Programmiamo Videogiochi

di Marco Pesce

prima parte

Eh! eh! eh! Credevate di esservi liberati di me vero? E invece, come si era intuito nella penultima puntata dedicata al Commodore sessantaquattro, siamo passati alla «signorina» Amiga. Non preoccupatevi perché non ho intenzione di realizzare un nuovo «megagame», quindi il nostro sarà un discorso fine a se stesso, anche se non mancheranno esempi pratici di corredo alle «lezioni» teoriche

Per chi ancora non avesse capito, questo è il primo di una serie di articoli su come realizzare un videogioco sull'Amiga, dando per scontato che siete già in possesso di un minimo di cultura di base sull'argomento, magari maturata in seguito alle puntate dedicate al Commodore 64. Il linguaggio di programmazione sarà l'Assembler (del 68000 ovviamente) e se non lo masticate non è il caso di preoccuparsi perché rientra negli obiettivi quello di trattare l'argomento in una apposita rubrica di prossima istituzione. Cominciamo al volo con le argomentazioni «visive».

La grafica

La prima cosa che non farò è quella di elencarvi i modi grafici di Amiga, perché li sa anche mio fratello (... scherzavo Paolo). La prima che farò invece è parlarvi di ciò che possiamo fare riguardo i videogiochi. Lo spasso di Amiga in questo campo è senz'altro lo scrolling fluido, che la esalta rispetto alla concorrenza; i vantaggi che riporta in questo settore sono soprattutto meriti del Copper e del metodo di gestione dei bitplane. Vediamo di spiegare brevemente. Un bitplane è l'insieme dei byte che costituiscono la pagina grafica. Per una pagina grafica multicolore occorrono più bitplane, una pagina di 320×200 pixel in 16 colori necessita di 4 bitplane da 8 Kbyte ciascuno (40 byte in orizzontale per 200 in verticale). I primi 8 pixel, dell'angolo video in alto a sinistra, sono realizzati con il primo byte (prendiamo in esame il caso ad un solo bitplane). Il primo pixel è rappresentato dal bit più significativo. Il successivo byte rappresenta i successivi 8 pixel, continuando in orizzontale verso destra, e così via fino alla fine della prima riga di 320 pixel e continuando quindi con la seconda. Fin qui sembrerebbe che questo sia solo un sistema un po' diverso da quello usato nel 64, ma c'è un'altra particolarità; il primo byte della serie può essere scelto indipendentemente tra i 512 Kbyte disponibili per i custom chip. È

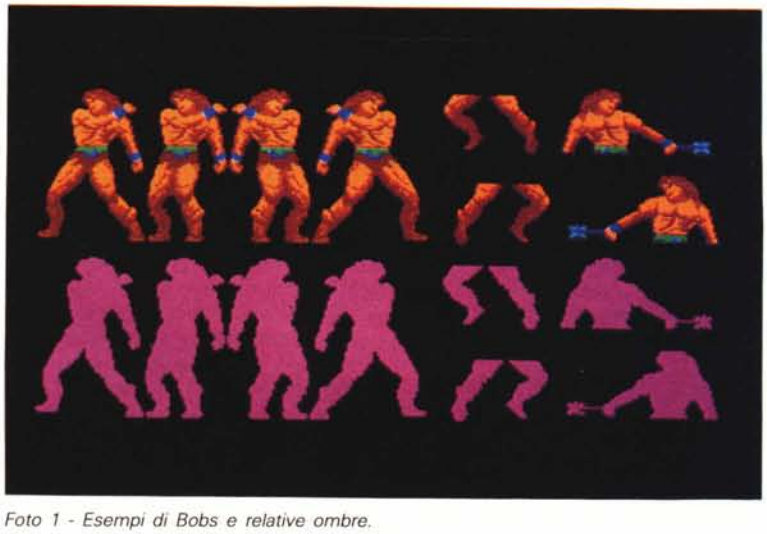


Foto 1 - Esempi di Bobs e relative ombre.

già un passo avanti, ma anche così sembrerebbe che il tutto abbia poco a che fare con lo scrolling fluido. Ed ecco l'ultima chicca; esiste un registro che permette di regolare il «pitch» fine del bitplane visualizzato. Adesso i più bravi avranno capito che per uno scrolling, ad esempio, verso sinistra è sufficiente decrementare il pitch del bitplane fin quando si azzera e una volta giunto a tal valore incrementare la posizione di start della prima parola (i primi due byte) e ristabilire il massimo valore nel pitch control per poi continuare all'infinito. L'unico accorgimento sarà quello di aggiornare in modo opportuno la grafica contenuta in ram, ma di questo parleremo in seguito. Per lo scrolling in verticale il discorso è ancora più semplice; basta incrementare (per uno scrolling verso l'alto) il puntatore della prima parola di 20 word (40 byte). Ed ora vediamo cosa c'entra in tutto questo il Copper. L'Amiga possiede 6 registri (per un totale di 24 byte, 4 byte a registro quindi) dedicati ai puntatori dei bitplane, che come abbiamo detto possono contenere un valore qualsiasi. La cosa buffa è che, ad ogni ciclo di scansioni del pennello elettronico, i puntatori devono essere riposizionati al loro valore iniziale (stabilito dall'utente), perché durante la visualizzazione della pagina grafica vengono alterati, o meglio incrementati fino all'ultima word dello screen. Un compito così ingrato poteva essere realizzato tramite un sistema di interruzioni collegato al pennello elettronico, ma è proprio qui che interviene il Copper, che grazie alla sua bravura in questo campo (l'azione in funzione della posizione del pennello elettronico), ci permette di ignorare questo problema o quasi. È evidente che le sue funzioni non si limitano a questo. Esso può essere considerato un piccolo coprocessore. Possiede solo tre istruzioni, Wait, Move e Skip, tuttavia queste ci permettono di programmare molti eventi, tutti collegati

Foto 2 - Il cattivone.

Foto 3 e 4 - I risultati.

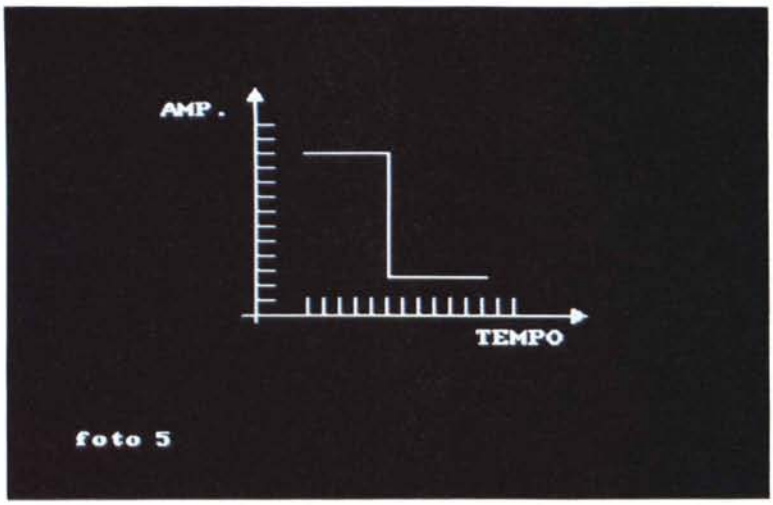


foto 5

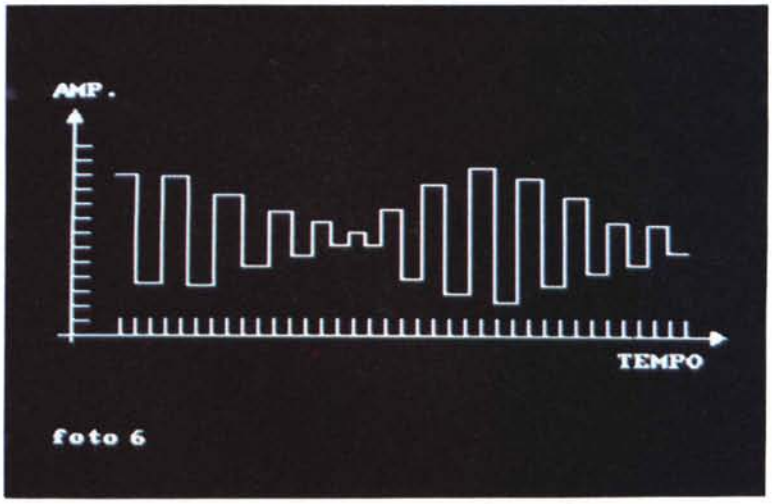


foto 6

alla posizione del raster. Brevemente, l'istruzione Wait mette il Copper in stato di attesa fin quando non viene raggiunta la posizione indicata dall'istruzione stessa, l'istruzione Move si usa dopo la Wait per intraprendere un'azione di trasferimento di valori nei registri di controllo dell'hardware e l'istruzione Skip serve per «saltare» istruzioni. In sostanza quindi possiamo fare in modo che ad una particolare posizione del raster vengano alterati i colori dello schermo, il modo grafico di visualizzazione delle prossime linee o qualunque altro (o quasi) avvenimento riguardo i registri dei chip di I/O. I «programmi» del Copper sono scritti con un sistema particolare che vedremo in seguito.

Vediamo un'altra caratteristica particolare di Amiga: il Blitter. Sicuramente ne avrete per lo meno sentito parlare; si tratta del coprocessore che manipola le aree ram con grande dimestichezza, in termini di velocità. Alcune particolari funzioni lo rendono particolarmente indicato alla produzione e gestione dei famosi «Bobs» ovvero sprite software, che necessitano anche delle «ombre» (vedi foto 1, 2, 3 e 4). Le sue capacità non consentono però di ottenere immediatamente i risultati desiderati, ma con un'opportuna gestione software si aggira ampiamente il problema; nelle prossime puntate vedremo come realizzare praticamente (in LM) questi Bobs, senza l'ausilio del (lento) sistema operativo. In alcune occasioni, tuttavia, preferisco servirmi di quest'ultimo perché risulta più pratico, e vedremo anche quali sono. Dopo questa brevissima occhiata alla grafica di Amiga passiamo ad occuparci un po' del sonoro.

L'audio

Un po' tutti sappiamo dell'esistenza delle 4 voci stereo (tranne mio fratello). Quello di cui ci occuperemo in questa parte della rubrica sarà l'utilizzazione diretta del chip che li gestisce, con annessi trucchetti. Le 4 voci sono costituite da convertitori digitale-analogico. Il funzionamento è simile a quello che troviamo nell'elettronica di un CD player, ad eccezione del fatto che in quest'ultimo i convertitori sono a 16 bit e sono solo due (uno per canale), mentre nell'Amiga sono ad 8 bit, ma in compenso ce ne sono 4 (due per canale). Tali convertitori possiedono (anche) un sistema di accesso diretto alla memoria (DMA), che gli permette di operare senza richiedere l'intervento del microprocessore, almeno per quanto riguarda il prelievo dei dati da convertire. Per generare una nota con questo sistema (il DMA) occorre memorizzare una serie di dati in ram relativi all'evoluzione dell'ampiezza del segnale nel tempo, per ottenere qualunque tipo di forma d'onda. Possiamo scegliere di memorizzare anche l'inviluppo (attack, decay, ecc.) e quindi prepararci ad utilizzare una porzione di ram maggiore, oppure agire direttamente sul volume (specifico) della voce. Vediamo di spiegare con un esempio. Ammettiamo di voler generare una forma d'onda quadra. I dati della ram possono contenere sia una breve tabella che indichi solo una ventina di valori (foto 5), suddivisi in stato alto e stato basso come più ci piace, sia una tabella molto più lunga che comprenda anche l'evoluzione del volume (vedi foto 6). Più ram consumiamo e meno affaticamento subirà il microprocessore. I dati possono anche riferirsi ad un segnale campionato tramite opportuna apparecchiatura. In ognuno di questi casi occorrerà tuttavia specificare la frequenza della nota tramite il sampling rate. E se proprio vogliamo liberarci di tutti i nostri doveri da musicista allora possiamo campionare l'intera canzone e mandarla in loop con opportuna frequenza fissa. L'ultimo metodo è il più dispendioso ovviamente e anche il più monotono in termini di varietà delle sonorità, in quanto il brano campionato non può durare molto a lungo, a meno di «taglia e incolla» musicali. Esiste un'ultima modalità, che occupa a tempo pieno il microprocessore, il Non-DMA. In tale sistema dovremo fornire i singoli campioni al chip sonoro agendo nei suoi registri, con evidente impegno del microprocessore. Anche se sembra il sistema peggiore esso permette di raggiungere nuovi risultati in termini di versatilità di sintesi sonora, in quanto possiamo simulare oscillatori software e inviare il risultato al chip sonoro, magari aumentando la polifonia dell'Amiga; più adatto per un sintetizzatore che per un videogioco. Concludiamo qui con la promessa di entrare più in dettaglio nei prossimi numeri della rubrica. MC

COMPUTER HSP COMPUTER

COMPUTER HSP COMPUTER

COMPUTER HSP COMPUTER

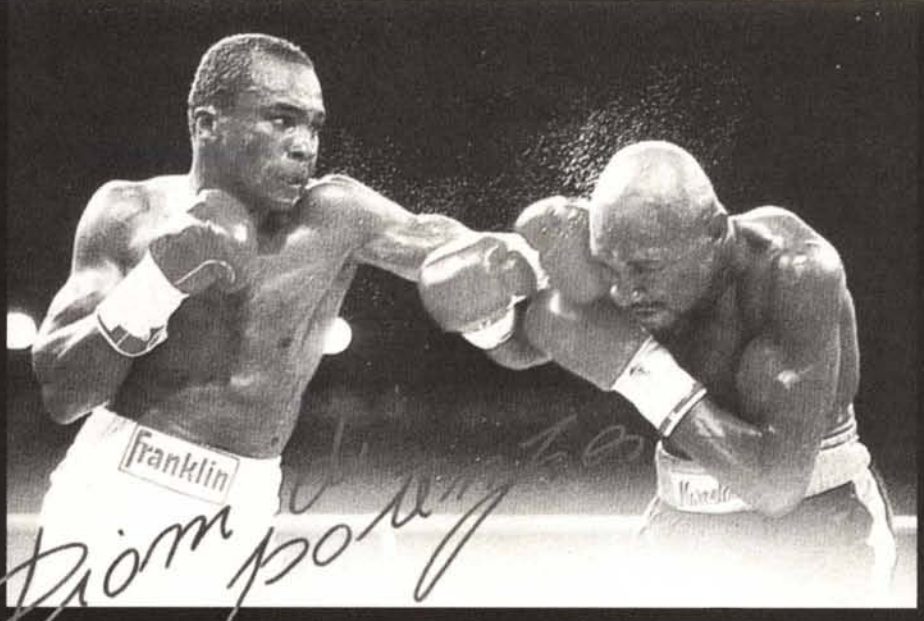
COMPUTER HSP COMPUTER

COMPUTER HSP COMPUTER

COMPUTER HSP COMPUTER

COMPUTER HSP COMPUTER

THE BIG APPLE

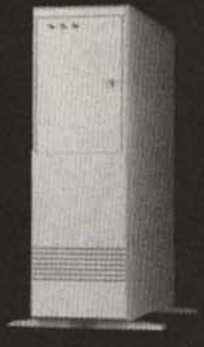


il campionato

386 33MHz
64 K CACHE

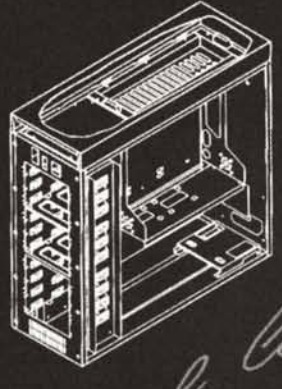
ANNO 1989

486 25 MHz



ANNO 1990

.....



ANNO 1991

e la storia continua

Uff. Comm. Via P. Fumaroli 12/A - Tel. 06/2251517 - 0337/726451 - ROMA
Conc. Centro Italia Info.Sist.: Via Malta 8 - Tel. 06/8842378-8411987 - 00198 Roma
Centro ass. PC Service Via Malta 8 - Tel. 06/8411987 - 00100 Roma
Cerchiamo concessionari per zone libere