VIC da zero +64

di Tommaso Pantuso

Prove di trasmissione

Dovendo continuare la nostra chiacchierata sull'RS 232 del Vic e del 64 lasciata in sospeso nello scorso numero, andiamo oggi ad illustrare qualche semplice esempio di trasmissione di informazioni tra questi due computer collegandoli tramite la user port che all'occasione si trasforma in porta d'ingresso-uscita RS 232.

L'hardware per il collegamento

La volta scorsa abbiamo parlato abbastanza estesamente di come avviene la trasmissione seriale di dati. Abbiamo descritto i due pseudo registri dell'UART simulata via software nel Vic e nel 64 corredando la descrizione con tabelline riassuntive e con qualche esempio teorico. Oggi passiamo alla pratica con qualche semplice esperimento per mezzo del quale potremo "vedere" le cose descritte in teoria e renderci conto dei problemi ad esse connesse. Premettiamo che i programmi d'esempio che presenteremo non hanno altra pretesa se non quella di aiutare la comprensione dei fatti esposti.

Prima di esaminare come aprire un canale RS 232 per la trasmissione o per la ricezione è il caso di spiegare, per chi volesse dilettarsi in qualche esperimento, in che modo collegare tra loro due dei computer in questione per realizzare una interfaccia a tre linee che è appunto la sola che prenderemo in considerazione in questa sede. Per le differenze tra lo standard vero e quello utilizzato di fatto dalla Commodore rimandiamo all'articolo precedente.

Nella figura 1 potete osservare la funzione svolta dalle linee d'uscita della user port del Vic o del 64 quando questi operano in modo RS 232 mentre in figura 2 sono riportate le corrispondenze con un connettore standard RS 232.

L'interfaccia che vogliamo realizzare per collegare due computer è composta solo da un cavo e da due connettori 12+12, cioè non è presente alcuno stadio che manipola i livelli elettrici (come descritto la volta scorsa) perché operando tra due Commodore tali livelli sono compatibili.

Per una trasmissione a tre linee abbiamo bisogno di:

una linea di massa (A o N);

una linea per la ricezione (B e C collegati insieme);

una linea per la trasmissione (M).

Lo schema del collegamento è illustrato nella figura 3. Per prima cosa vengono collegate insieme le masse di CN1 e CN2 collegando la linea A alla linea N; la linea M di CN1, cioè quella di trasmissione dei dati viene collegata all'insieme C-B del connettore CN2 che per quest'ultimo corrisponde alla linea di ricezione e lo stesso avviene per la linea C-B di CN2 che viene connessa all'ingresso M di CN1. Dopo queste sem-

USCITA	USER PORT VIC	USER PORT 64	FUNZIONE
A	GND		GND
B	CB1]	RECEIVED DATA	FLAG 2
C	PBØ]		PBØ
D	PB1	REQUEST TO SEND	PB1
E	PB2	DATA TERMINAL READY	PB2
F	PB3	RING INDICATOR	PB3
H	PB4	RECEIVED LINE SIGNAL	PB4
J	PB5		PB5
K	PB6	CLEAR TO SEND	PB6
L	PB7	DATA SET READY	PB7
M	CB2	TRANSMITTED DATA	PA2
N	GND		GND

Figura 1 - Tabella che riporta la configurazione delle linee d'uscita sulla user port del Vic e del 64 quando i due computer sono in modo RS 232.

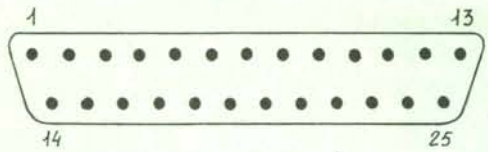


Figura 2
Il connettore standard RS 232 e la corrispondenza tra i suoi pin e quelli della user port del Vic o del 64.

USER PORT CBM	CONNETTORE STANDARD	SEGNALE
A	1-7	GND
B		
C	3	RD
D	4	RTS
E	20	DTR
F	18	RI
H	8	DCD
J		n.c.
K	5	CTS
L	6	DSR
M	2	TD
N	1-7	GND

plici operazioni il cavo è pronto per essere utilizzato.

N.B. Quando i due computer sono connessi correttamente, i pin dei connettori su cui sono stati saldati i fili devono appartenere alla seconda fila, quella più in basso, guardando il connettore dall'alto.

Apertura di un canale

Come abbiamo accennato già la volta scorsa, i comandi più importanti usati per la comunicazione RS 232 sono:

OPEN per aprire il canale;

PRINT# per trasmettere un dato;

GET# o INPUT# per ricevere dall'altra parte lo stesso dato.

La corretta sintassi di apertura è la seguente:

```
OPEN nf,2,0 CHR$(ctrl) + CHR$( cmd).
```

nf è un numero preferibilmente compreso tra 1 e 128;

ctrl rappresenta il numero che vogliamo scrivere nello pseudo registro di comando.

Sull'uso di questi registri rimandiamo all'articolo precedente. Vi ricordiamo solo che tramite il numero in essi contenuto vengono stabilite tutte le modalità sulla trasmissione.

Supponiamo di voler trasferire dei dati da un computer (ad esempio un Vic) in modo half duplex con ciascuna parola inviata composta da otto bit più due bit di stop ed alla velocità di 2400 baud senza un controllo di parità. Se queste sono le condizioni volute, dovremo porre ctrl = 138 e cmd = 16 nel comando di apertura. Il programma completo per trasmettere è il seguente:

```
10 OPEN 2,2,0, CHR$(138) + CHR$(16)
20 GET A$ : PRINT A$ : IF A$ < > "" THEN
H$ = H$ + A$
30 IF A$ = CHR$(13) THEN PRINT#2,H$:
H$ = ""
40 GOTO 20
```

La linea 10 apre il canale RS 232 con le modalità descritte; la 20 attende l'inserimento del primo carattere e compone la parola man mano che si inseriscono i caratteri da cui essa è formata; la 30 provvede all'invio della stringa quando viene premuto il tasto <RETURN>; la 40 riavvia il ciclo.

Dall'altra parte, cioè sul computer ricevente, dovremo inserire un programma che sia capace di prelevare l'informazione in arrivo sulla porta RS 232. Un programma atto allo scopo può essere composto dalle seguenti linee fondamentali:

```
10 OPEN 2,2,0, CHR$(138) + CHR$(16)
20 GET#2,A$; : PRINT A$
30 GOTO 20.
```

Il significato delle varie linee è abbastanza evidente: la linea 10 apre il canale in ricezione, in maniera identica al programma precedente cioè con velocità, questa volta di ricezione, di 2400 baud, parole di otto bit ecc.; la linea 20 contiene l'istruzione GET# tramite la quale è possibile ricevere la parola in arrivo che poi viene impressa sullo schermo con PRINT A$. Si tenga presente che per la ricezione può essere impiegato anche un INPUT#; la differenza fra quest'ultimo ed il GET# è che mentre il GET # riceve la parola in arrivo considerando come variabile ciascun carattere da cui è composta la parola stessa, con è considerata una variabile l'intero blocco trasmesso. In altre parole se inviamo i caratteri che compongono la parola MC, con essi vengono ricevuti come due caratteri separati mentre con INPUT# il blocco MC è considerato nella sua interezza.

Per finire, la linea 30 rimette la macchina in attesa.

Una volta capito come avvengono separatamente la ricezione e la trasmissione, facciamo un passo avanti scrivendo un segmento che permetta di utilizzare il computer contemporaneamente sia come ricevitore che come trasmettitore in maniera automatica ponendo la macchina in attesa quando non riceve ma sempre pronta a trasmettere. Il segmento è molto semplice ed è riportato nel listato A a fianco del quale, nella figura 4, riportiamo il diagramma a blocchi che ne permetterà più semplicemente la comprensione.

Se quanto illustrato fino ad ora vi è chiaro, possiamo fare un ulteriore passo avanti nella nostra chiacchierata.

Trasferiamo un programma da 64 a Vic 20.

Compiamo ora un'operazione più difficile (si fa per dire) trasferendo un programma dalla memoria del 64 a quella del Vic. Per tale scopo utilizzeremo sul primo computer un programma di sola trasmissione e sul secondo uno di sola ricezione. I due programmi che svolgono queste funzioni sono riportati nei listati B e C. Come potete facilmente osservare le istruzioni di trasmissione e di ricezione sono uguali a quelle precedentemente illustrate; cambia solo il contenuto del registro di controllo che da 138 diventa 136 modificando così la velocità di scambio che non avviene più a 2400 baud ma a 1200 baud.

Prima di andare avanti è bene ricordare

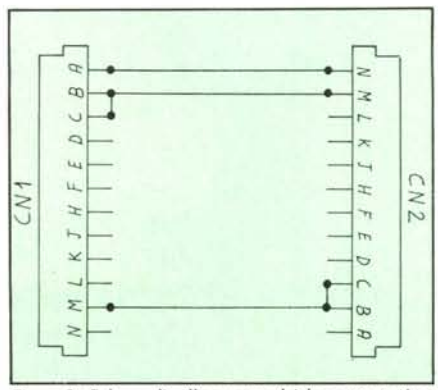


Figura 3 - Schema di collegamento dei due connettori per realizzare l'interfaccia tra i due computer.

un fatto molto importante già ampiamente discusso precedentemente in questa stessa rubrica. Un programma nella memoria di un Vic o di un 64 viene suddiviso a blocchi ciascuno dei quali rappresenta una linea di programma. Ogni blocco viene concatenato per mezzo di un link posto in testa al blocco stesso. In altre parole, se la prima linea occupa in memoria uno spazio che va ad esempio da 4096 a 4126, all'inizio del primo blocco troveremo un link che punta alla locazione 4127 da cui inizia il secondo blocco e così via. In virtù di questo fatto, se trasferiamo un programma in Basic dal 64 al Vic dovremo adottare un piccolo accorgimento. Infatti un programma scritto per un Vic in configurazione base inizia dalla locazione 4097 per cui se vogliamo trasferirlo in un 64 perché esso giri correttamente dovremo far si che esso venga allocato a partire dalla locazione 4097 e ciò si ottiene mettendo gli opportuni valori nei puntatori di inizio programma sul 64, questo perché in un semplice trasferimento di memoria i vari link non vengono riadattati. Naturalmente potremmo mettere a punto un programma che effettui la conversione dei link rilocando automaticamente il programma trasferito ma di ciò non ci occuperemo. Viceversa se vogliamo trasferire un programma dal 64 al Vic 20 e poi vogliamo che esso giri correttamente su quest'ultima macchina, dovremo scrivere il programma sul 64 a partire dalla locazione 4097 spostando preventivamente i puntatori di inizio programma altrimenti esso comincerebbe naturalmente da 2049.

Detto ciò eseguiamo il nostro semplice esperimento spostando un programma dal 64 al Vic.

Per prima cosa dobbiamo collegare le due macchine con il cavo che abbiamo costruito ed accenderle. Poi dobbiamo caricare il programma "RX SINCRONO" (listato C) sul Vic e quello chiamato "TX SINCRONO" (listato B) sul 64. Questi

```
1 REM * TX/RX RS 232 *
10 OPEN 2,2,0,CHR$(138)+CHR$(32)
20 GET#2,A$:PRINTA$;
30 GETA$:PRINTA$;:IF A$<>""THEN H$=H$+A$
40 IF A$=CHR$(13)THEN PRINT#2,H$;:H$=""
50 GOTO20
```

Listato A - Programma ricevitore/trasmettitore in RS 232.

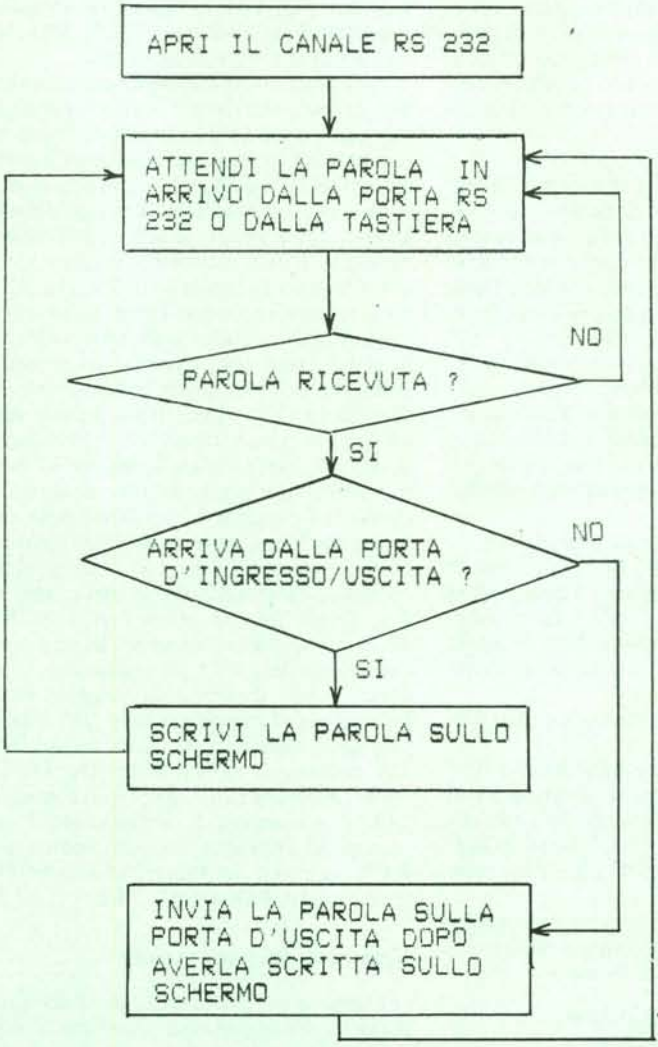


Figura 4 Diagramma di flusso che illustra in dettaglio le funzioni svolte dal programma del listato H.

```
10 REM * TX SINCRONO *
20 OPEN2,2,0,CHR$(136)+CHR$(16)
30 GETA$:IFA$=""THEN20
40 FORI=4096TO4191
50 A=PEEK(I):PRINTA,I
60 GOSUB100
70 NEXT
80 END
100 PRINT#2,STR$(A):RETURN
```

Listato B - Questo programma trasmette i contenuti delle locazioni da 4096 a 4191.

```
10 REM * RX SINCRONO *
20 OPEN2,2,0,CHR$(136)+CHR$(16)
30 PRINT"OK"
40 FORI=4096TO4191
50 GOSUB200
60 POKEI,VAL(A$):PRINTA$,I
70 NEXT
80 END
200 INPUT#2,A$:IFA$=""THEN200
210 RETURN
```

Listato C - Questo programma riceve i contenuti delle locazioni da 4096 a 4191 provenienti da un computer e li memorizza tra gli stessi estremi nella propria memoria.

programmi andranno allocati in entrambe le macchine in una zona in cui non subiranno interferenze con altri programmi e sceglieremo per tale scopo una parte abbastanza alta della memoria disponibile (n.b. — il Vic è considerato senza espansioni di memoria). A questo punto possiamo esaminare passo passo tutte le operazioni da svolgere.

Per primo occupiamoci del Vic:

1) — Per prima cosa spostiamo l'inizio del programma Basic a partire da 6657 ponendo il numero 26 nella locazione 44 con POKE 44,26. Nella locazione 43 è contenuto 1. Ricordiamo che i contenuti delle locazioni 43 e 44 puntano l'inizio del programma; facciamo una verifica: 1 + 26*256 = 6657, questo risultato indica che le nostre operazioni sono corrette.

2) — Eseguiamo POKE 6656,0. Questo passo è necessario perché il sistema deve trovare sempre uno 0 nella locazione immediatamente precedente all'inizio effettivo del programma.

Lo 0 opera appunto da segnalatore.

3) — A questo punto possiamo scrivere o caricare da nastro o disco il programma "RX SINCRONO" e dare il <run>, operazione che metterà la macchina in attesa facendo comparire sullo schermo la scritta "OK".

Esaminiamo ora le operazioni da compiere sul 64:

1) — Spostiamo l'inizio del Basic a 4097 ponendo il flag di inizio programma in 4096 (a proposito, i numeri sono espressi tutti in decimale) con POKE 44,16: POKE 4096,0 (nella locazione 43 è già contenuto un 1);

2) — Possiamo ora scrivere il programma da trasferire che ad esempio può essere:

```
10 PRINT"<shift + clr/home>"
20 FOR I=1TO10
30 PRINT "MICROCOMPUTER"
40 FOR T=1TO100: NEXTT
50 NEXTI
60 GETA$: IF A$="" THEN60
70 GOTO 10
```

3) — Annotiamo i contenuti delle locazioni 45 e 46 ottenuti con PRINT PEEK(45), PEEK(46) che nel nostro caso sono 94 e 16 ed indicano la fine del programma.

4) — Prima di scrivere il programma "TX SINCRONO" dobbiamo fare in modo che esso vada ad allocarsi a partire da 39937 (per nostra scelta) ed in ogni caso che occupi una posizione tale da non subire interferenze con il programma da trasmettere. Ciò sarà ottenuto con POKE 44,156: POKE 39936,0.

N.B. — Gli estremi del ciclo FOR... NEXT nei due programmi sono tali da trasferire solo l'area interessata. Il primo estremo è ovvio perché è il punto da cui inizia il programma da trasferire mentre il secondo può essere facilmente ricavato con PRINT(PEEK(45) + PEEK(46)*256) - PEEK(43) + PEEK(44)*256)

Compiute tutte le operazioni descritte non ci resta che dare il <run> anche al programma del 64. La macchina si metterà in attesa della pressione di un tasto, evento che avvierà il trasferimento durante il quale vedremo impressi sullo schermo del 64 i numeri della locazione letta ed il valore decimale in essa contenuto, il quale viene di volta in volta trasferito al Vic, che scorrono velocemente verso l'alto. Sullo schermo collegato all'altro computer vedremo la stessa cosa, solo che in questo caso i contenuti delle locazioni indicano ciò che di volta in volta viene effettivamente memorizzato in esse. Dopo un certo numero di secondi, quando l'elaborazione ha avuto termine non ci resterà altro da fare che ripescare il programma trasferito nella memoria del Vic. Questo è facilmente ottenibile risistemando i puntatori in maniera corretta all'inizio ed alla fine del programma stesso. Intanto effettueremo POKE 44,16 che sposterà l'inizio del Basic a 4097 (nella locazione 43 è già contenuto 1). A questo punto ricordandoci i valori letti nelle locazioni 45 e 46 del 64 dopo aver caricato il programma da trasferire (erano 94 e 16) digiteremo in modo diretto POKE 45,94 : POKE 46,16. Dopo ciò il comando LIST ci permetterà di verificare che il trasferimento è avvenuto correttamente e con RUN vedremo imprimersi sullo schermo la scitta MICROCOMPUTER.

Commenti ed osservazioni

Esaminiamo più in dettaglio i due programmi. Consideriamo per primo RX, cioè quello di ricezione.

La linea 20 apre il canale con le modalità che ormai conosciamo.

La 30 imprime sullo schermo il messaggio di pronto a ricevere; la 40 avvia il ciclo e la 50 invia alla subroutine 200-210 che pone il computer in attesa di un carattere ed "inchioda" la macchina a tale subroutine finché non viene ricevuta la prima stringa, dopo di che il programma ritorna alla linea 60 in cui la stringa ricevuta viene trasformata in un numero che viene depositato nella locazione corrispondente al numero d'indice del ciclo FOR...NEXT.

Il programma TX, dopo aver aperto il canale di I/O, si mette in attesa della pressione di un tasto e se ciò si verifica viene avviato il ciclo che legge il contenuto delle locazioni interessate, scrive sullo schermo il numero di locazione ed il suo contenuto (linea 50) ed invia alla subroutine della linea 100 che trasforma il numero letto in una stringa per inviarlo in uscita.

Facciamo qualche osservazione sui due programmi.

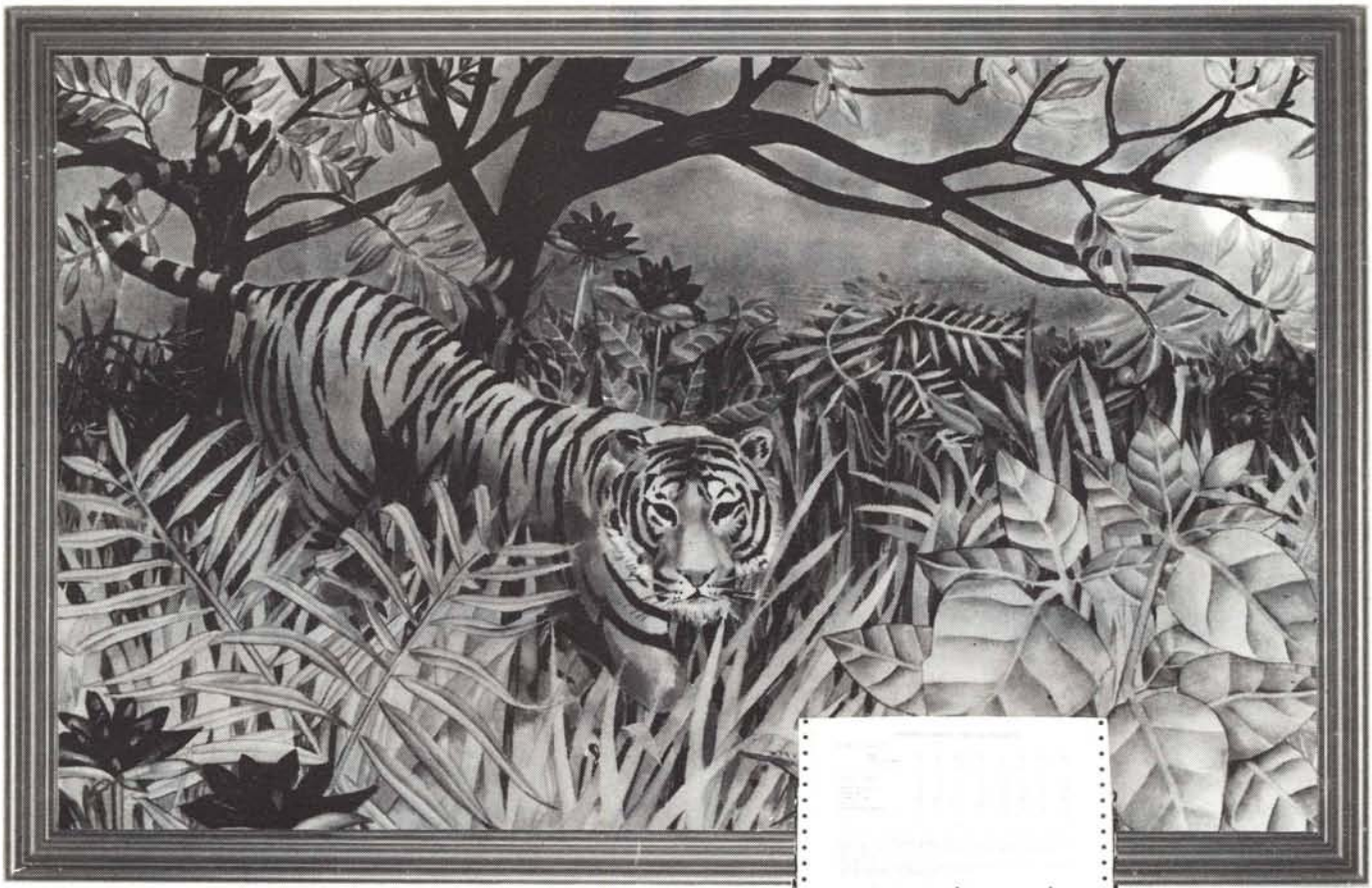
Durante l'elaborazione, come già detto vedremo scorrere sullo schermo di entrambi i computer i numeri delle locazioni interessate ed il loro contenuto. La trasmissione avviene sostanzialmente in questo modo: il dato viene posto sulla porta d'uscita e ivi mantenuto per un certo tempo dopo di che esso viene ritirato per lasciar posto al dato successivo. Se durante il tempo di permanenza del dato sulla porta il ricevitore per una qualunque ragione non fa in tempo a "catturarlo", esso viene perduto.

Per verificare ciò mentre i numeri scorrono sugli schermi tenete premuto sul Vic che sta ricevendo il tasto CTRL: vedrete che lo scorrere dei numeri sullo schermo di quest'ultimo computer rallenterà evidenziando un modo di ricezione più lento mentre su quello dell'altra macchina non noterete alcuna variazione. Ciò significa che mentre il trasmettitore manda i dati il ricevitore non è sempre pronto a riceverli tutti ed il trasmettitore non perde tempo ad aspettare che dall'altra parte ci sia disponibilità alla ricezione.

Se invece compirete la stessa operazione dall'altra parte, cioè sul 64, aumentando l'intervallo tra l'invio successivo di due dati tenendo premuto il tasto CTRL, lo scorrere dei numeri diventerà più lento su entrambi i teleschermi. Questo non deve meravigliarvi perché il programma ricevitore contiene un controllo (IF A$="" THEN 200) che lo tiene in attesa impedendogli ulteriormente l'elaborazione se nessun dato viene inviato dall'altra parte. La linea in questione in pratica è come se dicesse: "se stai ricevendo la stringa nulla che equivale per te al non ricevere nessun dato rimani in attesa alla linea 200 se no continua".

L'inconveniente descritto (se inconveniente si può chiamare) può essere eliminato sia utilizzando uno scambio a più linee introducendo delle linee di handshake, sia sincronizzando la ricezione e la trasmissione per mezzo dell'invio di caratteri di controllo che avvertono che il dato è stato memorizzato e che quindi si può procedere all'invio del successivo. È quest'ultimo il sistema che noi utilizzeremo e... sarebbe bello illustrarlo tra queste pagine ma lo spazio non ci è molto amico, per cui vi diamo appuntamento al prossimo numero nel quale continueremo i nostri esperimenti. MC

LA TIGRE È IN AGGUATO



State cercando una stampante per il vostro micro:

Deve essere facile da usare (manuale in italiano, selezione dei parametri da pannello e memorizzazione permanente).

Deve essere multifunzione e permettervi di passare dalla qualità listing (180 cps.) alla qualità lettera per il trattamento testi.

Deve essere facilmente interfacciabile ed immediatamente compatibile con il vostro micro... qualunque esso sia.

Deve essere lo strumento per riprodurre in modo perfetto i vostri grafici.

Deve essere molto affidabile, avere una probabilità di guasto solo ogni 18 mesi ed essere ciononostante supportata da una rete nazionale di assistenza postvendita.

Deve far parte di una gamma completa e compatibile (80 - 132 colonne, grafica, colore, inserimento del foglio singolo manuale e automatico, caratteri scientifici e APL...).

Deve sempre inserirsi nei vostri limiti di spesa e soddisfare le vostre esigenze odierne e future.

Deve essere pensata, messa a punto, prodotta e commercializzata dal PIÙ GRANDE COSTRUTTORE MONDIALE INDIPENDENTE DI STAMPANTI.

LA VOSTRA SCELTA È FATTA

SERIE SPG 8000 "PAPER TIGER"

Dataproducts

DATAPRODUCTS s.r.l.
Via Vincenzo Monti, 8 - 20123 MILANO - Tel. 3452211-860347

DEDICHIAMO IL NUOVO POCKET COMPUTER CASIO® PB-700 AL MINISTRO GORIA.

Prima di tutto perchè un uomo come lui, perennemente impegnato a 'dare numeri' nel corso di riunioni, consultazioni, conferenze stampa, ecc. ha bisogno di uno strumento di calcolo e gestione finanziaria, potente e flessibile.
E soprattutto portatile.
Poi perchè come ministro italiano ha proprio diritto ad uno dei primi, completi manuali di programmazione per personal computer tascabili in lingua italiana. E ancora perchè, il PB-700 CASIO, espandibile a 16 KB, è uno dei pocket più potenti oggi disponibili. E con le cifre che l'onorevole tratta questo è essenziale.
Un display a 4 colonne per 20 caratteri permette di evitare errori di impostazione o lettura e fornisce grafici descrittivi di grande chiarezza.
Mentre un libro di programmi fornirà un utile repertorio di software utilizzabile per risolvere qualsiasi problema.
Il PB-700, è dotato come accessorio a richiesta di registratore con microcassette, per non perdere mai un dato per strada, e di una fantastica stampante grafica plotter a 4 colori. Uno strumento di immediata evidenza per le analisi dei dati, che consentirà al nostro simpatico ministro - finalmente - di mettere un po' di colore in mezzo a tanti grigi calcoli. Lo invitiamo, anzi, fin d'ora a Milano presso i nostri uffici: saremo lieti di consegnargliene uno dimostrandogli di persona tutti i pregi del PB-700.

CASIO®

Gioielli della microinformatica.

DITRON
DISTRIBUTORE NAZIONALE
SpA

Viale Certosa, 138 Milano - Tel. 02/3085645 (5 linee ric. aut.)