System fly

Michelangelo Lupone
SIM, *Società Informatica Musicale, Roma*

Il Fly è stato progettato e realizzato con lo scopo di giungere ad un sistema-laboratorio che, dedicato alla composizione ed esecuzione musicale, permettesse anche di intervenire in modo sperimentale sulle strutture algoritmiche di sintesi e di controllo. Il progetto, tenuto conto delle specifiche di flessibilità indispensabili alla sperimentazione, ha individuato una tecnologia il cui costo risultasse opportuno alla fascia di utenza del *personal computer*. Proprio questa caratteristica ha richiesto il maggior impegno progettuale perché il contenimento economico del sistema ha implicato una precisa delimitazione delle specifiche d'uso ed una adeguazione puntuale, soprattutto nell'uso in tempo reale, della progettazione *software* alle caratteristiche *hardware*. Con questo intendo dire che per ottenere delle prestazioni adeguate il *software* è stato progettato in relazione diretta con la velocità di calcolo e la struttura del Fly, ciò ha implicato, ad esempio, valutazioni sulle precisioni numeriche utilizzate, in base anche allo spazio di memoria occupato e/o il tempo di calcolo.

Ciò che risulta a mio avviso importante è, però, la totale possibilità di riconfigurazione della struttura *software*, ed è ciò che maggiormente risulta utile ad una sperimentazione sia musicale che scientifica.

Il sistema per sua natura si propone come un flessibile laboratorio di *signal processing*, di cui l'utilizzazione musicale non è che un aspetto particolare; avvalendosi poi di un principio di modularità si propone a richiesta di prestazioni anche più gravose.

La figura 1 mostra i blocchi essenziali del sistema, cioè la configurazione scelta come base per un uso indirizzato alla ricerca, alla composizione ed alla esecuzione in tempo reale.

Al *computer* ospite (IBM o APPLE) è affidata la gestione dei dati in ingresso e la loro conservazione permanente, la gestione delle *routines* di controllo e sincronizzazione dei moduli Fly, la programmazione e scansione della tastiera.

La prima interfaccia realizzata per APPLE ha permesso la comunicazione con due moduli Fly, senza che venissero rilevati problemi

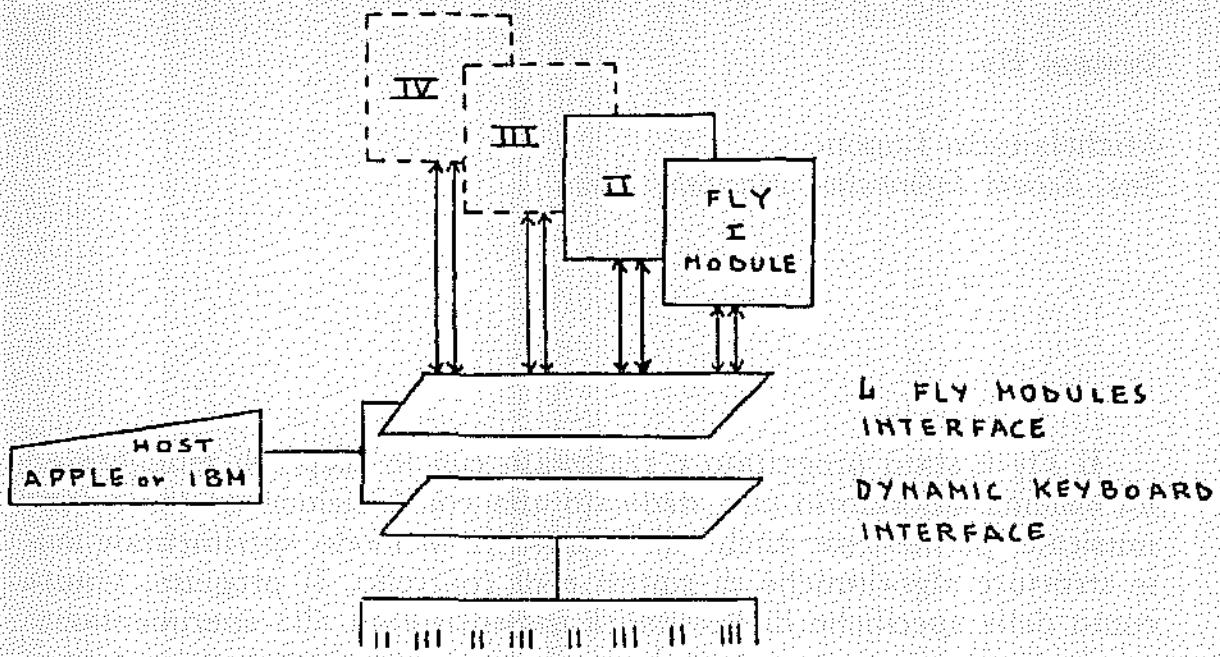

Figura 1

di insufficiente velocità da parte del *computer* ospite; va considerato però che un funzionamento a pieno regime, cioè scansione della tastiera ed assegnazione di venti eventi simultanei, registrazione degli eventi assegnati, controllo ed invio differenziato a ciascun modulo Fly, è al limite dei 10 ms., e per quanto in alcune applicazioni risulti ancora efficiente, non ottimizza delle prestazioni in tempo reale.

La possibilità di utilizzo di un Pc di più elevate prestazioni supera questa condizione ed emancipa a quattro moduli il controllo.

La figura 2 mostra i blocchi che compongono il modulo singolo. Sono due sistemi in parallelo, ciascuno formato da una *board* che adibita al processo dei segnali, un convertitore Dac a 12 bit in complemento a due, un filtro passa basso di tipo *butterworth* (quattro celle del 2° ordine) con due frequenze di taglio selezionabili (4.5 Khz – 9.3 Khz).

La *board* di processo è basata sul Dsp Tms 320 e la figura 3 ne mostra la macrostruttura. Si tratta di una *board*, costituita da 25 integrati, realizzata in formato singolo Eurocard facilmente adattabile agli standard *rack*. La presenza del Dsp ha reso possibile l'esigua componentistica ed un conseguente sfruttamento semplificato della logica di controllo.

L'architettura Harvard di questo processore e la struttura dei suoi elementi aritmetici sono alla base della flessibilità del Fly. La Alu è di tipo generale ed il moltiplicatore 16 × 16 *bit* è in grado di fornire il prodotto in 600 o 800 ns. a seconda delle istruzioni utilizzate, la manipolazione dei dati utilizza un *barrel shifter* nel trasferi-

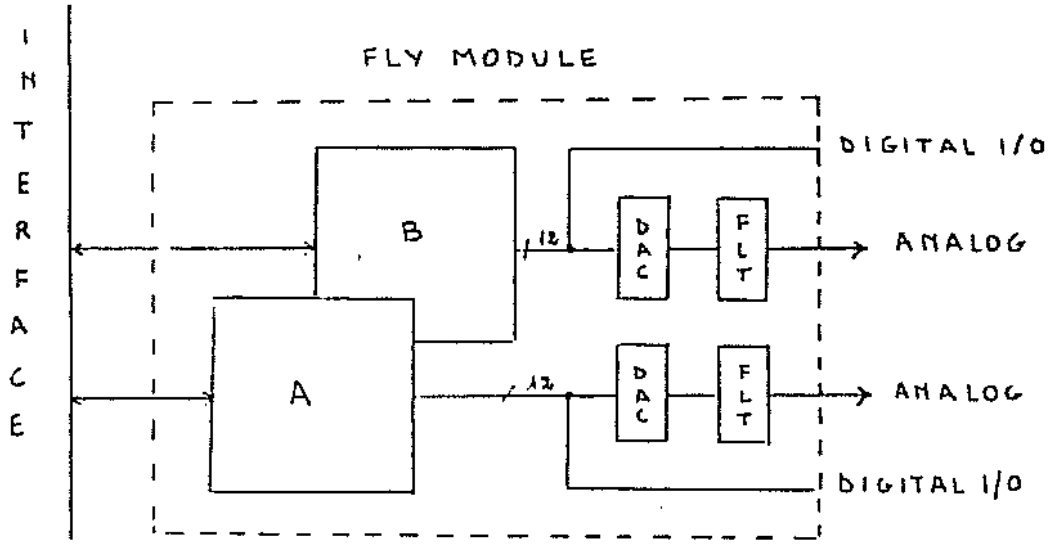

Figura 2

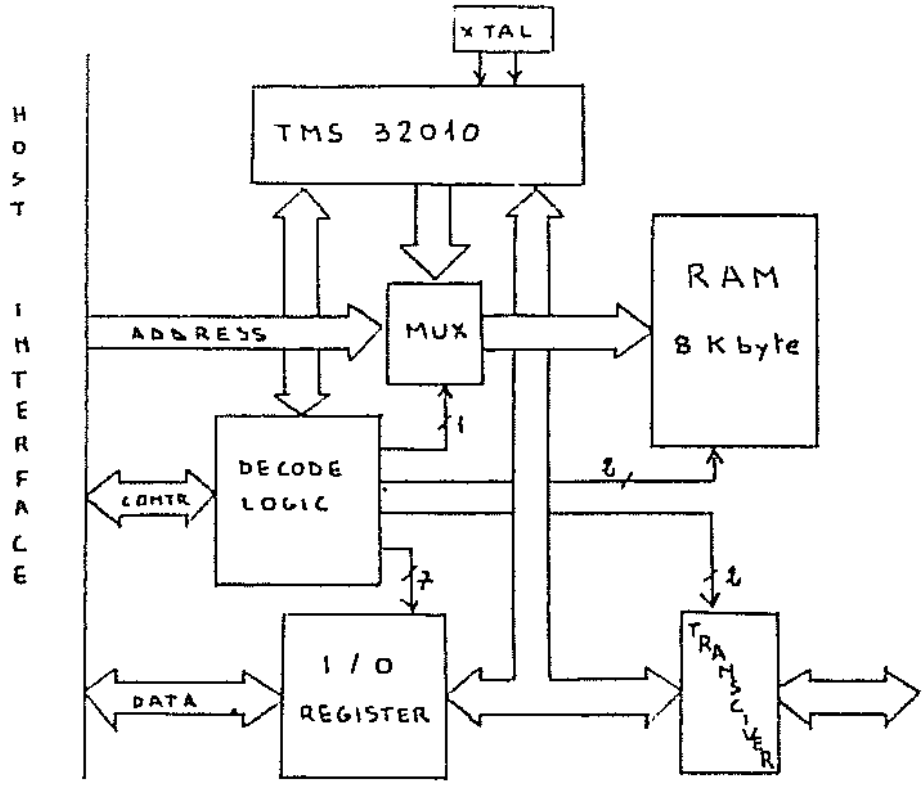

Figura 3

mento dalla Data RAM alla ALU (opera *left-shift* 0 ÷ 15) ed un *parallel shifter* (opera *left shift* Ø, 1, 4) allo scaricamento dell'accumulatore. Una RAM statica da 55 ns. è vista dal processore come memoria di programma ed il Fly ne consente l'utilizzazione di 3 *Kwords* per la lettura e scrittura ed 1 *Kword* per la sola lettura (la struttura *software* sfrutta queste mille locazioni per il *Monitor*, la *routine* di *Interrupt*, i dati di inizializzazione).

Quando una *board* è attiva, il *computer* ospite comunica con essa attraverso otto successive locazioni di memoria, ma può accedere direttamente alla RAM in scrittura e lettura, attraverso un *multiplexer* durante lo stato di *Halt* della *board*.

Il DSP utilizzato offre per molti aspetti, caratteristiche di microprogrammazione; questo rende estremamente agile la implementazione di algoritmi ricorsivi, presentando inoltre il vantaggio di un ciclo singolo (200 ns.) per la maggior parte delle istruzioni, sono semplificate le strutture con processi paralleli.

Da questo è derivato un progetto *software* sostanzialmente aperto, ma anche una decisa ottimizzazione del tempo di calcolo in *routines* gravose per il sistema ed una conseguente rappresentazione numerica. In particolare nelle *routines* per l'uso in tempo reale è stata usata un'aritmetica in virgola fissa, ed operazioni in singola precisione mantenendo 16 *bit* di risultato.

L'impostazione, ritenuta opportuna in questa prima fase, del progetto è stata di sperimentare le possibili condizioni d'uso del sistema, cioè quelle procedure su cui maturare non necessariamente un linguaggio, ma un ambiente dove la trasparenza del Fly risultasse sufficientemente elevata per potervi operare con risorse anche minime di programmazione, oppure offrire all'utilizzatore più esperto la possibilità di riconfigurare il sistema per le proprie esigenze espressive e di ricerca.

In questo senso sono state individuate due procedure fondamentali tuttora sottoposte ad analisi e solo parzialmente soddisfatte da risultati sperimentali:

I) PROCEDURA STATICA
L'operatore compone la struttura parametrica
a) assegna la struttura ai tasti
ac) esegue alla tastiera la disposizione temporale
b) assegna la struttura agli eventi di una *Score*
bb) la fa eseguire automaticamente.

II) PROCEDURA DINAMICA
L'operatore presenziona ed attiva le funzioni ed i controlli sulle funzioni.
L'operatore, selezionata la funzione di ogni *board*, (4 possibili nel siste-

ma con APPLE) potrà disporre di procedure di attivazione e/o controllo da tastiera distinte su ogni singolo tasto, differenziate dalla velocità di articolazione, per un insieme simultaneo di 20. Con funzioni si estende il concetto ai processi di sintesi (fino a 4 simultaneamente), calcolo di parametri degli algoritmi, controlli di livello superiore come densità, variazione, riferimento, velocità, controlli di device.

Il lavoro fin qui svolto ha soddisfatto solo la Procedura Statica che si è articolata in tre segmenti indipendenti relazionati a coppie. Due segmenti hanno trovato anche una immediata applicazione musicale che ne ha messo in evidenza le potenzialità e gli aspetti di ridondanza.

Con questa Procedura è stato realizzato *Presente Continuo* di Laura Bianchini, per clarinetto e nastro magnetico; *Lontano* di Francesco Galante; e *Mira*, un mio lavoro per *computer* in tempo reale.

La figura 4 mostra i segmenti della Procedura statica. I segmenti ultimati sono lo USER e il KEYBOARD.

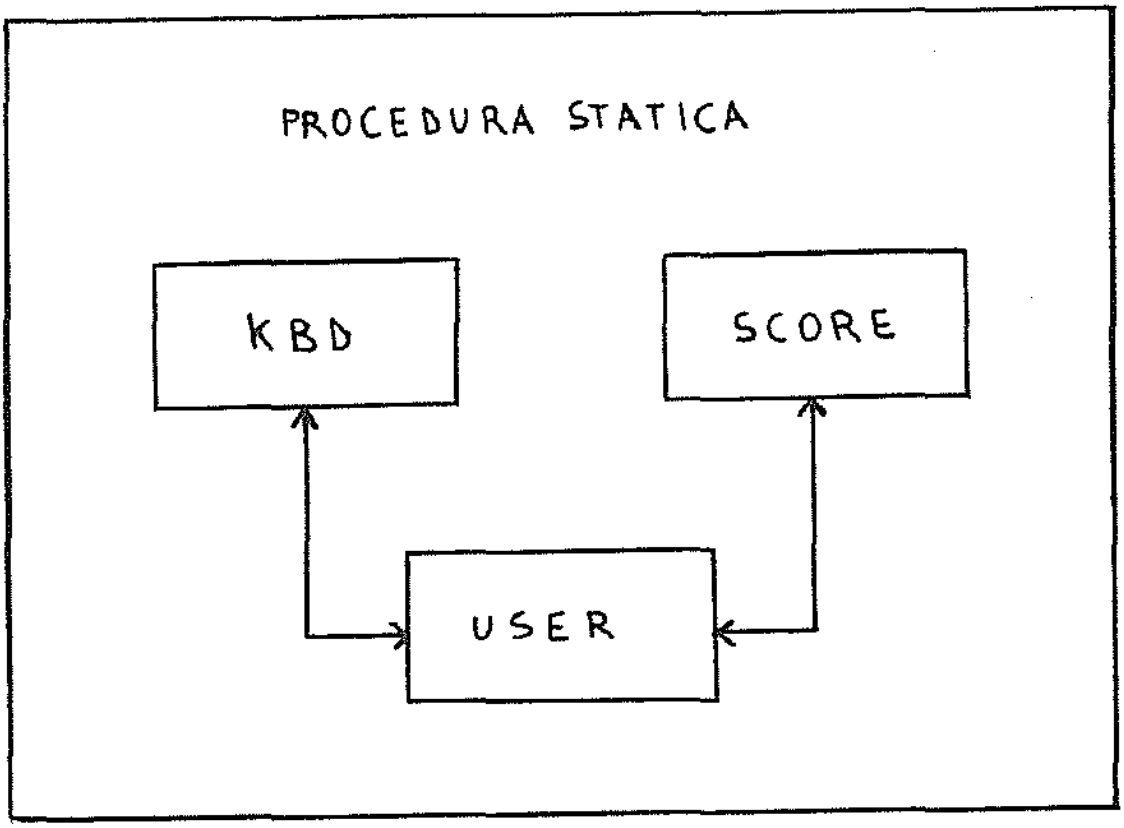

Figura 4

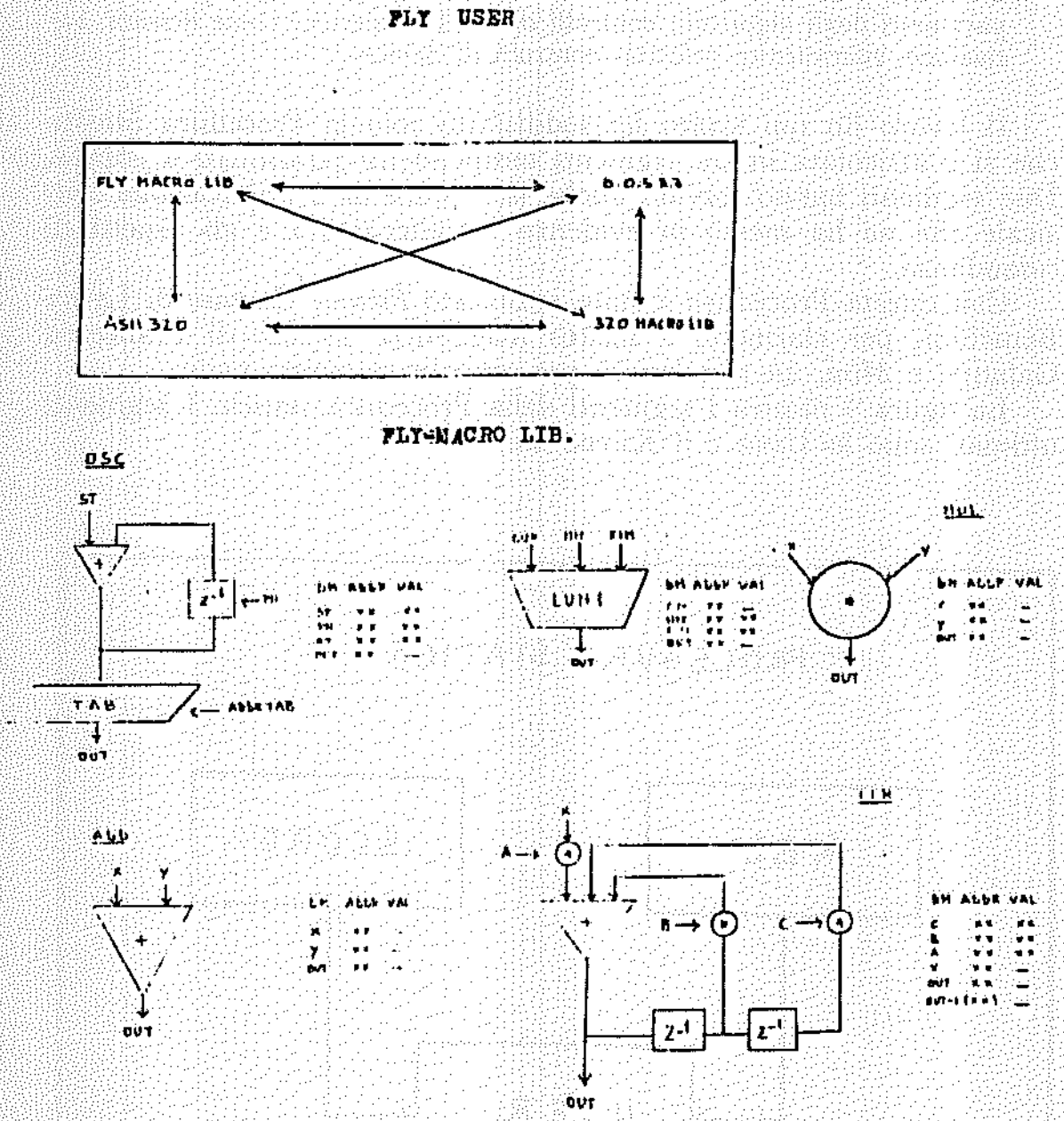
FLY USER
FLY MACRO LIB
ASM 320
320 MACRO LIB
FLY-MACRO LIB.
OSC
ST
TAB
OUT
LUT
OUT
OUT
ADD
OUT
OUT

Figura 5

Lo USER si pone a un livello più basso rispetto agli altri due; è sostanzialmente un ambiente vicino alla macchina e si relaziona all'utente in modo meno rigido ma più complesso da gestire (la figura 5 mostra la struttura macro Fly e le connessioni a diamante tra le parti).

Il segmento USER viene chiamato dai segmenti KBD e SCORE che ne gestiscono i parametri, l'uno in tempo reale, l'altro in tempo differito.

Il segmento KBD (Fig. 6) si rivolge all'utente con immediatezza e non richiede come il precedente una gestione della struttura di memoria. La struttura di KBD, realizzata al SIM da Laura Bianchini, è prevista per il massimo rendimento delle strutture di sintesi, per questa ragione presenta una minore flessibilità ma raggiunge una più elevata prestazione in termini di voci simultanee (20 strumenti FM o 40 oscillatori in sintesi additiva o 20 strumenti in AM – 20 Khz di campionamento).

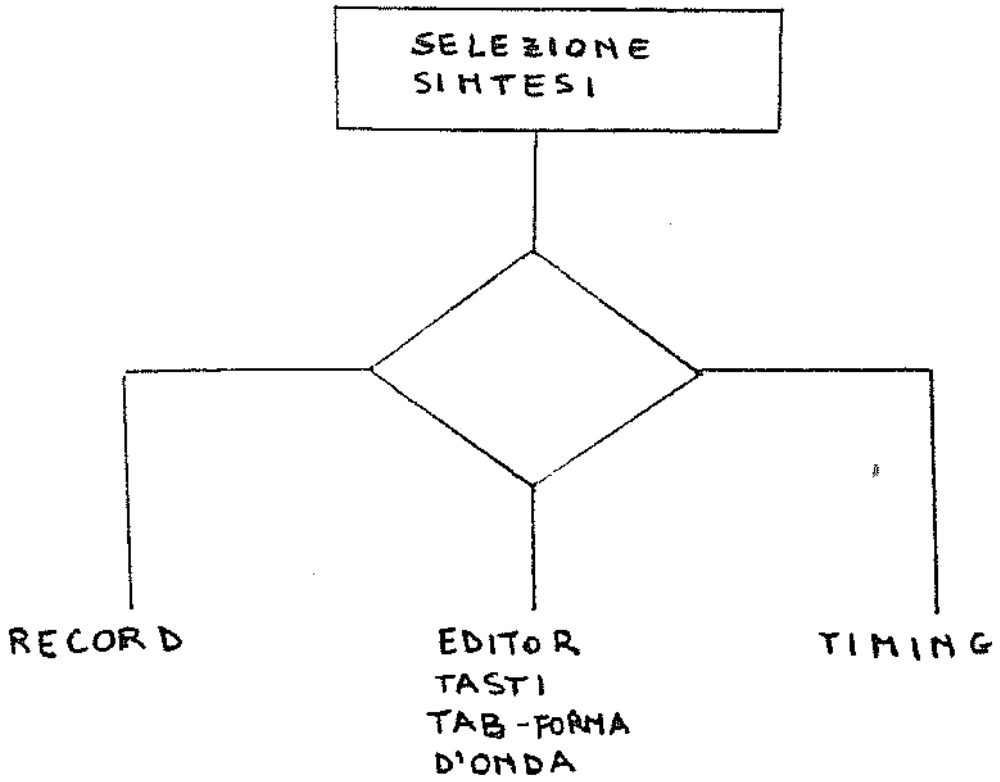

Figura 6

Bibliografia

M.V. MATHEWS, *The Technology of Computer Music*, Ed. Mit Press, 1969.

A.V. OPPENHEIM - R.W. SHAFER, *Digital Signal Processing*, Ed. Prentice Hill, 1975.

L. DEL DUCA, *Introduzione ai Filtri*, Ed. Quaderni di Informatica Musicale n. 2, S. E. Varese, Pescara.

S. SAPIR - G. DE POLI, *Verso Music5 in tempo reale: un software per il processore numerico di suoni 4I*, Atti del 5° Colloquio Informatica Musicale Univ., Ancona 1983.

G. NOTTOLI, *L'unità per l'analisi, elaborazione e sintesi del suono* SPU *ØI*, Atti del 5° Colloquio Informatica Musicale, Univ., Ancona 1983.