrato su disco, al costo di 30.000 lire (il disco contiene sia la versione sorgente sia quella compilata).

Il LINK-64 compilato occupa in memoria circa 14 kbyte (54,5 blocchi da 256 byte), il che permette di riservare ad area temporanea di lavoro (buffer) circa 24 K di memoria (circa 94 blocchi da 256 byte); ricordate che se vorrete operare delle modifiche al programma sorgente dovrete poi compilarlo, verificare la quantità di memoria occupata, modificare di conseguenza nelle righe 170, 230 e 250 il valore 16478 delle variabili «hm» e «b» al nuovo valore consentito (superiore di almeno 500 al numero fornito da «print peek (46)*256 + peek(45)» effettuato dopo avere caricato la nuova versione compilata) ricompilare il tutto. Ovviamente il buffer verrà ridotto di tanti byte quant'è la differenza fra il nuovo valore e 16478.

Il collegamento con l'esterno avviene in vero ASCII, ovvero i codici Commodore vengono convertiti sia in uscita che in ingresso. Ciò significa che dentro al C-64 si troveranno sempre codici-carattere Commodore e fuori sempre ASCII, tranne che durante i trasferimenti in X-Modem, che permettono di ricevere ed inviare file esattamente «come sono» (ovvero anche i codici di controllo, non utilizzabili durante i trasferimenti in ASCII) elliminando anche totalmente il problema dei disturbi di linea. Fra le utility del programma vi sono fra l'altro:

— Selezione dei parametri di collegamento: Baud (300/600), Parola (8/7), Parità (Nome/Even), Mode (Modem/Local), Modem (0/1). La parità viene posta automaticamente None con parola di 8 bit e Even con la selezione 7 bit. Lo stop bit è sempre 1 ed il collegamento è sempre Full duplex. I modem 0 e 1 differiscono nel test della portante; se durante un trasferimento X-Modem ricevete subito il messaggio «Persa Portante» dovete cambiare l'operazione.

— Modo terminale: trasmissione e ricezione dirette, da tastiera e su schermo.

— Scrittura diretta da tastiera nel buffer: per preparare brevi messaggi «on line».

— Load: carica nel buffer, da disco, il contenuto di un file, SEQ o PRG che sia.

— Save: salva il contenuto del buffer in un file su disco a scelta.

— Stampa del buffer, su schermo o su carta.

— Trasmissione del buffer con protocollo X-Modem o ASCII.

— Ricezione del buffer con protocollo X-Modem o ASCII.

— Apertura e chiusura del buffer manuali ed automatiche.

— Cancellazione del buffer.

— Lettura della directory del disco (tratta da un programma di G.L. Rutigliano).

L'uso del programma è abbastanza intuitivo; ricordate però che non tutte le operazioni sono «Idiot Proof», ovvero a prova di errore, e che per passare dal Modo Terminale al Modo Comando, nel quale il LINK-64 accetta i suoi comandi senza inviare nulla all'esterno, basta premere il tasto freccia a sinistra.

Il listato che pubblichiamo è stato stampato con il set di caratteri Maiuscolo/Minuscolo, che su schermo si ottiene premendo contemporaneamente i tasti Commodore e SHIFT. I due punti ed il numero che compaiono alla fine di ogni riga dovranno essere digitati solo se avrete attivato la routine di Checksum di MC; le righe più lunghe di 80 caratteri possono essere memorizzate sia utilizzando i comandi Basic abbreviati (vedi manuale C64) sia evitando di digitare i due punti ed il numero di checksum: la riga viene comunque memorizzata.

Se vorrete utilizzare il LINK-64 non compilato, ricordate che la sua lentezza non gli consente di ricevere correttamente messaggi troppo lunghi (in modo terminale) e che la ricezione X-Modem da MC-Link del LINK-64 compilato richiede più di venti minuti!.

Ricordate che i nuovi utenti alla richiesta del codice devono rispondere «?» e premere <RETURN>, seguire tutte le istruzioni, e non potranno scaricare programmi o inviare messaggi (tranne che a MC0001) fino a quando non saranno abilitati.

Buon collegamento. MC

apricot XEN
MULTI-USER

CENTRO COMUNICAZIONE

Il sistema Apricot Xen MULTI-USER consente di immagazzinare tutti i vostri dati sui Mainframes Apricot, potenti file servers da 20, 40 o 100 Mb sotto sistema operativo MS-DOS MULTI-TASKING.
Ogni mainframe è dotato di un processore 80286 con una memoria RAM di 2 Mb, un'unità di back up a nastro (stream tape) da 20 Mb ed un F.D.D. da 720 Kb (3,5"). Può pilotare sino a 32 stazioni di lavoro.
Ogni stazione di lavoro Xen workstation è dotata di un processore 80286, una memoria Ram di 1 Mb, un monitor Apricot paper white o colori, la tastiera Xen da 102 tasti ed il microscreen Apricot.
Possono essere collegati in rete sino a 10 mainframes per un totale complessivo di 64 stazioni di lavoro.
Apricot Xen Multi-user non si limita ad operare in ambiente MS-DOS. Un mainframe può anche operare sotto Xenix pilotando sino a 16 terminali.
Grazie al software NETBIOS, Xen Multi-user è compatibile con le reti PC NET e TOKEN RING dell'IBM®.
Qualsìasi personal computer Apricot, IBM® o compatibile, può essere utilizzato come stazione di lavoro nell'ambito del sistema.
Il completo software in dotazione comprende: MS-DOS MULTI-TASKING al mainframe, MS-DOS 3.2 alle stazioni di lavoro, MS NET, MS WINDOWS, NETWORK MANAGER, Remote Diagnostics, NET BIOS, GEM VDI e AES (Gem application support) e le applicazioni MS Windows, MS Write, MS Paint.

DITRON Distribuzione esclusiva Ditron S.p.A. - Viale Certosa, 138 - 20156 Milano - Tel. 02/3085645-3538441