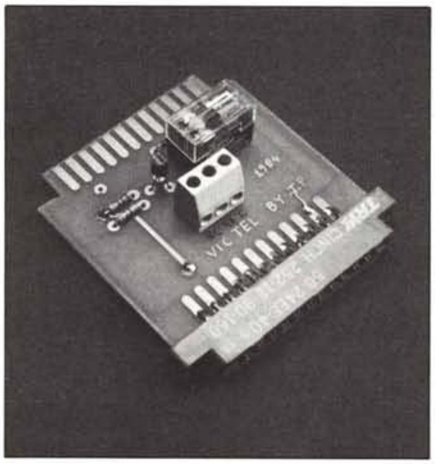
software COMMODORE 64

A cura di Leo Sorge

Nelle pagine di VIC da Zero, la rubrica hard-soft che MC pubblica da svariati numeri, era stato presentato un progettino che, tramite una scheda pilotata da un programma in Basic, permetteva al VIC di generare le sequenze necessarie ad effettuare la chiamata telefonica (MC 29, pagg. 104-108). Dopo un colloquio avuto con Tommaso Pantuso, il prof. Giacomo lo Vecchio, proprietario di un 64, ha operato al software le modifiche necessarie per adoperare anche sul 64 la schedina del VICTEL, e ce le ha inviate: eccovele quindi nella prima proposta di questo mese, insieme ad un breve sunto del citato articolo di MC 29, cui rimandiamo il lettore scrupoloso.

Come seconda parte vi offriamo un lungo gioco, Chase, che differisce fondamentalmente da tutti quelli mostrati finora: si tratta non di un arcade, del quale non avrebbe la velocità, né di un adventure, bensì di un gioco di stategia, ove quello che importa è il metodo risolutivo.



Comtel

di Giacomo Lo Vecchio - Messina

L'hardware

Quando usate il telefono, il disco combinatore, quello su cui voi impostate le singole cifre, ruotando, genera un segnale

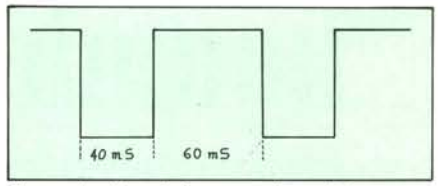
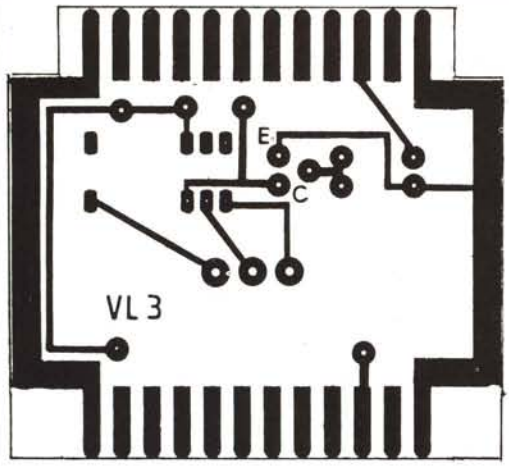


Figura 1 - Tempi di un segnale codificante una parte di una sequenza telefonica. I contatti del disco combinatore aprono la linea per 40 mS e la tengono chiusa per 60 mS per ogni unità costituente la cifra (ad esempio per il "5" questo processo viene ripetuto cinque volte).

seriale, che è composto da una successione di forme d'onda elementari come quella mostrata in figura 1 (40 millisecondi di "silenzio" seguiti da 60 ms di segnale); ogni cifra viene codificata con un numero di

```
0 POKE56579,128:POKE56577,0:PRINTCHR$(147):GOSUB730
1 IFA$="L"THENRETURN
2 GOTO10
3 POKE56579,128:POKE56577,0:GOSUB730
4 IFA$="M"THENRETURN
9 :
10 REM ******************************
11 REM *****                    *****
12 REM *****       COMTEL       *****
13 REM *****                    *****
16 REM ******************************
20 :
34 GOTO37
35 LOAD"INDIRIZZI",8
36 END
37 CLR
40 REM **** MAIN PROGRAM ****
50 POKE56579,128:POKE56577,0:PRINTCHR$(147):DIMA$(17)
60 N=0:POKE53280,6:POKE53281,
70 PRINT"▮▮ COMPORRE IL NUMERO  ■
75 PRINT"▮TELEFONICO DESIDERATO▮■" :PRINT:PRINT
80 PRINT"▮ M RIPETIZIONE        ■":PRINT
90 PRINT"▮ C CANCELLAZIONE      ■":PRINT
100 PRINT"▮ T START TIMER       ■":PRINT
110 PRINT"▮ S STOP TIM+RESET LIN■":PRINT
120 PRINT"▮ L RESET LINEA       ■":PRINT
130 PRINT"▮ R RESET PROGRAMMA   ■":PRINT
140 PRINT"▮ B BUZZER            ■":PRINT
145 PRINT"▮ I CARICA INDIRIZZI  ■":PRINT
150 GETA$:IFA$=""THEN150
160 IFA$="M"THENPRINTCHR$(13):GOSUB3:GOSUB350:GOTO150
170 IFA$="C"THENCLR:PRINTCHR$(67):PRINTCHR$(17):GOTO150
180 IFA$="T"THEN530
190 IFA$="L"THENGOSUB0:GOTO70
200 IFA$="R"THENPRINTCHR$(147):CLR:GOSUB0:GOTO70
210 IFA$="B"THENGOSUB670:GOTO150
215 IFA$="I"THENPRINTCHR$(147):GOSUB1000:GOTO35
220 IFASC(A$)<48 ORASC(A$)>57THEN150
230 IFA$="0"THENA$="10"
240 REM
250 REM *** POSIZIONAMENTO DI ACR E CARICAMENTO DEI LATCH H ED L
260 REM
270 POKE56579,128
280 POKE56577,0:GOSUB410
300 REM
310 REM *** TRASFORMA LA VARIABILE A$ IN UN ELEMENTO DI UN VETTORE
320 N=N+1:A$(N)=A$:GOSUB490
330 GOSUB470
340 GOTO150
350 REM *** CICLO RIPETIZIONE NUMERO DOPO LA PRESSIONE DEL TASTO M
360 FORL=1TON
370 A$=A$(L)
380 GOSUB410:GOSUB470:GOSUB490
390 NEXTL
400 RETURN
410 REM *** GENERA UN NUMERO D'IMPULSI 40/60 PARI A VAL(A$)
420 FORS=1TOVAL(A$)
430 POKE56579,128:POKE56577,0:FORI=1TO21:NEXTI
431 POKE56579,  0:POKE56577,128:FORY=1TO14:NEXTY:NEXTS
440 POKE56579,128:POKE56577,0
450 RETURN
460 REM *** GENERA LA PAUSA INTERDIGITALE E MOSTRA IL NUMERO
470 FORT=1TO500:NEXTT
480 RETURN
490 IFA$="10"THENA$="0"
500 PRINTA$;
505 POKE56579,128:POKE56577,0
510 RETURN
520 REM *** CRONOMETRO E BEEP DI SEGNALAZIONE ***
530 TI$="000000"
540 POKE53280, 6:POKE53281,7:P$=" ":PRINT
550 PRINT"————————"
560 PRINT"H  M  S"
570 PRINT"‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾‾"
580 H$=MID$(TI$,1,2)+P$+MID$(TI$,3,2)+P$+MID$(TI$,5,2)
590 IFPEEK(203)=10THENPRINT"J":GOTO100
600 PRINTH$
610 PRINT"____________________TJ"
620 H=VAL(MID$(H$,7,2))
630 IFINT(H/30)-H/30=0THEN GOSUB 960
640 GETA$:IFA$="S"THENPRINTH$:CLR:GOSUB730:GOTO70
650 GOTO580
660 REM *** BUZZER ***
670 FORI=1TO10
680 POKE56577,128:POKE56579,0:FORX=1TO21:NEXTX
685 POKE56577,  0:POKE56579,128:FORY=1TO14:NEXTY:NEXTI
690 RETURN
720 REM ************************
730 REM *** RIPRISTINO LINEA ***
740 POKE56577,128
750 POKE56579,0
770 FORI=1TO1000:NEXT
780 POKE56579,128
790 POKE56577,0
795 FORI=1TO1500:NEXTI
800 RETURN
930 REM
940 REM BEEP
950 REM
960 VV=54272:POKE54296,15:POKE54295,0
970 POKEVV+6,0:POKEVV+5,31:POKEVV+1,180:POKEVV+4,33
975 FORNM=1TO500:NEXTNM
976 POKEVV+4,0
980 RETURN
1000 PRINT"▮▮▮▮▮▮▮▮▮               ATTENDERE"
1010 PRINT"  LA CHIAMATA DEL PROGRAMMA INDIRIZZI"
1020 PRINT"             E RUBRICA TELEFONO"
1040 RETURN
```



Vista delle piste del lato inferiore del circuito stampato che alloggia i componenti del VICTEL in grandezza naturale.

Lato superiore del circuito stampato. Su tale circuito si può saldare un connettore per connetterlo alla user port oppure può essere inserito (dall'altra parte) nel bus VL1 del VICLAB. I triangolini indicano i pin corrispondenti ai capi della bobina del relè.

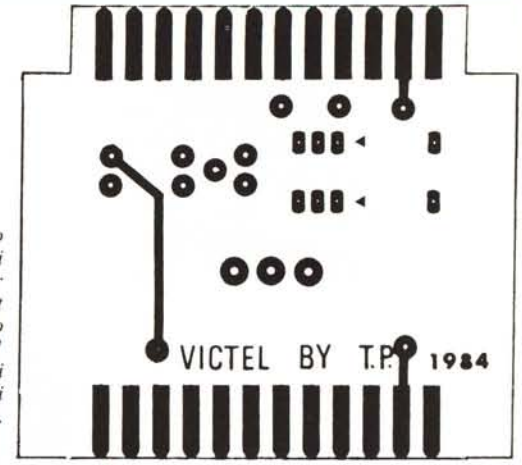


Figura 2

impulsi pari al suo valore (1 per l'uno, 2 per il due, etc) tranne lo zero cui corrispondono 10 impulsi. Le cifre vanno poi separate da un periodo di silenzio (detto pausa interdigitale) di durata compresa tra 450 e 550 ms.

Il problema della generazione di sequenze tramite computer si scompone in due passi: individuazione di un generatore di segnali opportuni, sua gestione da programma. La gestione dell'ingresso/uscita dei dati, nel VIC e nel 64, è affidata a circuiti integrati particolarmente complessi (il VIA 6522 per il VIC; il CIA 6526 per il 64) che sono largamente compatibili, tanto che la schedina del VICTEL, basata sui segnali della porta utente del VIC, va benissimo e senza modifiche anche sul 64.

Questi due circuiti integrati possono essere programmati in modo da tirar fuori degli impulsi per periodi prestabiliti, ma non possono essere usati direttamente in quanto elettricamente incompatibili con le caratteristiche di una linea telefonica (e se un 6522 si trova a circa 20.000 lire, il 6526, essendo di produzione della MOS-Commodore, non è in vendita, quindi non si deve rompere). Il metodo usato si basa su un relé pilotato dal computer tramite un transistor. La figura 2 mostra le due facce del circuito stampato che va connesso direttamente alla user port del 64; la figura 3 ne mostra lo schema elettrico e le connessioni con la linea telefonica; la figura 4 porta l'elenco dei componenti.

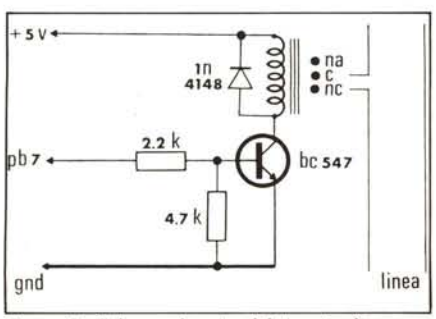


Figura 3 - Schema elettrico del circuito di comando. La linea si chiude da una parte sull'apparecchio telefonico e dall'altra va verso la centrale. La tensione di lavoro del relè da noi usato è di 6 volt.

Elenco componenti

R1	2.2 kΩ
R2	4.7 kΩ
Diodo	1N 4148
Tr	BC 547

Figura 4

Il software

Il programma proposto dal prof. Lo Vecchio è ricavato da quello presentato da Tommaso Pantuso (a pag. 107 del già citato MC 29), nel quale sono stati modificati gli indirizzi dei registri, oltre ad alcune modifiche minori. Va citato l'inserimento dell'opzione "I", che carica da disco un eventuale programma di agenda telefonica: in questo modo viene cancellato Comtel, che quindi andrà caricato un'altra volta (magari tramite un'opzione sul menu dell'agenda) dopo la consultazione dei dati contenuti nel programma di nome INDIRIZZI (vedi riga 35).

Chase 64

di Dario Accornero - Roma

Il programma che vi propongo è un gioco, ma non è un adventure, né un vero e proprio arcade, né intendeva esserlo. È un gioco di inseguimento, fra il vostro ometto e un robot nemico (dopo il 2° quadro si inizia ad odiarlo, verso l'ultimo dareste l'anima per pochi soldi pur di non vederlo più, comunque il gioco è simpatico). Non starò a dilungarmi su come si gioca, trucchi e roba del genere: preferisco descrivervi alcune cosette che ho imparato proteggendo il mio gioco. Proprio così, proteggendo. Poiché una software house non può far copiare i propri programmi al primo che arriva. Dopo vari e disperati tentativi (ne elenco uno: ottenere l'effetto di un RUN chiedendo il LIST!!!), sono riuscito a proteggere il programma da RUN / STOP&RESTORE, da LIST e da SAVE in maniera non permanente. Una nuova IRQ mi ha consentito di raggiungere il livello desiderato; poi però ho deciso di non includere questa nuova IRQ nel programma (a proposito, si chiama CHASE 64), poiché i DATA sono già tanti, e per esperienza personale so quanto siano odiati dai fruitori di software su rivista. Mi sono accontentato di realizzare una protezione non permanente, tra l'altro per un gioco che non ne meritava una permanente. Vi dirò i miei trucchi, ma non tutti: altrimenti, che software house sarei?

In breve, ecco le poke necessarie per proteggere un programma da RUN / STOP&RESTORE, LIST e SAVE:

POKE	PROTEZIONE SOMMARIA
POKE808,225	tasti RUN/STOP&RESTORE
POKE775,225	questa la sapete già
POKE774, lobyte	———
POKE819,244	comando SAVE
POKE818,lobyte	———

POKE	PROTEZ. PERSONALIZZATA
POKE808,225	impossibile, vettore hardware
POKE775,225	POKE775, hibyte new routine
POKE 774, lobyte	low byte new routine
POKE819, hibyte	POKE819, hibyte new routine
new routine	low byte new routine

Le protezioni da RUN e da LOAD cercatevele voi, sono facili. Queste dovrebbero bastare per proteggere un programma, a patto che il RUN sia dato. Consiglio di scrivere un programma caricatore, proteggere (tasti SHIFT/INST) le linee dove proteggete il prossimo, e far rimanere allibiti tutti i pirati e gli sprotettori (potranno solo listare il caricatore!).

Le protezioni da me suggerite inibiscono i detti comandi rimpiazzando il vettore che il S.O. legge quando la CHARGET o la HANDLE NEW BASIC LINE routine sono eseguite. Ogni volta, verrà eseguita la propria routine, che magari scrive solo: NO., oppure: NO, NON MI VA (un po' cafone).

La mia scrive:

[C] DARIOSOFT - PROGRAMMA PROTETTO

```
1 REM ***********************************
2 REM ***      C H A S E - 6 4      ***
3 REM *** [C] 1984 BY DARIOSOFTWARE ***
4 REM ***********************************
8 :
9 POKE53281,3:POKE53280,4:PRINT"■"CHR$(8):POKE55,255:POKE56,47
10 A$="*** SETTING UP * - * - * PLEASE WAIT ***"
11 PRINT"■■■■■■■■■■■■**********************************************";:PRINTA$;
12 PRINT"****************************************";
15 POKE774,20:POKE775,205:POKE818,20:POKE819,205:POKE808,225
16 FORI=49152TO49185:READA:POKEI,A:NEXT:POKE56334,0:POKE1,51:SYS49152
20 POKE1,55:POKE56334,1:POKE53272,29
30 FORI=12568TO12615:READA:POKEI,A:POKEI+1024,255-A:NEXT
34 FORI=12808TO12887:READA:POKEI,A:POKEI+1024,255-A:NEXT
35 FORI=12552TO12559:READA:POKEI,A:POKEI+1024,255-A:NEXT
36 FORI=12768TO12799:READA:POKEI,A:POKEI+1024,255-A:NEXT
45 FORI=49152TO49168:READA:POKEI,A:NEXT
50 FORI=50000TO50050:READA:POKEI,A:NEXT
53 FORI=51000TO51050:READA:POKEI,A:NEXT
55 FORI=704TO737:READA:POKEI,A:NEXT
57 FORI=52500TO52528:READA:POKEI,A:NEXT
62 C=1024:SC=0:AS=3:S=54272:CN=0:MI=20:X=20:Y=12:A=1:B=3:LV=1:CC=0:S1=S+7:K=0
64 GOSUB5000
67 POKE53281,3
68 PRINT"■■■[C] ↑|‾‾_ 1984"TAB(16)"■■SCORE:■"SCTAB(30)"■■LIVES:■"AS
70 POKE53281,14:PRINT"■■";:FORI=1TOAS:PRINT"$";:NEXT:PRINT"■■■■ — #";
71 PRINTTAB(11)"| K \"TAB(16)"■■TOPSC:■"HITAB(30)"■■LEVEL:■"LV
75 FORI=1104TO1143:POKEI,63:POKEI+S,11:NEXT
77 FORI=1144TO1983STEP40:POKEI,63:POKEI+39,63:POKEI+S,11:POKEI+39+S,11
80 NEXT:FORI=1984TO2023:POKEI,63:POKEI+S,11:NEXT:PRINT"◣":FORI=1TOMI
87 X1=INT(38*RND(1))+1:Y1=INT(21*RND(1))+3
88 IFX1=20ANDY1=12ORX1=1ANDY1=3THEN87
89 IF(X1>36ANDY1>22)OR(X1<2ANDY1>22)OR(X1>36ANDY1<4)OR(X1<2ANDY1<4)THEN87
90 POKE53245,X1:POKE53246,Y1:POKE53247,37:SYS49152:NEXT
95 PRINT"■":POKE53245,A:POKE53246,B:POKE53247,35:SYS49152
97 PRINT"■":POKE53245,X:POKE53246,Y:POKE53247,36:SYS49152
98 FORI=0TO24:POKES+I,0:NEXT:POKES+5,41:POKES+6,89:POKES+24,15
100 POKES1,255:POKES1+5,41:POKES1+6,89:POKES1+4,33:FORI=0TO999:NEXT
200 JV=PEEK(56320):TA=PEEK(197)
203 IFJV=111ORTA=60THENGOSUB4000
205 IFJV=123ORTA=46THENXL=-1:YL=0:CH=38:GOTO230
210 IFJV=119ORTA=54THENXL=1:YL=0:CH=39:GOTO230
215 IFJV=126ORTA=8THENXL=0:YL=-1:CH=36:GOTO230
220 IFJV=125ORTA=23THENXL=0:YL=1:CH=36:GOTO230
221 IFJV=121THENXL=-1:YL=1:CH=36:GOTO230
222 IFJV=117THENXL=1:YL=1:CH=36:GOTO230
223 IFJV=118THENXL=1:YL=-1:CH=36:GOTO230
224 IFJV=122THENXL=-1:YL=-1:CH=36:GOTO230
225 GOTO300
230 PE=PEEK(C+(X+XL)+40*(Y+YL)):IFPE=63THENSN=50:WA=17:GOSUB900:GOTO301
235 IFPE=37ORPE=35ORPE=33THENGOSUB950:GOTO3000
237 IFPE=62THENGOSUB950:GOTO2000
238 IFPE=60THENGOSUB700
239 IFPE=61THENSC=SC-200:GOTO10000
245 PRINT"◣":POKE53245,X:POKE53246,Y:POKE53247,32:SYS49152:PRINT"■"
250 X=X+XL:Y=Y+YL:POKE53245,X:POKE53246,Y:POKE53247,CH:SYS49152
300 IFSFTHENSF=0:SN=50:WA=33:GOSUB900:GOTO305
301 SF=-1:SN=33:WA=33:GOSUB900
305 P(K)=0:K=K+1:IFK>2THENK=0
307 P(K)=-1:CS=INT(RND(0)*3)
308 IFP(CS)THEN355
309 IFX<ATHENAL=-1
310 IFX>ATHENAL=1
315 IFX=ATHENAL=0
320 IFY<BTHENBL=-1
325 IFY>BTHENBL=1
330 IFY=BTHENBL=0
335 P1=PEEK(C+(A+AL)+40*(B+BL)):IFP1=37THENSN=50:WA=129:GOSUB900:GOSUB800
336 IFENTHENEN=0:GOTO10000
340 IFP1=36ORP1=38ORP1=39THENGOSUB950:GOTO3000
341 IFP1=33THENGOSUB500
342 IFP1=62THENFORT=0TO4:GOSUB500:NEXT
343 IFP1=60THENFORT=0TO1:GOSUB500:NEXT
345 PRINT"◣":POKE53245,A:POKE53246,B:POKE53247,32:SYS49152:PRINT"■"
350 A=A+AL:B=B+BL:POKE53245,A:POKE53246,B:POKE53247,35:SYS49152
355 CC=CC+1
360 IFCC=1THENPOKE12575,129
365 IFCC=2THENPOKE12575,36:CC=0
370 VL=RND(1)
371 IFP(CS)THEN200
375 IFVL>.57ANDVL<.62THEN400
376 IFVL>.11ANDVL<.12THEN410
377 IFVL>.08ANDVL<.09THEN420
380 GOTO200
400 PRINT"■":CA=A-AL:CB=B-BL
405 POKE53245,CA:POKE53246,CB:POKE53247,33:SYS49152:GOTO200
410 PRINT"■":CA=A-AL:CB=B-BL:POKE53245,CA:POKE53246,CB:POKE53247,60:SYS49152
415 GOTO200
420 PRINT"■":CA=A-AL:CB=B-BL:POKE53245,CA:POKE53246,CB:POKE53247,62:SYS49152
425 GOTO200
500 SYS50000:SC=SC+50
505 PRINT"■"TAB(22)"■■"SC:SYS50000:GOSUB8000
510 IFFLTHENHI=SC:PRINT"■■"TAB(22)"■"HI
515 RETURN
700 SN=80:WA=17:GOSUB900
705 PRINT"■":POKE53245,20:POKE53246,2:POKE53247,61
710 SYS49152:POKE53246,24:SYS49152
715 POKE53245,0:POKE53246,13:SYS49152:POKE53245,39:SYS49152:RETURN
800 SYS50000:SC=SC+10
805 IFSC>=HITHENHI=SC:FL=-1
810 PRINT"■"TAB(22)"■■"SC
813 SYS50000:GOSUB8000
815 IFFLTHENPRINT"■■"TAB(22)"■"HI
820 CN=CN+1
825 IFCN=MITHENEN=-1
830 RETURN
900 POKES+4,WA:FORI=0TO2:POKES+1,SN
905 POKES1+1,255-SN:NEXT:POKES+1,0:POKES+4,WA-1:POKES1+1,0:RETURN
950 POKES+4,129:POKES1+4,129
955 FORI=240TO10STEP-.9:POKES+1,I:POKES1+1,250-I:NEXT:SYS50000
960 FORI=0TO50STEP.9:POKES+1,5+INT(PEEK(S+27)/2)
965 POKES1+1,1:NEXT:SYS50000:POKES+1,0:POKES1+1,0:POKES+4,0:POKES1+4,33:RETURN
979 REM ********************************
980 REM ROUTINE L.M. MEMORIA CARATTERI
```

(continua a pag. 144)

e pesca i caratteri da stampare dalla zona-sprite II (704-766).

Il programma

Il programma che propongo è un gioco di inseguimento tra un ometto e un robot cattivo, ove, ovviamente, il "chaser", cioè l'inseguitore, è il robot cattivo, e voi, la preda, l'ometto. Non utilizza peculiarità grafiche del 64 se non i caratteri ridefiniti dall'utente. Per essi, ho leggermente variato una routine in linguaggio macchina (un solo dato alterato) comparsa su MCmicrocomputer nº 30. L'ho fatto per ovvie ragioni di velocità, poiché PEEK e POKE dal Basic bastavano. Ho ridefinito una decina di caratteri per il gioco (nel range da 0 a 64), e altri dieci per le scritte comprendenti il marchio di fabbrica (DARIOSOFT) e il nome del gioco (nel range da 65 a 128). La grafica è piuttosto accurata, i suoni un po' meno a causa della mia poca dimestichezza con gli apparati sonori del 64. La velocità non è da arcade, siccome un gioco del genere non la richiedeva, essendo una questione di scelta logica e NON di riflessi. Vediamo com'è il gioco. In un campo delimitato da quattro muri (ai quattro lati dello schermo), c'è un certo numero di mine, all'inizio venti, poi ad ogni quadro se ne aggiungono venti, un robot e ún ometto. Il robot parte in alto a sinistra, mentre voi all'inizio siete al centro dello schermo; il robot comincia a venire verso di voi, con velocità RND, cioè vi segue. Il vostro compito è quello di fargli calpestare le mine (10 punti l'una), tutte le mine per passare al prossimo quadro. Voi però non dovete toccarle, poiché perdereste un uomo dei tre iniziali che avete (ma a 500, 1000, 1500, 2000, 2500, 3000 punti ne ricevete un altro; attenzione però, devono essere 500, 1000, etc., non prendete un ometto ogni 500 punti). Questo per cinque quadri, finito uno dei quali le mine aumentano, rendendo più difficile il passaggio; alla fine del quinto quadro, avrete vinto il gioco. Alla fine di ogni quadro, avete 100 punti bonus. Ma non è finita qui, altrimenti il gioco sarebbe monotono. Intanto, le mine sono sempre in posizione RND. Poi, cosa più importante, ogni tanto (sempre RND), il robot lascia dietro di sé degli oggetti, alcuni dei quali sono molto pericolosi. Questi possono essere:

Oggetto	Effetto su di voi	Effetto sul robot
ciliegia	perdita un uomo	+50 punti
(mina)	(perdita un uomo)	(+10 punti)
vampiro	perdita di tutto il gioco	+250 punti
chiave	apertura porte nei muri	+100 punti

Quando riuscite a toccare una chiave, vi si aprono 4 porte, una per muro, toccando una delle quali potrete passare al prossimo quadro, perdendo 100 punti. Il vampiro è molto pericoloso (io lo odio), ma due vampiri di seguito valgono un altro ometto. La

(segue da pag. 143)

```
981 REM ********************************
982 DATA169,0,133,252,133,254,169,208,133,255,169,48,133,253,160,0,177
983 DATA254,145,252,200,208,249,230,255,230,253,165,255,201,215,208,239,96
997 REM ********************************
998 REM DATI CARATTERI
999 REM ********************************
1000 DATA126,90,255,189,189,36,66,36,24,24,126,90,90,36,66,195
1005 DATA60,102,219,165,165,219,102,60,24,24,252,26,26,56,46,98
1010 DATA24,24,63,88,88,28,116,70,145,4,66,24,66,0,161,68
1011 DATA195,162,162,187,138,195,195,251,239,40,40,239,42,41,44,44,39,164,164
1015 DATA164,36,54,182,183,192,79,72,79,65,65,79,192,0,123,74,75,74,74,122,0
1020 DATA0,222,12,140,12,12,12,0
1025 DATA250,138,130,131,130,195,203,251,47,40,40,239,40,44,44,44,190,160,160
1030 DATA190,130,134,134,190,248,128,128,224,128,192,192,248
1035 DATA3,4,24,100,246,255,111,6,0,0,0,96,159,145,96,0
1040 DATA24,36,66,36,36,66,66,126,60,90,255,165,165,36,0,0
1045 DATA252,252,252,0,63,63,63,0
1051 REM ********************************
1052 REM DATI ROUTINE CURSORE L.M.
1053 REM ********************************
1055 DATA172,253,207,174,254,207,24,32,240,255,173,255,207,32,210,255,96
1060 REM ********************************
1061 REM DATI ROUTINE "REVERSE" L.M.
1062 REM ********************************
1065 DATA169,0,133,252,133,254,169,4,133,255,133,253,234,234,160,0,177,254,73
1070 DATA128,145,252,200,208,247,230,255,230,253,165,255,201,7,208,237,162,0
1075 DATA189,0,7,73,128,157,0,7,232,224,232,208,243,96
1080 REM ********************************
1081 REM ROUTINE "SOSTITUZIONE" L.M.
1082 REM ********************************
1085 DATA169,0,133,254,169,4,133,255,160,0,177,254,201,37,208,4,169,40,145,254
1090 DATA200,208,243,230,255,165,255,201,7,208,235,162,0,189,0,7,201,37,208,5
1095 DATA169,40,157,0,7,232,224,232,208,239,96
1097 REM ********************************
1098 REM CARATTERI PER ROUTINE "HIDER"
1099 REM ********************************
1100 DATA91,67,93,32,68,65,82,73,79,83,79,70,84,32,45,32,80,82,79,71,82,65,77
1105 DATA77,65,32,80,82,79,84,69,84,84,79
1197 REM ********************************
1198 REM DATI ROUTINE "HIDER" L.M.
1199 REM ********************************
1200 DATA169,192,133,251,169,2,133,252,169,13,32,210,255,160,0,177,251,32,210
1205 DATA255,200,192,34,208,246,32,49,168,96
2000 POKES+4,33:POKES1+4,17:SYS51000:PRINT"█N   "
2005 FORI=190TO230:POKES+1,I:POKES1+1,255-I:NEXT:SYS50000
2010 FORI=90TO70STEP-.8:POKES+1,I:POKES1+1,255-I:NEXT:SYS50000
2015 FORI=80TO130:POKES+1,I:POKES1+1,255-I:NEXT:SYS50000
2016 FORC=1TO2:FORI=0TO50:POKES+1,5:NEXT:POKES+1,0:FORM=1TO150:NEXT:NEXT
2020 POKES+1,0:POKES1+1,0:POKES+4,0:POKES1+4,0:FORI=0TO99:NEXT:SYS50000
2021 FORI=1TO500:NEXT
2022 POKE211,0:POKE214,24:SYS58732:FORI=0TO24:PRINT:FORN=1TO50:NEXT:NEXT
2025 FORI=0TO999:NEXT:PRINT"█   █ ▶\ 64, BY █|‾‾_█"
2030 PRINT"█YOUR SCORE IS:"SC:PRINT"█HIGH SCORE IS:"HI
2031 PRINT"█◣ANOTHER GO [█Y█/█N█]: ";:POKE204,0:POKE207,0:POKE198,0
2035 GETA$:IFA$=""THEN2035
2040 PRINTA$;:POKE204,1:POKE204,0:IFA$="Y"THEN2065
2045 IFA$<>"N"THEN2025
2050 PRINT"█BYE BYE █FROM █|‾‾_":POKE204,1:POKE207,0:END
2065 POKE204,1:RUN62
3000 AS=AS-1:PRINT"█"TAB(36)"█"AS"◣"
3005 IFAS=0THEN2000
3015 POKE53245,AS:POKE53246,1:POKE53247,32:SYS49152:POKE53245,X:POKE53246,Y
3016 POKE53247,40:SYS49152:FORI=1TO1500:NEXT
3017 POKE53245,X:POKE53246,Y:POKE53247,32:SYS49152:POKE53245,A:POKE53246,B
3020 SYS49152:A=1:B=3:X=20:Y=12
3025 POKE53245,X:POKE53246,Y:POKE53247,36
3030 SYS49152:POKE53245,A:POKE53246,B:POKE53247,35:SYS49152:GOTO200
4000 IF(INT(RND(0)*4))=1THEN4004
4001 RETURN
4004 SYS50000:SN=255:WA=129:GOSUB900:SYS50000:PRINT"◣"
4005 POKE53245,X:POKE53246,Y:POKE53247,32:SYS49152:PRINT"█"
4010 X=INT(RND(1)*38)+1:Y=INT(RND(1)*21)+3
4015 IFPEEK(C+X+40*Y)<>32THEN4010
4020 POKE53245,X:POKE53246,Y:POKE53247,36:SYS49152:RETURN
5000 POKE198,0:POKE53281,14:POKE53280,1:PRINT"█◣"
5005 PRINT"█|      ▶| ▶▶▶| ▶|      ▶|      ▶|     "
5010 PRINT"█| ▶▶▶| ▶| ▶▶▶| ▶| ▶▶▶| ▶| ▶▶▶▶▶| "
5015 PRINT"█| ▶▶▶▶▶|      ▶|      ▶|      ▶|   ▶▶█64◣"
5020 PRINT"█| ▶▶▶| ▶| ▶▶▶| ▶| ▶▶▶| ▶▶▶▶▶| ▶| "
5025 PRINT"█|      ▶| ▶▶▶| ▶| ▶▶▶| ▶|      ▶|     "
5030 PRINT"██████████████████████████████|█[C] 1984 ↑|‾‾_"
5035 PRINT"███████████ #█ : ██KILLER ROBOT██"," $ : █ ▶\█←TU█","█ % : ██MINA█"
5036 PRINT"██CONTROL PORT N°2 [JOYSTICK],   █OPPURE█"
5040 PRINT"█ 3:█SU","█X:█GIU"","█@:█SIN.","█↑:█DEST."
5055 PRINTTAB(8)"█████PRESS█ ANY █KEY█ TO█ BEGIN"
5070 POKE12575,129:FORT=1TO75:NEXT:POKE12575,36
5075 GETA$:IFA$=""THENFORT=1TO75:NEXT:GOTO5070
5080 FORT=1TO10:FORM=1TO5:NEXT:POKE53281,0:FORI=1TO5:NEXT:POKE53281,14:NEXT
5085 POKE198,0:PRINT"██"
5100 PRINTTAB(14)"██SCORE  TABLE██"
5105 PRINT"██%.......10 PNT":PRINT"██%.......-1 UOMO [$]"
5110 PRINT"██!.......50 PNT":PRINT"██!.......-1 UOMO [$]"
5115 PRINT"██<.......100 PNT":PRINT"██<.......APERTURA PORTE"
5120 PRINT"██=.......-100 PNT - PROSSIMO QUADRO"
5125 PRINT"██>.......250 PNT":PRINT"██>.......FINE GIOCO"
5130 PRINT"███PRESS ANY KEY TO BEGIN PLAYING"
5135 GETA$:IFA$=""THEN5135
5140 RETURN
6000 POKES+4,33:POKES1+4,17
6005 FORI=255TO0STEP-1:POKES+1,I:POKES1+1,255-I:NEXT
6010 POKES1+1,0:POKES+1,0:POKES1+4,33:POKES+4,0:RETURN
7000 FORI=0TO15:SN=INT(256*RND(0)):WA=33:GOSUB900:NEXT:SYS50000:FORI=0TO99:NEXT
7005 PRINT"████YOU WON THE GAME":GOTO2025
8000 IFSC=5000ORSC=10000ORSC=15000ORSC=20000ORSC=25000ORSC=3000THENAS=AS+1:CL=-1
8005 IFCLTHENCL=0:PRINT"█"TAB(36)"█"AS:GOSUB6000:CH=36:GOSUB9000
8010 RETURN
9000 CX=AS-1:POKE53245,CX:POKE53246,1
9005 POKE53247,CH:SYS49152:RETURN
10000 FORT=0TO9:SN=INT(256*RND(0)):WA=17:GOSUB900:NEXT:SYS50000
10005 LV=LV+1:PRINT"██"TAB(36)"█"LV:SC=SC+100
10010 MI=MI+20:IFMI=120THEN7000
10015 POKE211,15:POKE214,12:SYS58732
10020 PRINT"█SCREEN END":FORI=1TO2000:NEXT:A=1:B=3:X=20:Y=12:CN=0:GOTO67
```

Note al listato

Il programma usa alcune routine in linguaggio macchina: sono spiegate nel prossimo paragrafo. Vediamo il nostro listato:

LINEE da-a	CONTENUTO
8-55	inizializzazioni caratteri grafici e routine L.M.
62-199	inizializzazione schermo, variabili, presentazione
200-425	loop principale (controllo uomo e robot)
500-599	routine score ciliegia (usata anche per chiave, vampiro)
700-799	routine apertura porte
800-830	routine score mina
900-910	routine suono generico
950-970	routine suono esplosione
979-1999	dati per caratteri e routine in L.M.
2000-2999	fine gioco
3000-3999	perdita di un ometto
4000-4999	routine spostamento nuova posizione RND
5000-5999	presentazione gioco
6000-6999	routine musichetta per il bonus di un ometto
7000-7999	gioco terminato e vinto
8000-8999	routine controllo se deve esserci bonus
9000-9999	routine scrittura ometti da bonus
10000-10020	fine schermo

Commenti al listato

Numero di spazi da battere nelle linee di presentazione per ottenere correttamente la scritta gigante: Chase 64

LINEA	NUMERO SPAZI
5005	5,1,1,5,5,5
5010	1,1,1,1,1,1,1,1
5015	1,5,5,5,3
5020	1,1,1,1,1,1,1,1
5025	5,1,1,1,1,5,5

Caratteri grafici da digitare per ottenere i caratteri ridefiniti

LINEA	CARATTERI DA DIGITARE
5050	shift GHIJ
67	shift ABCDEF
71	shift GHIJ
5030	shift ABCDEF
2050	shift ABCDEF
2025	shift GHIJ, shift ABCDEF

Elenco variabili principali usate nel programma Chase 64

VARIABILI	USO
C	inizio memoria schermo
SC	punteggio
HI	high score
AS	ometti rimasti
S	inizio SID/differenza memoria schermo-colore
CN	numero mine colpite
MI	numero mine
X	ascissa ometto
Y	ordinata ometto
A	ascissa robot
B	ordinata robot
LV	livello raggiunto

Questo programma è disponibile su cassetta presso la redazione. Vedere l'elenco dei programmi disponibili e le istruzioni per l'acquisto a pag. 129.

ciliegia (in realtà nel carattere ce ne sono 2) è molto simpatica, ma vi fa perdere un uomo come la mina. Durante il gioco è costantemente indicato il punteggio, l'high score, gli ometti rimasti e il livello raggiunto.

Voi vi muovete con il joystick in porta 2; è però consentito anche il controllo da tastiera, e i tasti adatti li trovate nel listato. Il joystick è senz'altro più comodo, e vi consente anche di muovervi in diagonale, mentre con la tastiera siete limitati alle direzioni standard NSEO. Come detto, la velocità del gioco non è affatto eccezionale, ma è perfetta per un gioco del genere, nel quale dovete avere il tempo di pensare, di valutare la direzione. State attenti, perchè il robot è parecchio più veloce in diagonale, quindi non fategli prendere (con il termine "prendere" intendo toccare, in qualsiasi direzione) troppe mine in diagonale, o prima o poi sarà lui a prendervi. I colori del gioco sono simpatici, tuttavia è facile giocare anche su una TV in BN.

Il programma prevede un altro controllo dell'ometto, non esplicato nel paragrafo precedente. Premendo il tasto FIRE o la barra spaziatrice, con il 25% delle probabilità (RND), il vostro ometto si troverà spostato in un punto sicuro dello schermo, chiaramente non occupato da mine o simili. È piuttosto utile, ma appunto il suo uso non è sempre consentito.

```
*****************************************************************
ROUTINE IN LINGUAGGIO MACCHINA PER LA PROTEZIONE DA SAVE & LIST
*****************************************************************
CD14  A9 C0     LDA #$C0 : carica il low byte della zona-caratteri
CD16  85 FB     STA $FB  : lo memorizza
CD18  A9 02     LDA #$02 : carica l'high byte [$02C0=704]
CD1A  85 FC     STA $FC  : lo memorizza
CD1C  A9 0D     LDA #$CD : carica un RETURN [CHR$(13)]
CD1E  20 D2 FF  JSR $FFD2: lo stampa
CD21  A0 00     LDY #$00 : azzera il contatore
CD23  B1 FB     LDA ($FB),Y: legge un carattere (inizio loop)
CD25  20 D2 FF  JSR $FFD2: lo stampa
CD28  C8        INY      : incrementa il contatore
CD29  C0 22     CPY #$22 : guarda se ha finito la frase
CD2B  D0 F6     BNE $CD23: se non l'ha finita, torna al loop
CD2D  20 31 A8  JSR $A831: esegue la routine del BASIC "END"
CD30  60        RTS      : torna al BASIC.

Start location: $CD14 (52500)
End   location: $CD30 (52528)
Bytes occupied: 29
```
LM1

```
*****************************************************************
ROUTINE IN LINGUAGGIO MACCHINA PER IL POSIZIONAMENTO DEL CURSORE
*****************************************************************
C000  AC FD CF  LDY $CFFD: carica in Y le ascisse (1-38)
C003  AE FE CF  LDX $CFFE: carica in X le ordinate (3-23)
C006  18        CLC      : pulisce il flag CARRY
C007  20 F0 FF  JSR $FFF0: posiziona il cursore (routine "PLOT")
C00A  AD FF CF  LDA $CFFF: legge il carattere da stampare
C00D  20 D2 FF  JSR $FFD2: stampa il carattere (routine "CHROUT")
C010  60        RTS      : torna al BASIC.

Start location: $C000 (49152)
End   location: $C010 (49168)
Bytes occupied: 17
```
LM2

```
*****************************************************************
ROUTINE IN LINGUAGGIO MACCHINA PER IL REVERSE DELLO SCHERMO
*****************************************************************
C350  A9 00     LDA #$00 : carica il low byte
C352  85 FC     STA $FC  : lo memorizza
C354  85 FE     STA $FE  : "      "
C356  A9 04     LDA #$04 : carica l'high byte
C358  85 FF     STA $FF  : lo memorizza
C35A  85 FD     STA $FD  : "      "
C35C  EA        NOP      : "no operations"
C35D  EA        NOP      : "no operations"
C35E  A0 00     LDY #$00 : azzera il contatore
C360  B1 FE     LDA ($FE),Y: legge un carattere da schermo (loop)
C362  49 80     EOR #$80 : ne trova il reverse
C364  91 FC     STA ($FC),Y: lo visualizza (rimpiazza vecchio)
C366  C8        INY      : incrementa il contatore
C367  D0 F7     BNE $C360: torna al loop se Y<> top-value
C369  E6 FF     INC $FF  : altrimenti incrementa high byte
C36B  E6 FD     INC $FD  :      "       "    "
C36D  A5 FF     LDA $FF  : legge l'high byte
C36F  C9 07     CMP #$07 : guarda se deve passare al secondo loop
C371  D0 ED     BNE $C360: se no, torna al loop
C373  A2 00     LDX #$00 : altrimenti azzera il contatore (X)
C375  BD 00 07  LDA $0700,X: legge carattere da schermo (2° loop)
C378  49 80     EOR #$80 : ne trova il reverse
C37A  9D 00 07  STA $0700,X: lo visualizza
C37D  E8        INX      : incrementa il contatore
C37E  E0 E8     CPX #$E8 : guarda se ha raggiunto la fine schermo
C380  D0 F3     BNE $C375: se no, torna al 2° loop
C382  60        RTS      : altrimenti torna al BASIC.

Start location: $C350 (50000)
End   location: $C382 (50050)
Bytes occupied: 51
```
LM3

Le routine in linguaggio macchina

Il programma macchina usa cinque routine in linguaggio macchina, delle quali nessuna usa le peculiarità.

Lm 1: è la routine che sostituisce l'esecuzione dei comandi SAVE e LIST. Stampa dei caratteri di una frase presentandoli da una zona di memoria riservata.

Lm 2: la routine per mettere in reverse lo schermo è la più bella dopo quella che fa esplodere tutte le mine insieme (non proprio, a circa 10 nanosecondi di distanza una dall'altra). Essa mette in reverse lo schermo in tutti e due i casi possibili per lo stato dello schermo (reverse o no). Dà un effetto simile ad un sobbalzo.

Lm 3: muove il cursore nella posizione desiderata.

La protezione

Vorrei concludere dicendo due parole sulla protezione del programma. La linea 15 deve essere battuta SOLO E SOLAMENTE DOPO aver controllato che il programma giri perfettamente. Fate molta attenzione a ricopiare i DATA riguardanti la routine per lo spostamento dei caratteri, perché essa disabilita la tastiera e un errore di copiatura per essa porterebbe al blocco del sistema (leggi addio programma). Quindi, salvate il programma prima di dare il fatidico RUN.

Il motivo per cui la linea 15 deve essere battuta dopo aver controllato il corretto funzionamento del programma è perché essa disabilita i tasti RUN / STOP&RESTORE, il comando LIST e il SAVE. Sostituisce i vettori di puntamento per i comandi con la zona dove è memorizzata la routine LM 1. Leggendo i valori delle locazioni "incriminate" con un PEEK prima del RUN saprete quali sono i valori da ripristinare per il funzionamento normale. Nel paragrafo "commenti al listato" è presente un aiuto a battere i caratteri grafici di alcune linee, gli spazi e un elenco delle principali variabili, per una migliore comprensione del programma. Buon divertimento! MC