Programmare in C su Amiga (42)

di Dario de Judicibus

In questa puntata parleremo delle due tabelle principali contenute in una mappa di tastiera, e dei blocchi extra utilizzati per definire le stringhe di emissione ed i tasti morti. Nella Scheda Tecnica incominceremo a vedere le prime funzioni della ***gadtools.library*** *ed i tag utilizzati da queste funzioni per specificare i molti parametri necessari alla definizione dei controlli tramite questa libreria*

Introduzione

Nella 41ª puntata abbiamo visto sei delle otto tabelle che definiscono una mappa di tastiera, e cioè le sei tabelle minori, che servono a definire il formato utilizzato per le sequenze di emissione e le caratteristiche intrinseche del tasto (emissione continua e blocco del maiuscolo).

In questa puntata vedremo le due tabelle maggiori, quelle cioè che definiscono la sequenza di caratteri effettivamente emessa quando uno o più tasti vengono premuti dall'utente: la **km_LoKeyMap**, relativa alla sezione bassa della tastiera, e la **km_HiKeyMap**, relativa alla sezione alta della tastiera.

La prima è relativa ai tasti con codice di scansione che va da **0x100** a **0x3F**, la seconda a quelli con codice che va da **0x40** a **0x67**.

Le due tabelle hanno la stessa struttura, per cui non è necessario descriverle entrambe. Ogni tabella, detta anche mappa di emissione, è formata da due parti: una fissa, contenente quattro byte per ogni codice di scansione, ed una a lunghezza variabile, formata da un certo numero di blocchi extra che servono a definire i tasti di tipo **KCF_DEAD** e **KCF_STRING**.

Nella scorsa puntata abbiamo visto il formato base utilizzato nella parte fissa della mappa di emissione, che abbiamo chiamato *modello a quattro*. Ricordo brevemente che in questo formato ogni byte corrisponde ad un singolo carattere che viene emesso quando il tasto viene premuto da solo, od in combinazione con uno o più modificatori. Quali, dipende dai qualificatori nella corrispondente tabella dei formati. Facciamo un esempio. Prendiamo il tasto in alto a sinistra del tastierino numerico dell'A2000. Il codice di scansione è **0x54**. Stiamo quindi parlando delle tabelle relative alla sezione alta della tastiera. Supponiamo ora che il ventisettesimo byte nella tabella dei formati contenga il valore **0x05**. Questo vuol dire che i modificatori interessati sono *shift* e *control*, dato che **KCF_SHIFT** vale **0x01** mentre **KCF_CONTROL** vale **0x04**. Andiamo ora a vedere la mappa alta di emissione, e precisamente i byte che vanno dal 105-esimo al 108-esimo inclusi. Supponiamo di trovare la sequenza **0x3C7B5B28**. Si tratta di una sequenza a caso, non riferita ad alcuna mappa di mia conoscenza, ma comunque verosimile. Essa corrisponde ai quattro caratteri **«<»**, **«{»**, **«[»** e **«(»**, nell'ordine. Le combinazioni possibili e le corrispondenti sequenze di emissione sono allora:

Combinazione	Carattere emesso
tasto da solo	(parentesi tonda aperta
tasto + shift	[parentesi quadra aperta
tasto + control	{ parentesi graffa aperta
tasto + shift + control	< segno di minore

Vediamo ora in dettaglio la struttura completa delle tappe di emissione.

Le mappe di emissione

Una mappa di emissione è formata da una tabella a dimensione fissa, ed una serie di blocchi a dimensione variabile.

La tabella fissa contiene quattro byte per ogni codice di scansione, che chiameremo *blocco base*. In realtà, essa contiene tanti blocchi base quanti indicatori di formato ci sono nelle tabelle dei formati, e cioè 64 per la sezione bassa, e 38 per quella alta. Quest'ultima può anche essere formata da 64 byte come la precedente se si vuole, ma in questo caso tutti i blocchi in eccesso vanno riempiti da **«0»**, dato che in realtà sarebbero relativi a codici di scansione esistenti, anche se ignorati nel rimappaggio, come quelli per i bottoni del mouse, o per i tasti *Amiga*.

Un blocco base ha solo due formati. Se l'indicatore di formato corrispondente non contiene **KCF_STRING** oppure **KCF_DEAD**, allora va interpretato secondo il modello a quattro, di cui abbiamo già parlato. Altrimenti, esso rappresenta il *puntatore* ad un blocco extra, dato che in tal caso sono necessari più

di quattro byte per definire le sequenze di emissione.

Ci sono due tipi di blocchi extra: uno per le stringhe di emissione, ed uno per i tasti morti, di cui abbiamo parlato nella scorsa puntata. Analizziamone la struttura in dettaglio.

Le stringhe di emissione

Il blocco per le stringhe di emissione è formato da due parti: l'area per i descrittori di stringa, e quella per le stringhe vere e proprie.

A seconda del numero di modificatori che possono essere usati con il tasto, infatti, un blocco per le stringhe di emissione può contenere da una (nessun modificatore permesso) fino ad otto stringhe (tutti e tre i modificatori possono essere usati). Per ogni stringa c'è un descrittore di stringa. Questi è formato da due byte, uno che fornisce la lunghezza della stringa, e l'altro che contiene la distanza fra il primo byte dell'area dei descrittori ed il primo byte della stringa associata a quel descrittore.

I più bravi di voi nel fare i calcoli si saranno resi conto a questo punto di un paio di cose.

Innanzi tutto che una stringa non può essere più lunga di 255 byte, dato che per la lunghezza è utilizzato un solo byte. Inoltre, dato che abbiamo bisogno di sapere la lunghezza della stringa, è evidente che le stringhe in questione non sono del tipo che termina con un **NULL** [*null-terminated strings*]. Questo perché il carattere esadecimale **0x00** può far parte di una stringa di emissione, e quindi non può ovviamente essere utilizzato per determinare la fine della stringa.

La seconda cosa è che anche la distanza tra una stringa ed il primo byte dell'area dei descrittori non può essere più lunga di 255 byte. Questo limita molto più seriamente la lunghezza delle stringhe, specialmente se ne abbiamo tante. Ad esempio, nel caso di otto stringe solo l'ultima può avere al massimo 255 byte. Le altre sette devono condividersi i 254 byte compresi fra il primo byte dell'area dei descrittori ed il primo byte dell'ultima stringa, *meno* per giunta i sedici byte degli otto descrittori. In tutto 238 byte al massimo, il che fa una media di esattamente 34 byte per stringa.

Il motivo principale per cui si è imposta questa limitazione è che comunque il *buffer* di ingresso della *console* permette solo 32 byte per tasto. Per cui i conti tornano.

Vediamo un esempio di blocco per le stringhe di emissione. Consideriamo il tasto funzione *F1* (codice di scansione **0x50**). Nelle mappe di tastiera dell'Amiga i tasti funzione hanno solo due sequenze di emissione: una se il tasto viene premuto da solo, e l'altra se viene premuto contemporaneamente anche il tasto per il maiuscolo (*shift*). L'identificatore del formato relativo ad *F1* è di solito, infatti, **0x41**, cioè **KCF_STRING + KCF_SHIFT**. Questo non vuol dire naturalmente che la pressione contemporanea di altri modificatori viene ignorata dal sistema. Essa è comunque memorizzata, tant'è che un programma può sempre ottenere questa informazione tramite la **console.device.** Il punto è che la mappa standard utilizzata, ad esempio dal CLI, emetterà solo a fronte delle due combinazioni suddette, e non altre.

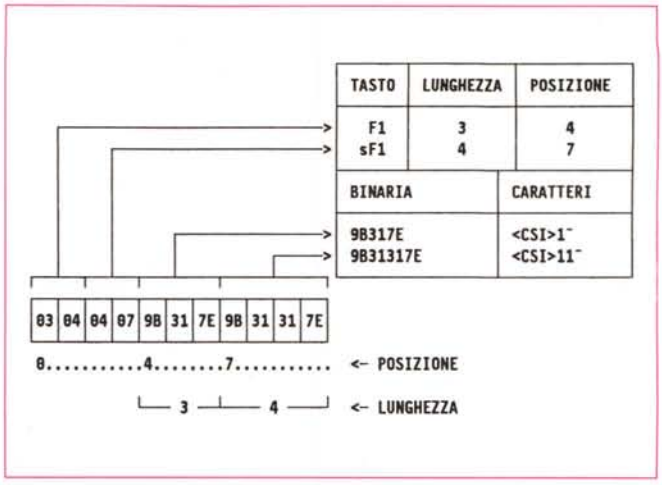


Figura 1
Analisi del blocco extra per F1.

Il blocco extra associato ad *F1* è generalmente il seguente:

..... 03040407 9B317E9B 31317E.....

I primi quattro byte sono i due descrittori di stringa: il primo per la stringa emessa a fronte della pressione del tasto da solo, il secondo per quella emessa quando anche lo *shift* è premuto in contemporanea.

Il primo descrittore dice che la stringa di emissione per *F1* è lunga tre byte, ed è posizionata quattro byte a partire dall'inizio del blocco extra. Si tratta quindi di **9B317E**, che corrisponde ai caratteri **<CSI>1~**.

Il secondo descrittore, invece, dice che la stringa di emissione per *shift+F1* è lunga quattro byte, ed è posizionata sette byte a partire dall'inizio del blocco extra. Si tratta quindi di **9B31317E**, che corrisponde ai caratteri **<CSI>11~**.

L'analisi del bocco è riportata in figura 1.

I tasti morti

Nella scorsa puntata abbiamo introdotto il concetto di *tasto morto*. Ricordo che si tratta di tasti che in particolari situazioni possono modificare la sequenza di emissione di altri tasti, o la cui sequenza di emissione può essere modificata da altri tasti. Esistono quindi due tipi di tasti morti: quelli *muti* [*dead key*], e quelli *parlanti* [*deadable key*]. Un tasto muto non è capace di emettere una propria sequenza di emissione, ma può modificare quella di un tasto morto parlante, se questi è premuto immediatamente dopo quello muto. Un tasto muto ed uno parlante non vanno cioè premuti contemporaneamente, come accadeva per i modificatori ed i tasti base, ma in sequenza. Questo perché in realtà non è detto che un singolo tasto sia sempre muto, ma può darsi che ad essere *muta* sia solo la combinazione di un tasto normale con un certo qualificatore.

Facciamo un paio di esempi.

Il tasto con l'accento grave **"`"** nella tastiera tedesca, è un tasto muto. Se lo si preme non succede niente. Se tuttavia subito dopo si preme ad esempio il tasto contrassegnato dalla **"A"**, invece della vocale normale viene emessa quella con l'accento grave, cioè una **"à"**. Lo stesso effetto si potrebbe ottenere su una tastiera americana usando come tasto muto la combinazione **Alt+G**. Il tasto contrassegnato dalla **"G"** quindi, per le tastiere USA, è un tasto morto muto, anche se nella maggior parte dei casi si comporta come tutti gli altri tasti alfabetici. Viceversa, il tasto **"A"** è, nei casi suddetti, un tasto morto parlante, capace cioè di emettere direttamente una sequenza di caratteri, che tuttavia può essere modificata da un tasto morto muto.

Ovviamente non tutti i tasti possono essere modificati da un tasto muto. Infatti, se nell'esempio precedente della tastiera americana si fosse premuta la **"T"** dopo **Alt-G**, avremmo avuto come risultato semplicemente la **"t"**.

I tasti morti, siano essi muti o parlanti, sono identificati nella tabella dei formati dal qualificatore **KCF_DEAD**, eventualmente insieme ad uno o più dei

La scheda tecnica: Inside 2.0

La **gadtools.library** è una nuova libreria formata da 18 funzioni, il cui scopo è quello di rendere più semplice la gestione dei controlli e dei menu sotto Intuition.

In pratica si tratta di una nuova tecnica di programmazione che, pur non togliendo al programmatore la possibilità di lavorare in modo tradizionale, cerca di fornire un meccanismo più potente per definire e manovrare interfacce grafiche.

Queste funzioni, tra l'altro, sfruttano appieno la nuova tecnica di programmazione a *tag* tipica della nuova versione del sistema operativo dell'Amiga, della quale parleremo in una delle prossime puntate.

In questa puntata vedremo le schede tecniche di due delle prime tre funzioni (quelle di *creazione*, due delle quali hanno una doppia versione: a parametri fissi ed a lista variabile).

CreateContext

Alloca un'area dati per memorizzare una serie di informazioni a carattere contestuale, relative alla finestra associata alla lista di puntatori fornita in ingresso.

prototipo

```
struct Gadget *CreateContext  // Puntatore all'area dati allocata
(
  struct Gadget **glistpointer  // Indirizzo di un puntatore ad un
                                // controllo
) ;

// CreateContext() ritorna NULL se fallisce
```

In realtà questa funzione crea un controllo invisibile e non selezionabile, contenente un'area di servizio sufficientemente grande da contenere i dati che *GadTools* deve memorizzare. Questa funzione va chiamata prima di qualunque altra funzione utilizzata per creare altri controlli.

CreateContext() richiede in ingresso l'indirizzo ad un puntatore ad un controllo, preventivamente impostato a **NULL**. Se l'allocazione ha avuto successo, e quindi il valore ritornato dalla funzione non è nullo, il puntatore ad un controllo di cui si era passato in ingresso l'indirizzo, può essere utilizzato con le funzioni tipo **AddGlist()**, **RefreshGlist()** e via dicendo, od assegnato al campo **FirstGadget** della struttura **NewWindow** utilizzata nella creazione di una finestra.

Un esempio è riportato in figura 3.

```
/*
**  Falso controllo utilizzato per la creazione dell'area dati per GadTools
*/
 struct Gadget *context ;

/*
**  Puntatore ad una lista di controlli associati alla finestra da creare
*/
 struct Gadget *gadglist = NULL ;

/*
**  Allocazione dell'area contestuale della finestra
*/
 context = CreateContext(&gadglist) ;

/*
** ------------------------------------------------------------------ **
** Qui vanno le chiamate alle altre funzioni di creazione dei controlli... **
** ------------------------------------------------------------------ **
*/

/*
**  OK. Creiamo la nuova finestra.
*/
 if (gad)
 {
   nw.FirstGadget = gadglist ; // Associamo alla finestra la lista dei controlli
   if (w = OpenWindow(&nw))    // Creiamo una nuova finestra.
   {
     DoWhatYouWant() ;  // OK. Fai quello che vuoi. Quando hai finito...
     CloseWindow(w) ;   // ...chiudi la finestra.
   }
 }

 FreeGadgets(gadglist) ;  // Deallochiamo la lista dei controlli
```

Figura 3 - Esempio di utilizzo della CreateContext().

CreateGadgetA

Alloca ed inizializza un nuovo controllo.

prototipo

```
struct Gadget *CreateGadgetA  // ratteri che stanno nei limiti posti
(
  ULONG             kind     , // Tipo di controllo da creare
  struct Gadget    *previous , // Controllo precedente a quello nuovo
  struct NewGadget *newgad   , // Caratteristiche del controllo
  struct TagItem   *taglist    // Lista di "tag"
) ;
```

Il nuovo controllo è agganciato di seguito a quello il cui puntatore è fornito in ingresso alla funzione nel parametro **previous**. Questo permette di costruire una lista di controlli.

Importante: se ai campi **ng_VisualInfo** ed **ng_TextAttr** non sono assegnati dei puntatori validi rispettivamente ad una struttura **VisualInfo** e **TextAttr**, questa funzione fallisce.

La lista dei *tag* complementa la struttura **NewGadget** nella definizione delle caratteristiche del controllo. Quest'ultima va pensata un po' come la **NewWindow** per la creazione di nuove finestre. Si tratta cioè di una struttura di appoggio che contiene le informazioni necessarie a definire il controllo.

Qui di seguito le schede relative ai vari *tag* validi per ogni tipo di controllo, ed il loro significato.

Controllo: ..._KIND	Tutti i tipi di controllo
Tag / Tipo	**Descrizione**
GT_Underscore char	Simbolo utilizzato nel testo del controllo per indicare il carattere di attivazione.

Controllo: BUTTON_KIND	Pulsante a pressione
Tag / Tipo	**Descrizione**
GA_Disabled BOOL	TRUE disabilita il controllo, altrimenti FALSE (default).

Controllo: CHECKBOX_KIND	Casella di spunta
Tag / Tipo	**Descrizione**
GA_Disabled BOOL	TRUE disabilita il controllo, altrimenti FALSE (default).
GTCB_Checked BOOL	Stato iniziale della casella di spunta. Il default è FALSE.

Controllo: CYCLE_KIND	Selezione ciclica
Tag / Tipo	Descrizione
GA_Disabled BOOL	TRUE disabilita il controllo, altrimenti FALSE (default)
GTCY_Active UWORD	Numero d'ordine (a partire da zero) della scelta iniziale (default = 0).
GTCY_Labels STRPTR *	Puntatore ad un vettore di stringhe corrispondenti alle possibile scelte. L'ultima è nulla.

Controllo: INTEGER_KIND	Campo intero
Tag / Tipo	Descrizione
GA_Disabled BOOL	TRUE disabilita il controllo, altrimenti FALSE (default).
GA_TabCycle BOOL	TRUE attiva il salto al campo intero successivo (TAB) ed a quello precedente (s-TAB) (default).
GTIN_MaxChars UWORD	Numero massimo di cifre che possono essere immesse nel campo (default = 10).
GTIN_Number LONG	Valore iniziale del campo (default = 0).
STRINGA_ExitHelp BOOL	TRUE fa sì che premendo HELP si esca dal campo e si riceva GADGETUP con codice 0x5F.

Controllo: LISTVIEW_KIND	Elenco scorrevole
Tag / Tipo	Descrizione
GTLV_Labels struct List *	Lista di voci. Il testo da visualizzare nell' elenco è nei campi "ln_Name".
GTLV_ReadOnly BOOL	Se TRUE, l'elenco può solo essere letto.
GTLV_ScrollWidth UWORD	Larghezza della barra di scorrimento (> 0). Il default è 16.
GTLV_Selected UWORD	Numero d'ordine della voce selezionata, o 0 se nessuna voce è selezionata (default = 0).
GTLV_ShowSelected struct Gadget *	Puntatore ad un campo stringa editabile in cui compaia la voce selezionata. Se NULL, la voce è posizionata sotto l'elenco stesso.
GTLV_Top UWORD	Prima voce visibile nell'elenco (default = 0).
LAYOUTA_Spacing	Spaziatura extra tra le linee dell'elenco che contengono le voci (default = 0).

Controllo: MX_KIND	Gruppo di bottoni mutualmente esclusivi
Tag / Tipo	Descrizione
GTCY_Active UWORD	Numero d'ordine (a partire da zero) del bottone inizialmente selezionato (default = 0).
GTMX_Labels STRPTR *	Puntatore ad un vettore di stringhe corrispondenti ai testi dei bottoni. L'ultima è nulla.
GTMX_Spacing UWORD	Aggiunta all'altezza del font, dà la distanza tra un bottone ed un'altro (default = 1).
LAYOUTA_Spacing	SOLO PER COMPATIBILITA' - Usa GTMX_Spacing Spaziatura extra tra i bottoni del gruppo. Il default è (AltezzaFont - 8).

Controllo: NUMBER_KIND	Campo numerico (sola lettura)
Tag / Tipo	Descrizione
GTNM_Border BOOL	Se TRUE, viene disegnato un bordo rettangolare intorno al campo.
GTNM_Number LONG	Intero (long) con segno da visualizzare in sola lettura (default = 0).

Controllo: PALETTE_KIND	Tavolozza dei colori
Tag / Tipo	Descrizione
GA_Disabled BOOL	TRUE disabilita il controllo, altrimenti FALSE (default).
GTPA_Color UBYTE	Colore inizialmente selezionato (default = 1).
GTPA_ColorOffset UBYTE	Primo colore da usare nella tavolozza (default = 0).
GTPA_Depth UWORD	Numero di piani di bit della tavolozza (default = 1, cioè due colori).
GTPA_IndicatorHeight UWORD	Altezza dell'indicatore del colore, posto al di sopra della tavolozza.
GTPA_IndicatorWidth UWORD	Larghezza dell'indicatore del colore, posto a sinistra della tavolozza.

Controllo: SCROLLER_KIND	Barra di scorrimento
Tag / Tipo	Descrizione
GA_Disabled BOOL	TRUE disabilita il controllo, altrimenti FALSE (default).
GA_Immediate BOOL	TRUE permette la ricezione di tutti gli eventi IDCMP_GADGETDOWN emessi dalla barra. Il default è FALSE.
GA_RelVerify BOOL	TRUE permette la ricezione di tutti gli eventi IDCMP_GADGETUP emessi dalla barra. Il default è FALSE.
GTSC_Arrows UWORD	Dimensione longitudinale dei pulsanti per lo scorrimento (frecce).
GTSC_Top WORD	Posizione del primo elemento visibile nella lista gestita dalla barra (default = 0).
GTSC_Total WORD	Numero totale di elementi della lista gestita dalla barra (default = 0).
GTSC_Visible WORD	Numero totale di elementi visibili della lista gestita dalla barra (default = 2).
PGA_Freedom	Orientamento della barra: LORIENT_VERT (verticale) o LORIENT_HORIZ (orizzontale, default).

Controllo: SLIDER_KIND	Indicatore di livello
Tag / Tipo	Descrizione
GA_Disabled BOOL	TRUE disabilita il controllo, altrimenti FALSE (default).
GA_Immediate BOOL	TRUE permette la ricezione di tutti gli eventi IDCMP_GADGETDOWN emessi dalla barra. Il default è FALSE.
GA_RelVerify BOOL	TRUE permette la ricezione di tutti gli eventi IDCMP_GADGETUP emessi dalla barra. Il default è FALSE.
GTSL_DispFunc LONG (*function)(struct Gadget *, WORD)	Funzione che calcola il livello da visualizzare. Non esiste alcuna funzione di default. I parametri in ingresso sono il puntatore al controllo, il livello (WORD). In uscita il livello (LONG).
GTSL_Level WORD	Livello attuale in cui si trova il cursore dell'indicatore (default = 0).
GTSL_LevelFormat STRPTR	Definizione del formato del livello secondo lo stile C. Usare sempre "l" -- Vedi RawDoFmt().
GTSL_LevelPlace STRPTR	Posizione del livello rispetto al cursore: PLACETEXT_LEFT (default), PLACETEXT_RIGHT, PLACETEXT_ABOVE, e PLACETEXT_BELOW.
GTSL_Max WORD	Livello massimo a cui può arrivare il cursore dell'indicatore (default = 15).
GTSL_MaxLevelLen UWORD	Massima lunghezza in caratteri dell'etichetta posta a fianco del cursore dell'indicatore.
GTSL_Min WORD	Livello minimo a cui può arrivare il cursore dell'indicatore (default = 0).
PGA_Freedom	Orientamento dell'indicatore: LORIENT_VERT (verticale) o LORIENT_HORIZ (orizzontale, default).

Controllo: STRING_KIND	Campo caratteri
Tag / Tipo	Descrizione
GA_Disabled BOOL	TRUE disabilita il controllo, altrimenti FALSE (default).
GA_TabCycle BOOL	TRUE attiva il salto al campo stringa successivo (TAB) ed a quello precedente (s-TAB) (default).
GTST_MaxChars UWORD	Numero massimo di caratteri che possono essere immesse nel campo.
GTST_String STRPTR	Valore iniziale del campo, o NULL (default) se il campo deve essere inizialmente vuoto.
STRINGA_ExitHelp BOOL	TRUE fa sì che premendo HELP si esca dal campo e si riceva GADGETUP con codice 0x5F.

Controllo: TEXT_KIND	Testo statico (sola lettura)
Tag / Tipo	Descrizione
GTTX_Border BOOL	Se TRUE, viene disegnato un bordo rettangolare intorno al campo.
GTTX_CopyText BOOL	Se TRUE il testo viene clonato, e non solo referenziato. Vale solo per il testo iniziale, e non va utilizzato se GTTX_Text è NULL.
GTTX_Text BOOL	Puntatore ad una stringa di caratteri da visualizzare in sola lettura, o NULL (default).

Nell'ultima versione della 2.0 tutti i campi stringa e numerici hanno il salto automatico al campo successivo attivo **(GFLG_TABCYCLE)**. Il programma riceve l'evento **GADGETUP** con il codice **0x09** se l'utente preme **TAB** o **shift-TAB**.

CreateGadget

Alloca ed inizializza un nuovo controllo. Versione a parametri variabili della **CreateGadgetA()**.

prototipo

```
struct Gadget *CreateGadget  // ratteri che stanno nei limiti posti
(
  ULONG             kind     , // Tipo di controllo da creare
  struct Gadget    *previous , // Controllo precedente a quello nuovo
  struct NewGadget *newgad   , // Caratteristiche del controllo
  (tagtype)         firsttag , // Primo tag
  ...
) ;
```

Si tratta praticamente della **CreateGadgetA()**, solo che invece di richiedere in ingresso come quarto parametro il puntatore ad una lista di *tag*, permette di introdurre direttamente i vari tag ed i valori loro eventualmente associati, secondo la tipica sintassi a parametri variabili (quella della **printf()**, tanto per intenderci).

qualificatori già visti in precedenza, escluso **KCF_STRING**. D'altra parte la combinazione **KCF_DEAD** + **KCF_STRING** avrebbe poco senso.

Anche in questo caso, come in quello delle stringhe di caratteri, il blocco base relativo al tasto morto contiene un puntatore da quattro byte ad un blocco extra. Ed anche in questo caso, la prima parte del blocco extra è formata da un blocco di descrittori. La lunghezza e le caratteristiche di questo blocco dipende dal tipo di tasto morto (muto o parlante), e dal numero (e tipo) di qualificatori permessi con il tasto morto.

Consideriamo prima quest'ultimi. A seconda di quanti e quali qualificatori possono essere usati con un tasto morto, la struttura del blocco dei descrittori cambia. Il numero dei qualificatori determina il numero di descrittori nel blocco, come succedeva anche per le stringe. Quindi, un descrittore se non ci sono modificatori, due se ce ne è uno, quattro se ce ne sono due, otto se tutti e tre i modificatori possono essere utilizzati. A seconda poi di quali modificatori sono permessi, abbiamo una ben definita associazione tra i singoli descrittori e le possibili combinazioni. Questa associazione è riportata nello schema in figura 2.

Analizziamo adesso i singoli descrittori. Anche in questo caso, sono formati da due byte, solo che l'interpretazione del descrittore non è unica, come suc-

QUALIFICATORI			DESCRITTORI							
S	A	C	1	2	3	4	5	6	7	8
			k	-	-	-	-	-	-	-
S			k	S	-	-	-	-	-	-
	A		k	A	-	-	-	-	-	-
		C	k	C	-	-	-	-	-	-
S	A		k	S	A	S A	-	-	-	-
	A	C	k	A	C	A C	-	-	-	-
S		C	k	S	C	S C	-	-	-	-
S	A	C	k	S	A	S A	C	S C	A C	S A C

Le colonne di sinistra indicano i qualificatori associati al tasto nella tabella dei formati (oltre a KCF_DEAD, ovviamente).

Quelle di destra indicano a quale combinazione di tasti ciascun descrittore è associato ("-" indica che non c'è un descrittore associato).

Esempio - Alla riga identificata a sinistra da S+A, la tabella di destra indica che ci sono quattro descrittori: il primo si riferisce al tasto premuto da solo, il secondo al tasto premuto insieme a SHIFT, il terzo al tasto ALT, e l'ultimo alla pressione contemporanea del tasto e di entrambi i modificatori.

Figura 2 - Ordine dei descrittori in base ai qualificatori.

cedeva con i descrittori di stringa (lunghezza, posizione), ma dipende dal valore assunto dal primo byte. Le cose quindi sono un pochino più complicate che nel caso precedente.

Il primo byte del descrittore di un tasto morto può assumere tre valori: **0**, **DPF_DEAD** e **DPF_MOD.**

Se il primo byte è nullo, allora il secondo byte rappresenta il carattere che deve essere emesso. Non sono necessari quindi ulteriori blocchi. Ad esempio, la sequenza **0051** porta l'emissione del carattere **"Q"**, il cui valore esadecimale è **0x51**, appunto.

Se il primo byte è **DPF_DEAD**, allora ci troviamo in presenza di un tasto morto muto, ed il secondo byte serve appunto a definire come viene modificata la sequenza di un eventuale tasto morto parlante se premuto di seguito. Vedremo subito come.

Se il primo byte è **DPF_MOD**, allora ci troviamo in presenza di un tasto morto parlante. In questo caso, il secondo byte contiene la distanza fra il primo byte del descrittore del tasto, ed un blocco speciale, che contiene una serie di sequenze di emissione, ognuna da *un solo* byte.

Il secondo byte di un tasto morto muto serve allora come indice per selezionare dal blocco speciale del tasto morto parlante quale sequenza va emessa. La sequenza selezionata è quella alla posizione **indice** + **1**, dato che la prima sequenza è utilizzata nel caso che il tasto morto parlante non sia stato preceduto da un tasto morto muto.

In pratica, riassumendo...

1. **KCF_DEAD** nella tabella dei formati indica che si tratta di un tasto morto, e che quindi ci dobbiamo aspettare nel blocco base il puntatore ad un blocco extra, di cui la prima sezione è formata da uno o più descrittori da due byte

2. Gli altri qualificatori nella stessa tabella determinano quanti descrittori ci sono in testa al blocco base, ed a quali combinazioni di tasti essi sono associati.

3. Il primo byte di ogni descrittore indica se la combinazione associata si comporta come un tasto normale, come un tasto morto muto, o come un tasto morto parlante.

4. Nel primo caso il secondo byte corrisponde al carattere da emettere.

5. Nel secondo caso esso indica in quale posizione nella tabella delle sequenze di emissione dei tasti morti parlanti si trova il carattere da emettere.

6. Nel terzo caso esso indica quanto dista dal primo byte del blocco dei descrittori la tabella delle sequenze di emissione.

7. La tabella delle sequenze di emissione di un tasto morto parlante è formata da un certo numero di byte. Ogni byte rappresenta una sequenza di emissione. Il primo rappresenta il carattere emesso quando il tasto non è stato preceduto da un tasto morto parlante.

Anche in questo caso possiamo fare una serie di considerazioni.

Innanzi tutto è chiaro che una tabella di emissione di un tasto morto parlante conterrà tanti byte quanto il valore più elevato che nella stessa mappa viene ad assumere il secondo byte dei vari tasti morti muti, più uno. Questo perché altrimenti si rischia di prendere un byte fuori tabella come sequenza da utilizzare. Il *più uno* è dovuto al carattere che rappresenta la sequenza non modificata.

In secondo luogo, un tasto morto può contenere allo stesso tempo combinazioni normali, mute e parlanti. Ad esempio, si potrebbe definire un certo tasto in modo che premendo il tasto da solo venga emesso il segno di minore (**"<"**), ma premendolo prima insieme ad **ALT**, e poi da solo, vengano emesse le virgolette aperte (**"«"**). In questo caso il tasto da solo è un tasto morto parlante, insieme ad **ALT** è un tasto morto muto. E magari, insieme a **CONTROL** viene emesso il carattere di controllo *Form Feed* indipendentemente dal fatto che sia stato premuto o meno prima un tasto morto muto.

Conclusione

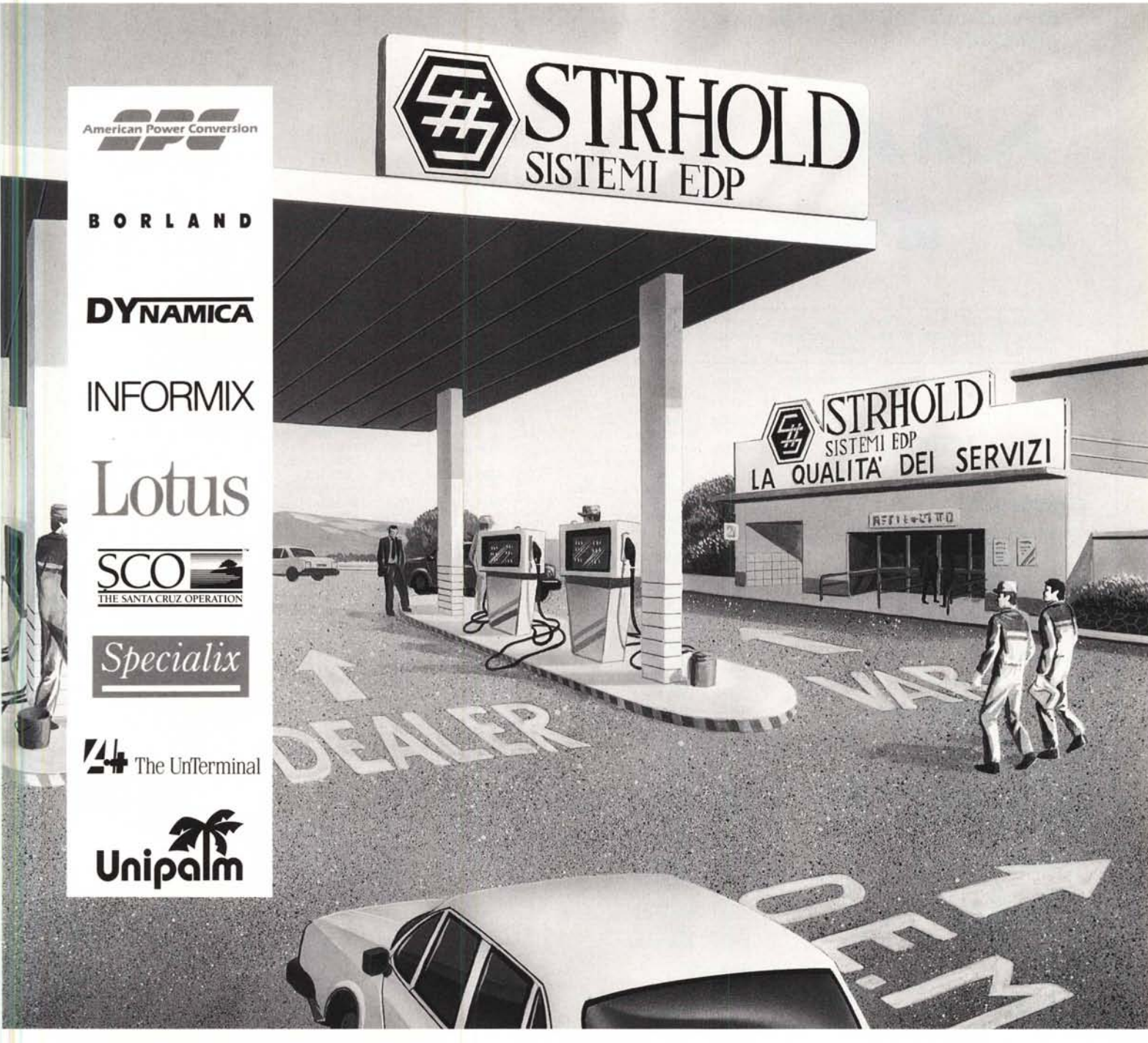
Nella prossima puntata parleremo dei *tasti morti doppi* ed incominceremo a mettere in pratica le nozioni acquisite in questa e nella precedente puntata. Analizzeremo insieme, infatti, la mappa di tastiera italiana fornita dalla *Commodore* insieme al sistema operativo.

Non preoccupatevi quindi se ancora non vi è perfettamente chiara la struttura dei blocchi extra, specialmente quella relativa ai tasti morti. Quando incominceremo ad applicare le nozioni apprese ad un esempio pratico, vedrete che anche gli aspetti più *oscuri* delle mappe di tastiera vi diventeranno chiari. Non solo. Sarete persino in grado di scrivervi la vostra mappa di tastiera senza alcun problema.

Arrivederci fra un mese, allora, e buon lavoro! MC

Dario de Judicibus è raggiungibile tramite MC-link alla casella MC2120.

AREE DI SERVIZIO UNIX IN ITALIA



American Power Conversion

BORLAND

DYNAMICA

INFORMIX

Lotus

SCO THE SANTA CRUZ OPERATION

Specialix

The UnTerminal

Unipalm

STRHOLD®
SISTEMI EDP

Presente al 3° Forum Pubblica Amministrazione
Pad. 22 Stand 32

PIU' DI UN DISTRIBUTORE

Reggio Emilia Via Cipriani 2 - Tel. 0522/792641 - Fax 0522/77846 - Tlx 531059 STRHOLI • Milano Via Dante 4 - Tel. 02/72002222 - Fax 02/72001474 • Torino Via Borgaro 49 - Tel. 011/2296949 - Fax 011/2296939 • Vicenza V.le Mazzini 123/125 - Tel. 0444/324292 - Fax 0444/545248 • Roma Via Pannonia 51 - Tel. 06/7004234 - Fax 06/7001673 • Napoli Via S. Alfonso de' Liguori 3 - Tel. 081/457084 - 290283 - Fax 081/290283 • Bari Via Resistenza 48/B - Tel. 080/228430 - Fax 080/364437 • Catania Via Asiago 35 Tel. 095/376686 - Fax 095/381369 • Palermo Via G. Bonanno 73 - Tel. 091/301650 - Fax 091/347451 • Matelica (MC) Via Circonvallazione 131 - Tel. 0737/787202 - Fax 0737/787200 • Prato (FI) V.le Montegrappa 183 - Tel. 0574/575656 - Fax 0574/572532 • Albisola Capo (SV) C.so Ferrari 109/2 - Tel. 019/487272 - Fax 019/480047 • ODTeam® Consulenza e supporto per SCO Open DeskTop

Il software MS-DOS, Amiga e Macintosh di Pubblico Dominio e Shareware distribuito da

in collaborazione con Microforum

Questo software non può essere venduto a scopo di lucro ma solo distribuito dietro pagamento delle spese vive di supporto, confezionamento, spedizione e gestione del servizio. I programmi classificati Shareware comportano da parte dell'utente l'obbligo morale di corrispondere all'autore un contributo indicato al lancio del programma.

MSDOS

COMUNICAZIONE

CODICE	TITOLO&DESCRIZIONE	REC. HARDWARE
COM/01	ONE TO ONE	mc104
COM/02	PROCOMM Noto programma di comunicazione	Hard disk
COM/03	OMEGA LINK	mc106
COM/04	BACKCOMM	mc103
COM/05	ZIP	mc110
COM/06	FOSSIL DRIVER & .TPU	mc110
COM/07	MAXIHOST	mc110

DATABASE

CODICE	TITOLO&DESCRIZIONE	REC. HARDWARE
DBS/02	VIDEO DATABASE	mc105 Hard disk
DBS/03	HOME MANAGER DataBase, calcolatrice e calendario	Hard disk
DBS/04	MAIL-MONSTER	mc103
DBS/06	PC-FILE+	mc106
DBS/07	TASK MASTER Projet Plaining	
DBS/09	DMS	mc107
DBS/10	ARCHIVIO PARROCCHIALE	mc109
DBS/11	ABSTRACT	mc115
DBS/12	GESTIONE DI BIBLIOTECA	mc116
DBS/13	RICETTARIO	mc116

EDUCATIVO

CODICE	TITOLO&DESCRIZIONE	REC. HARDWARE
EDU/01	ABC FUN KEYS	mc103
EDU/02	COMPUTER TUTOR Auto-apprendimento del computer	
EDU/04	GEOBASE ARCH. GEOGRAFICO	mc109

GIOCO

CODICE	TITOLO&DESCRIZIONE	REC. HARDWARE
GIO/04	ALDO'S ADVENTURE	mc103 EGA/VGA
GIO/05	CAESAR Strategia	BASIC+EGA/VGA
GIO/08	EGAINT	mc104 EGA/VGA
GIO/09	PC-JIGSAW Puzzle	
GIO/11	SUPER PINBALL Super Flipper	
GIO/12	ARK Clone di Arkanoid	EGA/VGA
GIO/13	BANYON WARS Strategia	EGA/VGA
GIO/14	CAPTAIN COSMIC Gioco grafico	EGA/VGA
GIO/16	EGA GOLF Gioco del Golf	EGA/VGA
GIO/17	EGA TREK Star Trek	EGA/VGA
GIO/18	JOUST VGA Gioco da bar	VGA
GIO/19	MINER VGA	mc104 VGA
GIO/21	MOSAIX Puzzle	VGA
GIO/22	OTHELLO EGA	mc103 EGA/VGA
GIO/23	POKER SOLITAIRE Poker da soli	EGA/VGA
GIO/24	QUATRIS Tetris con Bombe ecc.	EGA/VGA
GIO/25	SHARKS Giocate ai sommozzatori	EGA/VGA
GIO/26	SLOT EGA Slot Machine	EGA/VGA
GIO/27	BASSTOUR Pesca d'altura	EGA/VGA
GIO/29	GALACTIC BATTLE Clone di Invaders con sonoro	EGA/VGA
GIO/30	HOUSE OF HORRORS Casa degli orrori	EGA/VGA
GIO/31	NOID Consegnate la pizza all'ultimo piano	EGA/VGA
GIO/32	PINBALL EGA Super Flipper	EGA/VGA
GIO/34	MAHJONG EGA Gioco di società orientale	EGA/VGA
GIO/35	MR.SPOCK	mc105 EGA/VGA
GIO/36	MONUMENTS OF MARS	mc106
GIO/37	PHARAOH'S TOMB	mc106
GIO/38	POKER	mc107 EGA/VGA
GIO/39	NIM	mc108 CGA
GIO/40	TESORI	mc108 CGA
GIO/41	TOMBOLA	mc108 CGA
GIO/42	SMILE	mc109 VGA
GIO/43	CHINESE SOLITARIE	mc111 VGA
GIO/44	TRETRIX	mc111 VGA
GIO/45	SICHUAN	mc112 VGA
GIO/46	EGAWALLS	mc113 EGA
GIO/47	GRID HER	mc113 VGA
GIO/48	BANDIERE!	mc114
GIO/49	PETWORLD	mc114
GIO/50	FORZA4	mc114
GIO/51	CROBOTS	mc115
GIO/52	YAHTZEE!	mc115 VGA
GIO/53	PAROLOSO	mc115

GRAFICA

CODICE	TITOLO&DESCRIZIONE	REC. HARDWARE
GRF/01	FINGER PAINT Programma di disegno	
GRF/02	PC-KEY-DRAW	mc107 CGA
GRF/03	H&P CALENDAR	mc103
GRF/04	PC-DEMO SYSTEM	mc105
GRF/05	GRAPHICWORKSHOP	mc106
GRF/06	SOLAI & TRAVI	mc112
GRF/07	GOSTPAINT	mc112
GRF/08	DKBTrace	mc116

SPREADSHEET

CODICE	TITOLO&DESCRIZIONE	REC. HARDWARE
SPD/01	AS-EASY-AS	mc103
SPD/02	EXPRESS-CALC	mc104
SPD/03	EZ-SPREADSHEET Calcoli di budget	
SPD/04	INSTACALC	mc107
SPD/05	QUEBECALC Spreadsheet 3D	

UTILITY

CODICE	TITOLO&DESCRIZIONE	REC. HARDWARE
UTI/01	PC-DESK-TEAM	mc107
UTI/02	HARD DISK UTILITIES Per gestire l'Hard Disk	Hard Disk
UTI/03	DOS HELP	mc104
UTI/04	DISK SPOOL II	mc103
UTI/05	LOCKTITE Protegge i file con password	
UTI/07	LHARC	mc105
UTI/08	ARJ	mc105
UTI/09	LZEXE	mc105
UTI/10	DIET	mc105
UTI/11	PKLITE	mc105
UTI/12	NEWSPACE	mc105
UTI/13	CATDISK	mc105
UTI/14	POINT&SHOOT	mc105
UTI/15	SHEZ	mc106
UTI/16	ZZAP	mc106
UTI/17	GUARDIAN ANGEL	mc107
UTI/18	STORE	mc107
UTI/19	TXT	mc107
UTI/20	xSET	mc108
UTI/21	ZAPDIR	mc108
UTI/22	UTILITY COLLECTION	mc109
UTI/23	DiR	mc109
UTI/24	CLEANUP	mc111
UTI/25	SAB DISKETTE UTILITY	mc111
UTI/26	TIF2GRAY	mc111
UTI/27	FILLDISK	mc111
UTI/28	ORASCO	mc111
UTI/29	XDIR	mc111
UTI/30	WINCOMMANDER	mc112
UTI/31	MOUSE FORMATTER	mc112
UTI/32	WINZIP	mc112
UTI/33	MOUSE EDITOR	mc113
UTI/34	DEPURA	mc113
UTI/35	DISK FATTER	mc113
UTI/36	POWER DOS	mc116
UTI/37	SIM_LIB	mc114
UTI/38	UTILITY PC	mc114
UTI/39	DBOOK 1.0	mc115
UTI/40	SYSTEM COLOR SETTING	mc116

VARIE

CODICE	TITOLO&DESCRIZIONE	REC. HARDWARE
VAR/01	COMPOSER Per suonare al computer e stampare lo spartito	
VAR/02	CHECK-MATE Controllo delle finanze personali	
VAR/03	PIANO-MAN	mc104
VAR/04	BARTENDER Tutti i cocktail	mc103
VAR/05	DIET DISK La dieta al computer	
VAR/06	ELEMENTARY C Per programmatori in C	
VAR/07	RECIPES	mc104
VAR/08	PERSONAL C COMPILER	mc105
VAR/09	MOUSE.TPU & NEWEXEC	mc106
VAR/10	TSR, PRINT & GESTECC	mc106
VAR/11	ARIANNA	mc106
VAR/12	TOTOPROJET	mc108 CGA
VAR/13	COVER	mc108
VAR/14	CODICE FISCALE	mc109 Hard disk
VAR/15	FLIGHT	mc109
VAR/16	DIZIONARIO INFORMATICO	mc109
VAR/17	ITALIA90	mc110

CODICE	TITOLO&DESCRIZIONE	REC. HARDWARE
VAR/18	TATA-BIGNOMIX UTILITY	mc110
VAR/19	QUICK BASIC ROUTINES	mc110
VAR/20	MICROGESTS	mc113
VAR/21	CALCOLO INDICE ELO	mc113
VAR/22	MENU	mc113
VAR/23	PROMETEO	mc114
VAR/24	IRIS	mc115
VAR/25	MODELLI DI TERRENO	mc115
VAR/26	GESTIONE CAMPIONATO	mc116

WORDPROCESSOR

WPR/02	FREEWORD	mc103
WPR/03	PC-WRITE	mc106
WPR/05	GALAXY	mc104
WPR/06	EDITOR	mc110
WPR/07	NOTEBOOK	mc112
WPR/08	WORDY	mc113
WPR/09	VIX	mc114
WPR/10	BREEZE	mc116

AMIGA

COMUNICAZIONE

AMCO/01	AMIPAC	mc110
AMCO/02	FC FREE COMMUNICATION	mc113

DATABASE

AMDB/01	BADGER	mc113
AMDB/02	VIDEODAT	mc116

GIOCO

AMGI/02	WELLTRIX	mc105
AMGI/03	SYS	mc105
AMGI/04	SCOPONE SCIENTIFICO	mc108
AMGI/05	LA FINE DI UN TIRANNO	mc109
AMGI/06	LA PANTERA SIAMO NOI	mc109
AMGI/07	MEGABALL	mc110
AMGI/08	REVERSI	mc114
AMGI/09	FRIENDLY CARD	mc115
AMGI/10	EQUILOG	mc116

GRAFICA

AMGR/01	PRINTSTUDIO	mc104
AMGR/02	TEXTPAINT	mc105
AMGR/03	SCREENX	mc105
AMGR/04	SETPAL	mc105
AMGR/05	FREEPAINT	mc113
AMGR/06	LABEL MAKER	mc114
AMGR/07	PICTSAVER	mc114

SPREADSHEET

AMSP/01	SPREAD	mc104
AMSP/02	EQUATIONWRITER	mc110

UTILITY

AMUT/01	MACH III	mc104
AMUT/02	RULER	mc104
AMUT/03	HEX	mc104
AMUT/04	MOM	mc104
AMUT/05	CB	mc104
AMUT/06	ZETAVIRUS	mc104
AMUT/07	DIRMASTER	mc105
AMUT/08	KDC	mc105
AMUT/09	XCOPYIII	mc105
AMUT/10	CD2TAPE	mc105
AMUT/11	BBS & LOG	mc106
AMUT/12	UTILITIES	mc106
AMUT/13	VIEW80 II	mc106
AMUT/14	MATCALC	mc106
AMUT/15	ICONMASTER	mc106
AMUT/16	HERMIT	mc106
AMUT/17	TURBO IMPLODER	mc106
AMUT/18	FONTS2PRINTER	mc107
AMUT/19	SVD	mc107
AMUT/20	MC-PROGRAMS	mc107
AMUT/21	CHP&SAVE-PREFS	mc107
AMUT/22	CIDITEIP	mc108
AMUT/23	DISKEDITOR	mc108
AMUT/24	5 UTILITY	mc108
AMUT/25	OROLOGIO PARLANTE	mc108
AMUT/26	LSLAB	mc110
AMUT/27	DIRWORK	mc111
AMUT/28	SCREENMOD	mc111
AMUT/29	SYSINFO	mc111
AMUT/30	SUPERDUPER	mc111
AMUT/31	PRFONT	mc113
AMUT/32	TG	mc113
AMUT/33	ICONS	mc113
AMUT/34	TURBOGIF	mc113
AMUT/35	TMKBP	mc116
AMUT/36	ENVPRINT	mc116

VARIE

AMVR/01	FRACTUS	mc108
AMVR/02	RUBRICA, DACIA & GESTFAT	mc109
AMVR/03	FUNZ3D	mc109
AMVR/04	PLAYSMUS	mc110
AMVR/05	MULTI PLAYER	mc111
AMVR/06	DRAWMAP	mc111
AMVR/07	TOTAMIGA	mc112
AMVR/08	AUTO	mc112
AMVR/09	SOUNDMASTER	mc112
AMVR/10	AMIGA L8	mc112
AMVR/11	FRACTAL	mc112
AMVR/12	SPECTROGRAM	mc114
AMVR/13	CHEMESTHETICS	mc114
AMVR/14	DAY2DAY	mc114
AMVR/15	CEMENTO ARMATO	mc115
AMVR/16	CORTES	mc115
AMVR/17	TUCANENTA	mc115
AMVR/18	CALORIEBASE	mc115
AMVR/19	MperM	mc116

MACINTOSH

COMUNICAZIONE

MICO/01	RED RYDER	mc110
MICO/02	ZTERM	mc115
MICO/03	ARCMAC	mc116

EDUCATIVO

MIED/01	KID PIX	mc107
MIED/02	NUMBER TALK	mc107
MIED/03	ALPHA TALK	mc107

GIOCO

MIGI/01	STELLA OBSCURA	mc106
MIGI/02	PARARENA	mc106
MIGI/03	VIDEO POKER FOR FUN	mc106
MIGI/04	SPACE STATION PHETA	mc106
MIGI/05	STRATEGO	mc106
MIGI/06	THE LAWNZAPPER	mc107
MIGI/07	MACTRIS	mc107
MIGI/08	CANFIELD	mc107
MIGI/09	YAHTZEE	mc108
MIGI/10	GLIDER	mc108
MIGI/11	MACNINJA	mc108
MIGI/12	GLIPHA	mc108
MIGI/13	MONOPOLY	mc109
MIGI/14	GOLF	mc109
MIGI/15	WHEEL	mc109
MIGI/16	GUNSHY	mc109
MIGI/17	MEGAROIDS	mc110
MIGI/18	SHUFFLEPUCK	mc110
MIGI/19	CRIMINALS	mc111
MIGI/20	SQUIX	mc112
MIGI/21	HOTEL CAPER	mc112
MGI/22	RISIKO	mc115
MGI/23	SPACE INVADERS	mc115
MGI/24	CONTINUUM	mc115
MGI/25	QUESTER	mc115
MGI/26	SCEPTERS	mc115
MIGI	SCEPTERS	mc116

GRAFICA

MIGR/01	CALENDAR MAKER	mc106

SPREADSHEET

MISP/01	BIPLANE	mc112

STACK

MISK/01	FOOD 1	mc111
MISK/02	BUSINESS 1	mc111
MISK/03	SOUND 1	mc111

UTILITY

MIUT/01	OLIVER'S BUTTONS	mc107
MIUT/02	POPCHAR	mc107
MIUT/03	RAMDISK	mc108
MIUT/04	SCROLL2	mc109
MIUT/05	DECK EDITOR	mc109
MIUT/06	BANNER MAKER	mc110
MIUT/07	SPEEDOMETER	mc110
MIUT/08	LOODLE	mc112
MIUT/09	FAST FORMAT	mc112
MIUT/10	SOUND MASTER	mc112
MIUT/11	STUFFIT CLASSIC	mc112
MIUT/12	DISKDUP+	mc114
MIUT/13	DTPPRINTER	mc114
MIUT/14	FOLDER FROM HELL	mc114
MIUT/15	NUMBERCRUNCH	mc114
MIUT/16	PASTE-IT	mc114
MIUT/17	SAVE A TREE	mc114
MIUT/18	MACBINARY	mc114
MIUT/19	DOCMAKER	mc115
MIUT/20	APOLLO	mc116

VARIE

MIVR/01	RIDICOLO	mc108
MIVR/02	ELIZA	mc109
MIVR/03	HYPERSTAR	mc113
MIVR/04	MEGALOMANIA	mc116

Compilare e spedire a: ***MCmicrocomputer***

Desidero acquistare il software di seguito elencato al prezzo di **L. 8.000 a titolo (ordine minimo: tre titoli).** Per l'ordinazione inviare l'importo (a mezzo assegno, c/c o vaglia postale) alla: Technimedia srl, Via Carlo Perrier 9, 00157 Roma.

dischetti da ☐ 3.5" ☐ 5.25"

Codici:

Totale dischi ☐ x 8.000=Lire

Nome e Cognome

Indirizzo

CAP/Città

Telefono

MCmicrocomputer non offre alcuna garanzia e non si assume alcuna responsabilità sugli eventuali danni diretti o indiretti derivanti dall'utilizzo del software distribuito

HD SCSI CONTROLLER PER A2000

SYNTHESIS HARDITAL 0-8 MB	L. 240000
PER OGNI MB AGGIUNGERE	L. 100000
SERIE II GVP 0-8 MB	L. 410000
PER OGNI MB AGGGIUNGERE	L. 100000
A2091 COMMODORE 0-2 MB	L. 280000
PER OGNI MB AGGIUNGERE	L. 100000
ADSCSI ICD	L. 240000
DATA FLYER	L. 170000

HARD DISK SCSI

QUANTUM 52 MB-11ms	L. 440000
QUANTUM 80 MB-11ms	L. 790000
QUANTUM 105 MB-11ms	L. 890000
QUANTUM 210 MB-11ms	L. 1210000

HD SCSI PER A500

SYNTHESIS HARDITAL 0-8MB CON QUANTUM 52 MB-11ms	L. 580000
PER OGNI MB AGGIUNGERE	L. 100000
A 590 COMMODORE 0-2MB/20MB	L. 620000
PER OGNI MB AGGIUNGERE	L. 100000

HD IDE PER A500/1000/2000

DOTTO HARDITAL	L. 150000

HD IDE-ATBUS PER DOTTO

PRAIRIETEK 20MB-2,5"	L. 490000
PRAIRIETEK 40MB-2,5"	L. 790000
QUANTUM 40MB-3,5"	L. 420000

I COMPUTER AMIGA

AMIGA 500 CON GARANZIA COMM.ITALIA	L. 629000
COME SOPRA MA CON 1MB	L. 690000
COME SOPRA MA CON 2,5MB	L. 849000
AMIGA 500 PLUS CON 2.0 E 1MB RAM	L. 710000
CDTV COMMODORE	L. 1040000
AMIGA 2000 CON GAR. COMM. ITALIA	L. 1190000
COME SOPRA MA CON HD SCSI QUANTUM 52MB E 3MB RAM	L. 2190000
AMIGA 3000 25MHZ E HD QUANTUM 52MB	L. 4760000
COME SOPRA MA CON HD QUANTUM 105 MB	L. 5390000

I DISCHETTI

DISCHETTI SONY, BULK, DS-DD, DA 3,5"
1 L. 790 - 10 L. 690 - 100 L. 640 - 1000 L. 560

SCHEDE AUDIO-VIDEO

GENLOCK CARD A2300 COMMODORE	L. 390000
FLIKER FIXER A2000	L. 310000
FLIKER FIXER 500 INTERNA	L. 310000
MONITOR MULTISYNC 14" PER FLIKER FIXER	L. 490000
COLORBURST MAST PER A500/1000/2000	L. 990000

ESPANSIONI PER A2000

SYNTHESIS HARDITAL 2MB L. 340000
4MB L. 520000-6MB L. 700000-8MB L. 840000
SUPEROTTO HARDITAL 2MB L. 280000
4MB L. 460000 - 8MB L. 780000
A2058 COMMODORE 2MB L. 790000

ESPANSIONI PER A500

SYNTHESIS HARDITAL 2MB L. 380000 4MB L. 580000-6MB L. 740000-8MB L. 880000
INSIDER 05 HARDITAL 512 KB L. 59000 CON CLOCK L. 74000
INSIDER 1 HARDITAL 1MB PER A500 PLUS L. 99000
INSIDER 2 HARDITAL 2MB L. 259000
INSIDER 4 HARDITAL 4MB L. 390000

ESPANSIONI CHIP RAM PER A500 E A2000

MEGA AGNUS HARDITAL 2MB DI CHIP RAM L. 349000

ESPANSIONI PER A3000

RAM ZIP 1MBX4-2MB L. 190000-4 MB L. 320000-8MB L. 620000

I MONITOR

COMMODORE 1084S	L. 450000
PHILIPS 8833	L. 430000

LE STAMPANTI

1230 COMMODORE	L. 315000
1550 COLOR COMM.	L. 410000

COMPUTER CENTER

PER ORDINAZIONI E INFORMAZIONI:
VIA FORZE ARMATE 260
20152 MILANO
TEL 02 48016309/4890213
FAX 02 4890213

TUTTI I PREZZI SONO IVA COMPRESA

I PERSONAL COMPUTER IBM COMPATIBILI

HAR286-16 L.M.21 MHZ-CPU 286 A 0 WAIT STATE-1 MB RAM-1 DRIVE 1,44 MB 3,5"-2 SERIALI 1 PARALLELA-CASE CON DISPLAY DESK TOP O MONITOWER CON ALIM DA 200W-CONTROLLER PER 2 FD E 2 HARD DISK IDE AT BUS-SCHEDA VGA 800X600- TASTIERA ESTESA DA 102 TASTI-DR.DOS 5.0 CON MAN. ITALIANO A CORREDO. L. 690000

HAR 286-20. COME SOPRA MA CON CPU 286/20 L.M. 26MHZ L. 730000

HAR 386-SX16. COME SOPRA MA CON CPU 386 SX16 L. 849000

HAR 386-SX20. COME SOPRA MA CON CPU 386 SX20 L. 899000

HAR 386-25 L.M. 33MHZ-CPU 386/25 A 0 WAIT STATE-4 MB RAM-1 DRIVE 1,44" MB 3,5"- 2 SERIALI 1 PARALLELA 1 GAME-CASE CON DISPLAY DESK TOP O MINITOWER CON ALIM. 200W-CONTROLLER PER 2 FDD E 2 HARD DISK IDE AT-BUS-SCHEDA VGA 800X600-TASTIERA ESTESA 102 TASTI. DR. DOS 5.0 E MANU. ITALIANO A CORREDO. L. 1390000

HAR 386-33 L.M. 56MHZ. COME SOPRA MA CON CPU 386/33 E 64KB CACHE L. 1590000

HAR 486-33-SX20 L.M. 92MHZ COME SOPRA MA CON CPU 486SX20 L. 1690000

HAR 486-33 L.M. 151MHZ. COME SOPRA MA CON CPU 486/33 L. 1990000

NOTEBOOK CPU 386/20-LCD DISPLAY RETROILLUMINATO CON RISOL. VGA 640X480-1MB RAM- 1 HD 20MB-1 DRIVE 1,44" MB-CON ALIM. BATTERIE, BORSA TRASPORTO. L. 2990000

COME SOPRA MA CON HD DA 60MB L. 3490000

ACCESSORI E PERIFERICHE

MOTHER BOARD-286-16 L.M.20MHZ	L. 179000
MOTHER BOARD-286-20 L.M.26MHZ	L. 210000
MOTHER BOARD-386-SX16 L.M.21MHZ	L. 349000
MOTHER BOARD-386SX20 L.M.26MHZ	L. 419000
MOTHER BOARD-386/25 L.M.33MHZ	L. 570000
MOTHER BOARD-386/33 L.M.56MHZ	L. 749000
MOTHER BOARD-486SX20 L.M.92MHZ	L. 890000
MOTHER BOARD-486/33 L.M.151MHZ	L. 1340000
MOTHER BOARD-486/50 L.M.230MHZ	L. 1690000
COPROCESSORE INTEL 80287 10/12/16/20MHZ	L. 190000
COPROCESSORE INTEL 80387SX20MHZ	L. 290000
COPROCESSORE INTEL 80387/33MHZ	L. 430000
DRIVE 1,2 MB-5,1/4"	L. 125000
HARD DISK 40MB-17ms IDE AT-BUS	L. 390000
HARD DISK 130MB-17ms IDE AT-BUS	L. 720000
HARD DISK 210MB-15ms IDE AT-BUS	L. 1190000
CONTROLLER PER 2 HD AT-BUS	L. 29000
CONTR. 2FDD+2HD+2SER+1PAR+1GAME	L. 49000
MONITOR 14" VGA B/N SCH.PIATTO	L. 190000
MONITOR 14" SUPER VGA COLORI TRISCAN 1024X768	L. 549000
MONITOR 19" SUPER VGA COLORI TRISCAN 1024X768	L. 1590000
SCHEDA VGA 256 K 800X600	L. 89000
SCHEDA VGA 1024X768 1MB	L. 190000
MOUSE	L. 40000
HANDY SCANNER 200/300/400 DPI	L. 290000
HANDY SCANNER COLORI	L. 840000

SHEDE ACCELERATRICI

BANG 2081/2 HARDITAL CON 68020 E 68881 A 16 MHZ PER A 500/2000 L. 290000

BIG BANG HARDITAL C ON 68030 E 68882 A 25MHZ E 2 MB RAM PER A500/2000 L. 990000

COME SOPRA MA CON 4MB L. 134000-CON 8MB L.1690000. CON CLOCK A 50 MHZ AGGIUNGERE L. 990000

A2630 COMMODORE CON 68030, 68882 A 25 MHZ E 2 MB RAM L. 1760000

COME SOPRA MA CON 4MB RAM L. 2050000

COMBO GVP CON 60030, 68882 A 22MHZ 1MB RAM E CONTR. HD L. 1540000

COMBO GVP CON 68030, 68882 A 33MHZ 4 MB RAM E CONTR. HD L. 2690000

SUPER BIG BANG HARDITAL CON 68030,68882 A 25MHZ E CONTR. HD L. 990000. PER OGNI MB DI RAM AGGIUNGERE L. 120000

COME SOPRA MA CON 68030 E 68882 A 52MHZ L. 1990000

SCHEDA V XL CON 68030 E 60882 A 25MHz CON POSSIBLITÀ ESP. MEM 2MB 32 BIT PREZZO CHIEDERE

FUSION FORTY RCS CON 68040, 4 MB RAM L. 3990000

PROCESSORI

68000 16 MHZ L. 29000-68010 L. 24000-68020 16 MHZ L. 140000-68030 25MHZ L. 230000-68030 50MHZ L. 390000-68040 25MHZ L. 800000

INTEGRATI AMIGA

KICKROM 2.0 PER A500/2000 L. 99000
8373 SUPER DENISE ECS L. 129000
8372A FAT AGNUS 1MB L. 120000
8372B FATTEST AGNUS 2MB L. 149000
5719 GARY L. 29000

GLI EMULATORI MS-DOS

AT ONCE VORTEX CON EM.VGA L. 329000
AT ONCE PLUS CON 512K CACHE CHIEDERE AT ONCE ADAPTER PER A2000 L. 120000
JANUS XT COMMODORE L. 560000
JANUS AT COMMODORE L. 849000

I DRIVE

ADRIVE-DA 3,5" ESTERNO PER A500/1000/2000 CON INTERRUTTORE E PASSANTE L. 119000
ADRIVE 2000-INTERNO PER A2000 COMPLETO DI KIT L. 99000
SUPERDRIVE-ESTERNO PER A500/1000/2000 CON TASTO COPIATORE E ANTIVIRUS L. 139000

HARDITAL

SHOW ROOM VIA G. CANTONI 12
20144 MILANO
FERMATA METRO PAGANO
TEL 02 4983457-4983462