a cura di Maurizio Mauri

I componenti di ingresso/uscita

Eccoci qui a parlare dei componenti di I/O del sistema MSX. I tre moschettieri dell'MSX sono, come ormai tutti sanno, il PPI 8255, tripla porta parallela, il PSG AY 3-8910 generatore sonoro a 3 canali, il VDP TMS 9918, processore video. Il quarto è D'Artagnan, pardon, lo Z80A, che a dire il vero è la CPU. PPI, PSG e VDP sono visti dalla CPU come dispositivi di I/O. Ora vedremo il PPI e i dispositivi ammessi, come la tastiera anche dal punto di vista soft.

di Sergio e Dario Neddi

Il PPI 8255

Programmable Peripheral Interface

Il PPI 8255 è un componente programmabile della famiglia 8080, quindi un po' anzianotto, ma tutt'ora molto usato. Ha tre porte di I/O parallele (di 8 bit) programmabili in ingresso o in uscita individualmente, più un registro di controllo. Nell'MSX, questo componente si occupa della gestione dei banchi di memoria, dell'uscita seriale verso il registratore, del comando motore del registratore, del led CAPS LOCK, del click sonoro della tastiera (per questa funzione non viene impiegato il PSG) e della tastiera stessa.
Il componente ha tre modi di funzionamento, ma poiché due modi, l'1 ed il 2 richiedono una diversa configurazione hardware, a noi interessa particolarmente il modo 0. E solo di questo parleremo.
Le tre porte disponibili sono: la A, la B e la C. Quest'ultima può essere spezzata in 2 porte a 4 bit, che possono venire programmate indipendentemente una dall'altra in ingresso od in uscita. Le due porte a 4 bit si chiamano CL (dal bit 0 al 3) e CH (dal bit 4 al bit 7). Per programmare una porta in ingresso od in uscita bisogna configurare il componente inviando una parola di controllo opportuna all'apposito registro di controllo (vedi tabella) e poi leggere (o scrivere) il dato indirizzando la porta desiderata con una normale istruzione di I/O. Nella tabella A pubblichiamo le parole di controllo.

Tabella A

Control Word (hex)	Port A	Port B	Port CH	Port CL
80	out	out	out	out
81	out	out	out	in
82	out	in	out	out
83	out	in	out	in
88	out	out	in	out
89	out	out	in	in
8A	out	in	in	out
8B	out	in	in	in
90	in	out	out	out
91	in	out	out	in
92	in	in	out	out
93	in	in	out	in
98	in	out	in	out
99	in	out	in	in
9A	in	in	in	out
9B	in	in	in	in

Tabella B

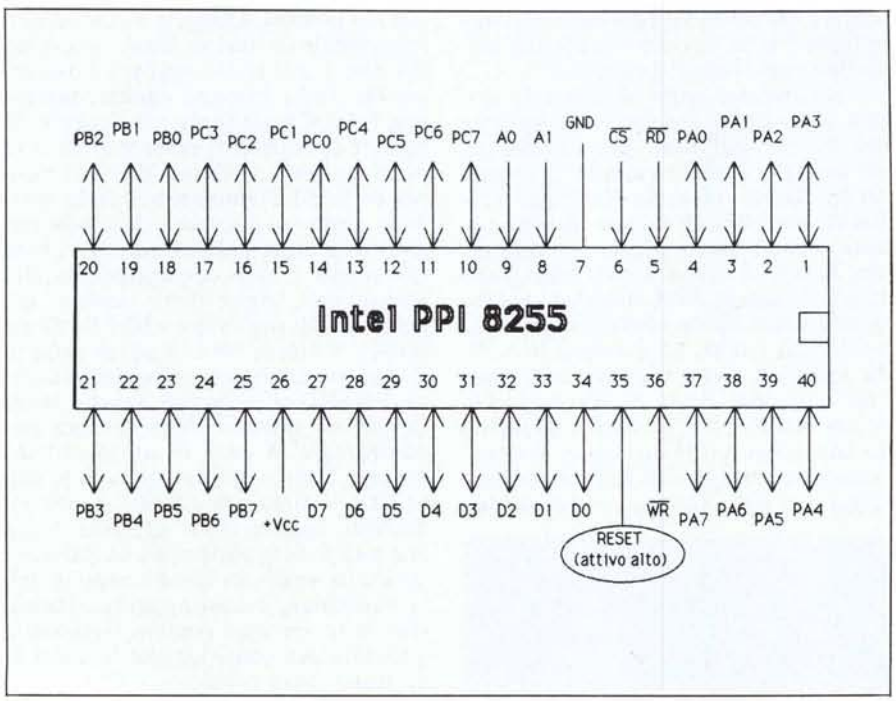
Indirizzi di I/O (hex)	Mansioni
A8: Port A	Commutazione dei banchi di memoria (uscita)
A9: Port B	Scansione tastiera (ingresso)
AA: Port C	Port CH: bit 7: click audio della tastiera (uscita) bit 6: led CAPS LOCK (uscita) bit 5: uscita seriale per registratore bit 4: controllo motore registratore (uscita) Port CL: bit 0-3: scansione tastiera (uscita)
AB: Control Port	

Nella tabella B gli indirizzi di I/O delle porte ed i loro compiti.
Come si vede la configurazione hardware implica che l'8255 venga impiegato sempre nella configurazione con Port A in uscita, Port B in ingresso e Port CH e CL in uscita, pertanto si potrà intuire che la programmazione del Control Port sarà sempre fissa. Dalla tabella dei codici di controllo si può ricavare il codice, che è 82H. Notate che il bit più significativo delle parole di controllo presenti in tabella è sempre ad 1. Questo bit, se posto ad 1, dà infatti alla parola il significato di definizione di modo, ed a seconda degli altri bit, configura le porte come nella tabella di programmazione vista sopra. Viceversa se il bit 7 è a 0 allora il componente si trova nel modo BIT SET/RESET, cioè ogni bit della porta C (e solo della C) può venire settato o resettato indipendentemente dagli altri. Per la programmazione più di tante parole vale la tabella C (ricordiamo che questi codici vanno inviati al Control Port).
Per provare quanto esposto basta un'istruzione Basic: con OUT & HAB,8 si accende il motore del registratore,

Tabella C

Codici (hex) di controllo	Effetto
00	resetta il bit 0
01	setta il bit 0
02	resetta il bit 1
03	setta il bit 1
04	resetta il bit 2
05	setta il bit 2
06	resetta il bit 3
07	setta il bit 3
08	resetta il bit 4
09	setta il bit 4
0A	resetta il bit 5
0B	setta il bit 5
0C	resetta il bit 6
0D	setta il bit 6
0E	resetta il bit 7
0F	setta il bit 7

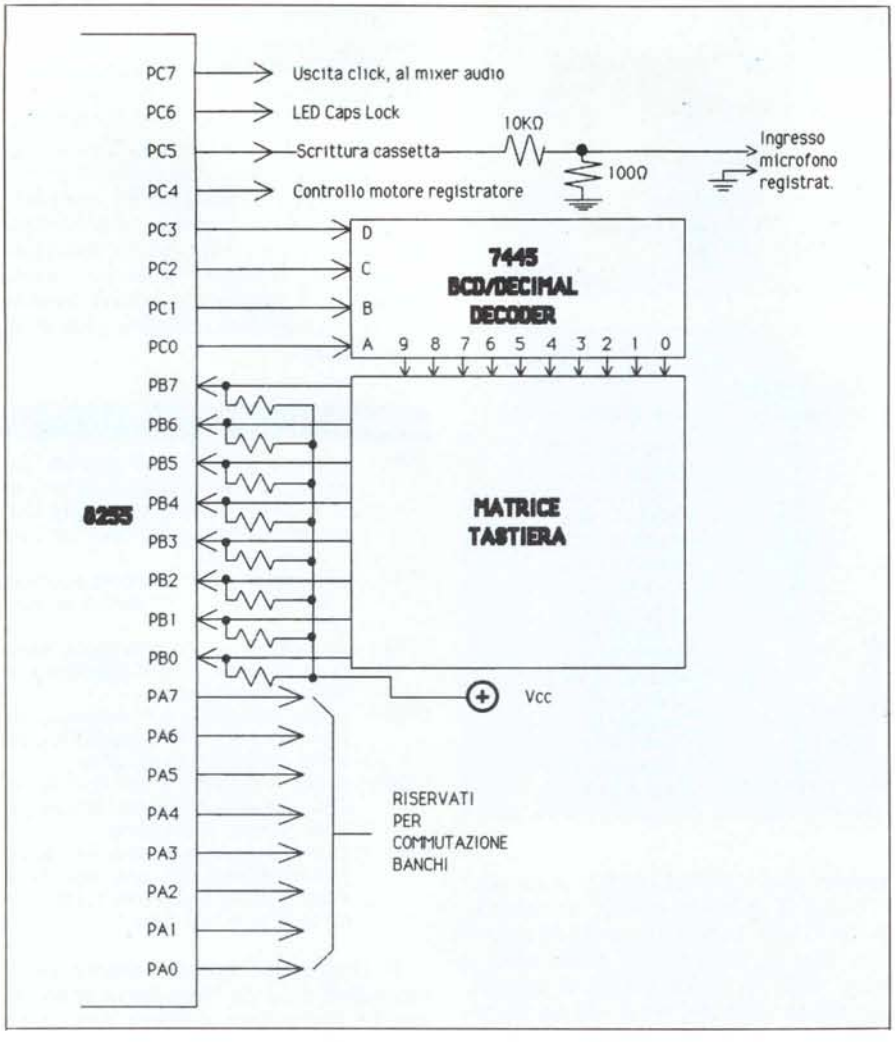
mentre con: OUT & HAB,9 si spegne; con: OUT & HAB, & H0C si accende il led CAPS LOCK (si accende solo il led, non si bloccano le maiuscole) e con: OUT & HAB,&H0D si spegne. Da ciò si vede che sia il led sia il motore si accendono quando i rispettivi bit sono a 0 e viceversa si spengono quando sono ad 1. La logica è inversa probabilmente per esigenze hardware. Attenzione: non crediate di poter leggere dal Control Port una parola appena introdotta con PRINT INP (&HAB): infatti il registro di controllo è a sola scrittura e l'operazione di lettura ritornerà un valore senza senso. Comunque è sconsigliabile accedere direttamente a questo componente perché è molto facile che la macchina si inchiodi inviando qualche parola sbagliata. È meglio accedervi tramite le routine del Bios. Eccone alcune di interessanti:



INDIRIZZO (HEX)	FUNZIONE
00F3	equivale all'istruzione MOTOR del Basic: se il registro A contiene 0 allora OFF, se contiene 1 allora ON, se contiene un valore con il bit 7 ad 1 allora inverte.
0132	se il registro A è a 0 allora accende il led CAPS LOCK, sennò lo spegne.
0135	se il registro A è a zero allora resetta il bit relativo al click della tastiera, sennò lo mette ad 1.
0136	legge nel registro A il contenuto del port A del PPI, cioè la parola di controllo dei banchi di memoria.
013B	scrive il registro A nel Port A del PPI, cioè scrive la parola di controllo dei banchi di memoria.
0141	se si pone nel registro A un codice corrispondente ad una riga della matrice della tastiera la routine ritorna in A la riga letta.

È chiaro che queste routine sono sfruttabili solo da linguaggio macchina. In particolare occorre una certa pratica per manipolare la routine di commutazione dei banchi: un errore significa inchiodare il computer.

Parleremo un'altra volta della gestione dei banchi di memoria. Intanto ci può interessare come l'8255 gestisce la tastiera: i 4 bit meno significativi dei port C sono collegati ad un circuito integrato denominato BCD/Decimal decoder, siglato 7445 (nel nostro caso) od equivalente. Questo componente provvede a portare a livello logico 0 una delle 10 uscite di cui è fornito (normalmente stanno a livello logico 1) a seconda del contenuto degli ingressi. Come conseguenza anche le righe della tastiera (che stanno collegate alle uscite del 7445) seguono le vicissitudini delle uscite di questo componente. Insomma, se il PPI gli manda 4 bit a 0000 allora viene messa a 0 l'uscita numero 0 e viene quindi selezionata la riga 0 della matrice della tastiera; se il PPI gli manda 0010 allora viene messa a 0 l'uscita 2 (infatti 0010 in binario equivale proprio a 2) e quindi selezionata tale riga della tastiera. La lettura vera e propria dello stato dei tasti associati a tale riga viene fatta leggendo il port B dei PPI: ad esso, infatti, sono collegate le colonne



relative alla matrice della tastiera, con delle resistenze di PULL-UP (3300 ohm nello SVI 728, ma non sono valori critici) che servono a forzare a livello logico 1 gli ingressi dei port B quando nessun tasto è premuto: così solo i tasti premuti sulla riga selezionata possono venire letti come bit a 0 e quindi identificati. Nella matrice della tastiera le righe tra 0 e 8 sono quelle effettivamente usate da tutti gli MSX. Le righe 9 e 10 servono invece per alcune versioni che dispongono di tastierino numerico secondo lo schema di tabella D, dove i tasti opzionali possono essere usati per qualunque scopo (normalmente per gli operatori aritmetici).

Fa eccezione a quella norma lo SVI728 che usa solo 4 tasti della riga 9 per i segni di operazione, mentre gli altri tasti sono la replica dei corrispondenti della tastiera principale ai quali sono collegati elettricamente.

Notiamo, infine, che alcuni programmi che leggono direttamente la tastiera (di solito i giochi) non testano le righe 9 e 10, per cui il tastierino numerico non viene riconosciuto.

Vediamo ora come sfruttare le nostre conoscenze (anche se divagheremo un po' dall'hard per «infilarci» nel soft) per leggere un tasto rimanendo in Basic; infatti le istruzioni INKEY$ ed INPUT$ dicono quando un tasto viene pigiato, ma se noi volessimo sapere quando viene rilasciato? Potendo leggere direttamente la tastiera potremmo anche rilevare la pressione di tutti i tasti, ad esempio lo shift da solo non viene rilevato dalle normali istruzioni Basic. Come fare? Di primo acchito viene in mente un metodo che, come vedremo, ha un difetto. Il metodo consiste nell'indirizzare nella matrice della tastiera la riga contenente il tasto da rilevare e leggerla direttamente (vedere il disegno relativo alla matrice, tenendo conto che i tasti della riga 9 sono tipici dello SVI 728). Come si fa? Semplice: mettiamo caso che vogliamo rilevare il tasto shift, dalla matrice della tastiera vediamo che si trova sulla riga numero 6, e precisamente in posizione 0. Per rilevare lo stato del tasto potremo fare, quindi:

```
10 A=IMP (&HAA) AND & HF0
20 OUT &HAA,A OR 6
30 IF IMP (&HA9) AND 1 THEN PRINT «IL TASTO SHIFT NON È PREMUTO» ELSE PRINT «IL TASTO SHIFT È PREMUTO»
40 GOTO 10
```

Le linee 10 e 20 fanno sì che non vengano alterati i 4 bit più significativi del port C, mediante mascheratura. La linea 30 fa il test del bit 0 della riga della tastiera letta. Se il valore di questo bit è 1 allora il tasto non è premuto, se è a 0 allora è premuto.

Se provate a tenere premuto lo shift per un po' di tempo vedrete, però che a volte non viene riconosciuto. Perché? Bene, il difetto di questo metodo è proprio questo: a volte il tasto non viene riconosciuto perché passa del tempo tra l'attimo in cui il programma seleziona la riga da leggere ed il momento in cui la legge effettivamente. Se arriva nel frattempo un interrupt da parte del VDP (ogni cinquantesimo di secondo ne viene generato uno) allora la CPU esegue la routine di interrupt. Questa routine, tra l'altro, prevede la scansione della tastiera, e quindi va ad alterare il nostro puntatore alla riga della matrice. Insomma, se arriva un interrupt noi andiamo a leggere un tasto sbagliato. Per questo bisognerebbe disabilitare l'interrupt prima di accedere all'8255 e riabilitarlo dopo aver letto la tastiera. Difficile anche se non impossibile da fare in Basic: vedremo ora che è più facile aggirare l'ostacolo. Per farlo bisogna sapere questo: non tutto il male viene per nuocere, la routine di scansione della tastiera ci fa pure un buon servizio; infatti ci ricopia in RAM «l'immagine» dello stato della tastiera. A partire dall'indirizzo &HFBE5 e per l'estensione di 11 byte troveremo i byte corrispondenti alla lettura delle righe della tastiera, già bell'e pronti per essere «PEEKati» da Basic! A dire il vero la stessa cosa la troveremo anche a partire dall'indirizzo &HFBDA, la doppia tabella serve al soft di gestione della tastiera per confrontare lo stato di un tasto letto durante il ciclo di interrupt con lo stato letto il ciclo precedente. A noi comunque questo non interessa, visto che alla fine di ogni ciclo di interrupt la tabella «nuova» viene ricopiata nella «vecchia», e così appaiono identiche viste da una routine «esterna». Vediamo ora come leggere lo stato di un tasto senza problemi:

Tabella D

	X7	X6	X5	X4	X3	X2	X1	X0
Riga 9	4	3	2	1	0	opz.	opz.	opz.
Riga 10	punto	virgola	meno	9	8	7	6	5

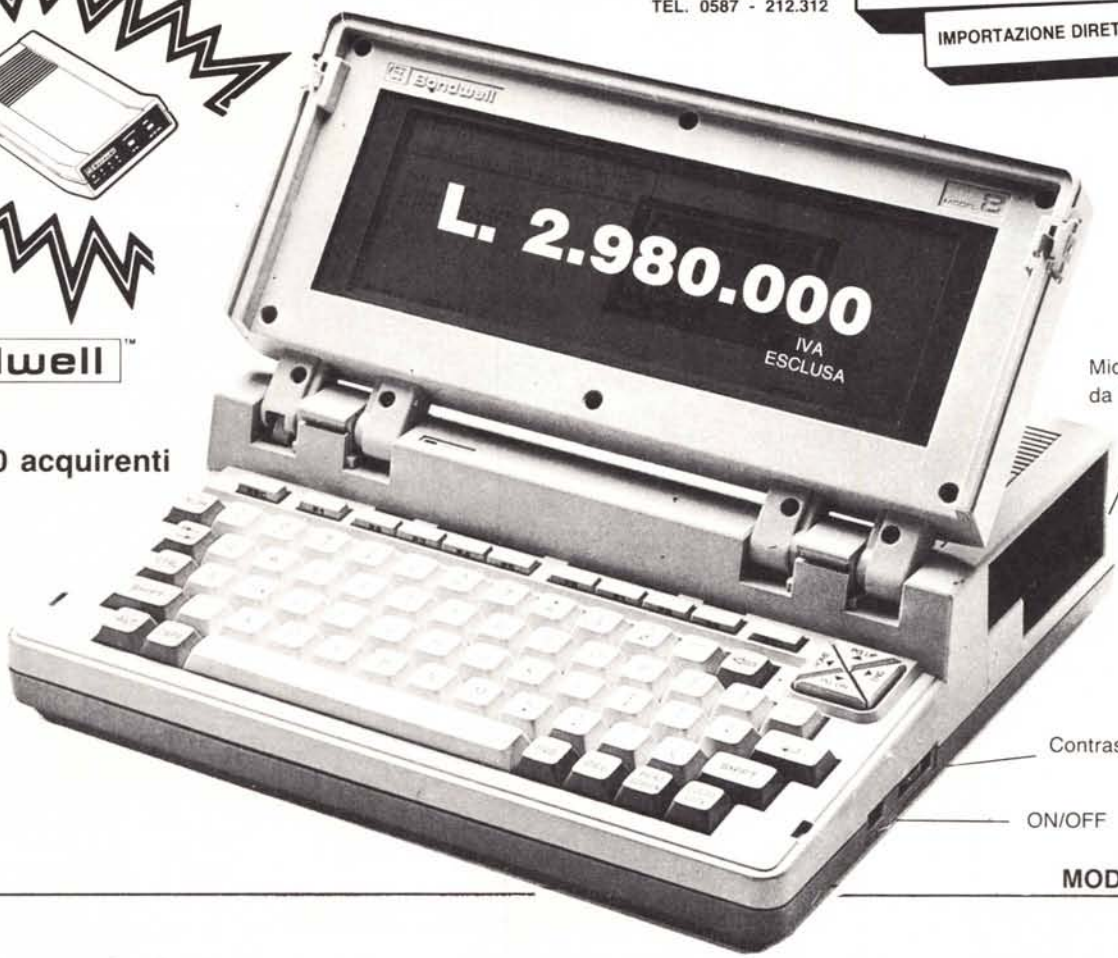
Bondwell™

DISK-DRIVE
STAMPANTI
COMPUTER
MONITOR

LA CASA DEL COMPUTER
CASELLA POSTALE 142
56025 PONTEDERA (PI)
VIA MISERICORDIA, 84
TEL. 0587 - 212.312

IMPORTAZIONE DIRETTA

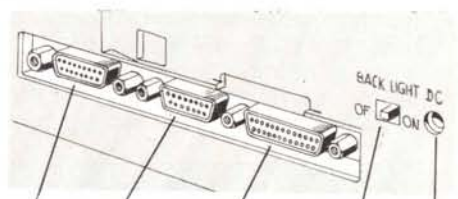
UN MODEM
Bondwell™
IN REGALO
ai primi 1.000 acquirenti

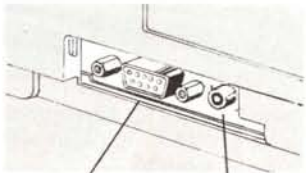


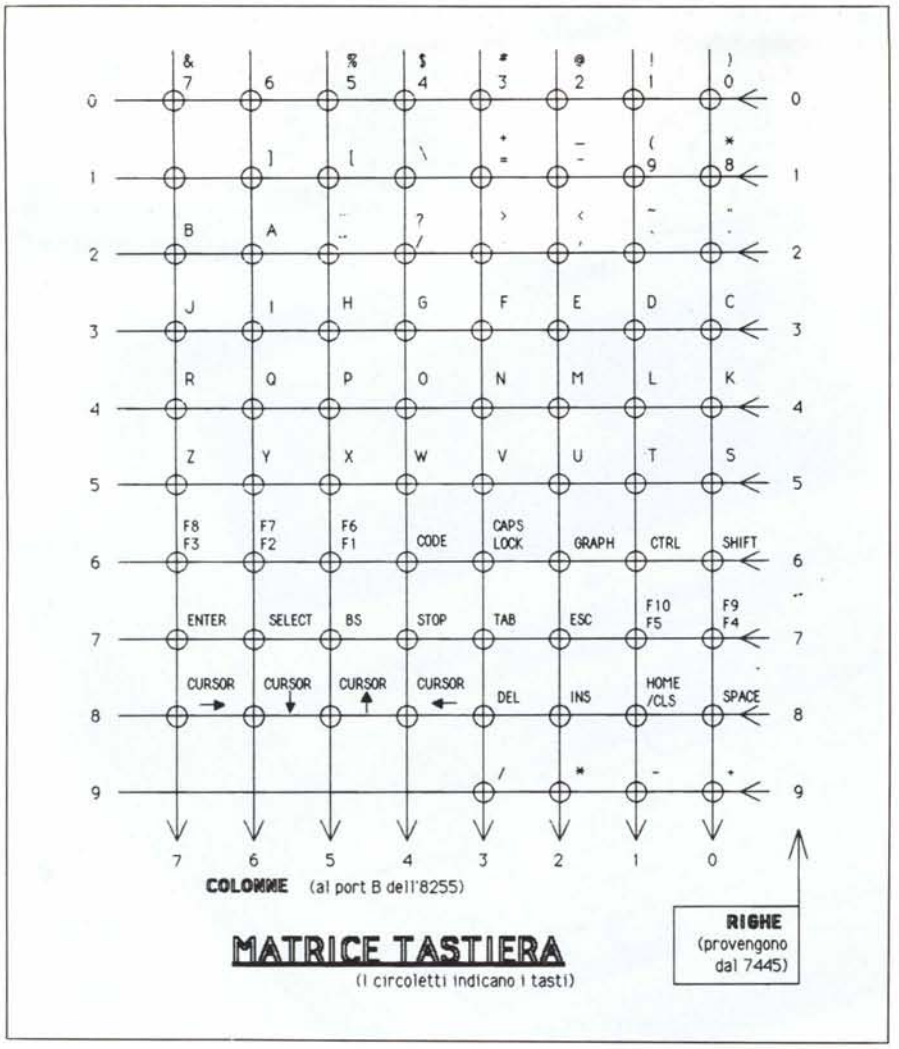
MODEL 8

IL PIÙ PICCOLO E POTENTE PC PORTABILE SI CHIAMA Bondwell™ 8

- Facilmente trasportabile
- Peso: Kg. 4,500
- Dimensioni: cm. 28 × 31 × 78
- IBM-PC compatibile (DOS 2.11 su licenza Microsoft)
- Dischetto con MS/DOS 2.11, GW Basic 2.0 e manuali inclusi
- Basso consumo ottenuto con l'impiego di componenti CMOS
- Microprocessore: 80C88, 4.77 MHz
- Memoria RAM: 512K
- Schermo a cristalli liquidi ad alto contrasto, illuminabile, e con risoluzione 640 × 200 (grafica), 80 × 25 (testo)
- Floppy disk interno da 3'' 1/2 doppia faccia/doppia densità da 720K formattati
- Orologio/Calendario mantenuto da batterie al nichel-cadmio ricaricabili
- Batterie ricaricabili 12V-3A
- Tastiera con 76 chiavi e basso profilo, compatibile con lo standard PC/XT, dotata di funzioni del PAD numerico, 10 tasti funzione ecc. ecc.
- Porta seriale standard R-232C
- Porta parallela per stampanti
- Porta per la connessione del 2° Drive (5'' 1/4 oppure 3'' 1/2)
- Uscite per video RGB/TTL e video-composito
- Led segnalatore intermittente di fine carica
- Alimentatore/Caricabatterie AC/DC
- Hard e Soft realizzati per ottenere il massimo della compatibilità IBM-PC. Possono essere eseguiti i più popolari pacchetti software come: Lotus 1-2-3, Symphony, D Base II e III, Wordstar, Flight Simulator, Frame work, Jem, Sidekick, PFS serie, ...








```
10 IF PEEK(&HFBE5+6) AND 1 THEN PRINT «IL TASTO SHIFT NON È PREMUTO» ELSE PRINT «IL TASTO SHIFT È PREMUTO»
20 GOTO 10
```

crediamo che sia abbastanza chiaro il metodo: si somma a &HFBE5 il numero della riga della tastiera che deve essere letta e si fa una PEEK. Poi si fa un AND per testare il bit voluto. Se il risultato è 0 allora il tasto è premuto, in caso contrario il contrario! Ricordiamo i valori da usare nell'operazione di AND per testare i vari bit:

BIT	VALORE PER L'AND
0	1
1	2
2	4
3	8
4	16
5	32
6	64
7	128

provate a scrivere una routine che riconosca la pressione di un altro tasto, con questo metodo.

Locazioni utili della RAM

Già che ci siamo nel discorso della tastiera vediamo alcune locazioni interessanti che hanno a che fare con essa, anche se non c'entrano direttamente con il PPI 8255.

Locazione &HFBB0: se contiene un valore diverso da 0 si ha un ritorno al Basic ogni volta che vengono pigiati assieme i tasti CONTROL, SHIFT, GRAPH e CODE. Funziona anche nei programmi in L/M, a patto che l'interrupt sia abilitato. Provate a fare POKE &HFBB0,1 e pigiate i suddetti tasti contemporaneamente per rendervi conto dell'effetto.

Locazione &HFBB1: se contiene un valore diverso da 0 vengono disabilitati STOP e CONTROL-STOP in un programma Basic: provate POKE &HFBB1,1 e poi tenterete (invano!) di fermare un programma (se però avete dato in precedenza la POKE &HFBB0,1 potrete uscirne pigiando i famosi 4 tasti).

Locazione &HFCAB: si tratta del flag associato al tasto CAPS LOCK: se contiene 0 vengono battute le minuscole, se contiene un numero diverso da 0 allora sono maiuscole. Pigiando il tasto CAPS LOCK il contenuto di questa locazione viene complementato (e quindi vale &HFF se prima valeva 0 e viceversa). Facendo però una POKE con un valore diverso allora potremo bloccare le maiuscole: infatti anche pigiando il CAPS LOCK non si otterrà mai 0. L'unica cosa è non far caso al led di CAPS LOCK, che si accende solo se questa locazione va a &HFF pigiando il CAPS LOCK, quindi mai dopo che avremo fatto POKE &HFCAB,1.

Il buffer di tastiera

L'MSX possiede un buffer di tastiera che può contenere al massimo 39 caratteri. È allocato a partire da &HFBF0 e si estende per 40 byte. Viene gestito da 2 puntatori. Per inserire una stringa nel buffer il sistema parte dall'indirizzo che sta nel puntatore contenuto in &HF3F8 e &HF3F9, arrivato alla fine del buffer continua dall'inizio dello stesso (il buffer è «circolare»). Per leggere i caratteri dal buffer il sistema li prende dall'indirizzo che si trova nel puntatore contenuto in &HF3FA e &HF3FB, incrementandolo (andando «a capo» alla fine del buffer) fino a raggiungere l'altro puntatore, il che significa buffer vuoto. Si può sfruttare questa caratteristica per forzare nel buffer una stringa da Basic. Per cancellare i buffer si può usare (anche da Basic, con una USR) la routine del Bios di indirizzo &H0156.

Ecco un esempio di introduzione di una stringa nel buffer:

```
10' CARICATORE PER PROGRAMMA BASIC
20'
30 A$ = «cload» + CHR$ (13) + «run» + CHR$(13)
40 FOR I=1 TO LEN(A$):POKE I-1+&HFBF0,ASC(MID$(A$,I,1)):NEXT
50 POKE &HF3FA,&HF0:POKE &HF3FB,&HFB ' posiziona puntatore al primo carattere del buffer
60 A=&HFBF0+LEN(A$)
70 POKE &HF3F8,A AND 255:POKE &HF3F9,(65536!+A)/256 ' posiziona il puntatore all'ultimo carattere della stringa presente nel buffer.
```

Questo programma può venire salvato in modo ASCII e mandato in esecuzione con RUN«CAS:» lui provvederà a caricare un programma Basic con CLOAD ed a mandarlo in esecuzione con RUN.

Per chi volesse saperne di più sull'8255 consigliamo uno dei seguenti manuali:

SAB 8080 Microcomputer User's Manual - Siemens
NEC CATALOG - NEC Electronics (Europe) GmbH (il nostro è dell'82)
Microsystem Components Handbook volumi I e II - Intel (1986)

HERCULES e COLOR GRAPHIC

FINALMENTE D'ACCORDO

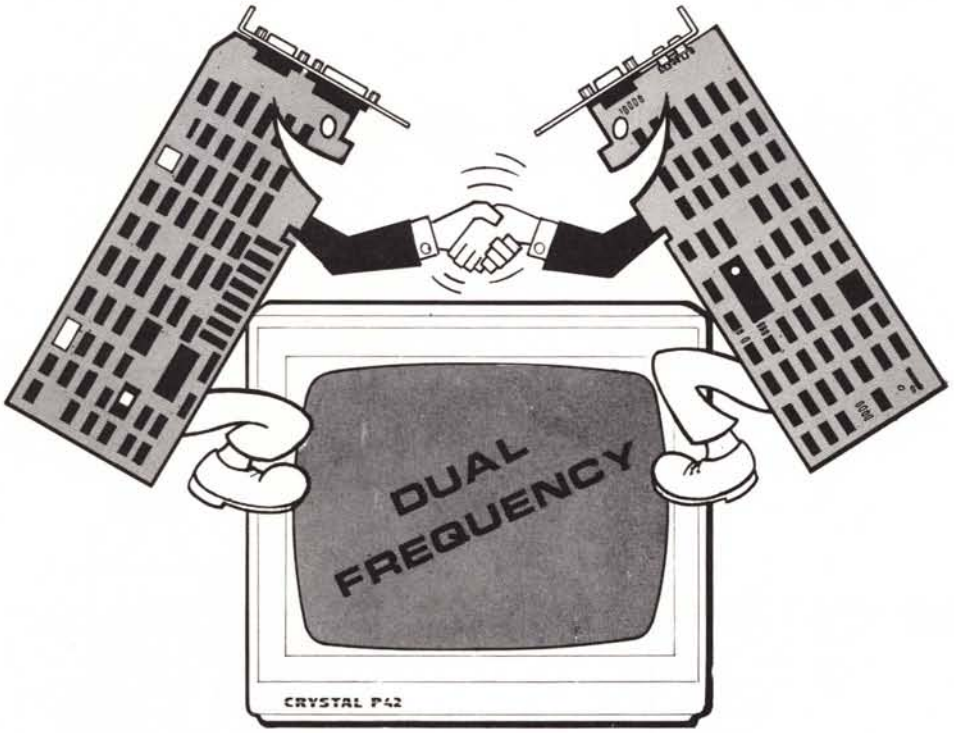
DISK-DRIVE
STAMPANTI
COMPUTER
MONITOR

LA CASA DEL COMPUTER

IMPORTAZIONE DIRETTA

DOPPIO INGRESSO
TTL + COMPOSITO

CRYSTAL P42



DISPONIBILE ANCHE NELLA VERSIONE TTL

BIANCO
CRYSTAL PWD

VERDE
CRYSTAL P39

AMBRA
CRYSTAL PLA

SWITCH PER SELEZIONE DELLA FREQUENZA ORIZZONTALE

MONITOR PER E.G.A. TVM MD7

- SETTAGGIO AUTOMATICO DELLA FREQUENZA ORIZZONTALE (da 18,5 a 21.85 MHz)
- POSSIBILITÀ DI SELEZIONE DEI COLORI VERDE ED ARANCIO CON SWITCH SUL FRONTALE
- VENTILATORE INTERNO E DEGAUSS AUTOMATICO

LA CASA DEL COMPUTER - VIA DELLA MISERICORDIA, 84 - 56025 PONTEDERA (PI) - Tel. 0587 - 212.312
(NUOVA SEDE) - VIA T. ROMAGNOLA, 63 - 56012 FORNACETTE (PI) - Tel. 0587 - 422.022

RICHIEDETECI IL CATALOGO - SCONTI AI SIG.RI RIVENDITORI