

**PROGETTO FINALIZZATO
SISTEMI INFORMATICI E CALCOLO PARALLELO**

SOTTOPROGETTO 7

Sistemi di Supporto al Lavoro Intellettuale
Coordinatore Cesare Maioli

Jacques Chareyron [^]

NOTE TECNICHE DEL MODULO

LASy

N. M/31

10/92

Rapporto Tecnico Serie MUSIC

[^] LIM - Laboratorio di Informatica Musicale
Dipartimento di Scienze dell'Informazione
Università degli Studi di Milano.

Capitolo 4: Algoritmi

4.1 Modo "68k"

4.1.1 Struttura del microprogramma

4.1.2 Utilizzo della memoria

4.2 Modo "56k"

4.2.1 Struttura del microprogramma

4.2.2 Utilizzo della memoria

4.3 Modo "56ks"

4.3.1 Struttura del microprogramma

4.3.2 Utilizzo della memoria

4.4 Poly

4.4.1 Struttura del microprogramma

4.4.2 Utilizzo della memoria

1 Introduzione

1.1 Generalità sull'applicazione LASy Workbench

Il LASy Workbench (LW) é un'applicazione sviluppata per la realizzazione dell'algoritmo di sintesi LASy (Linear Automata Synthesis) su un calcolatore Macintosh (dotato di una scheda Sound Accelerator o Audiomedia della Digidesign).

Il LW risponde ad una duplice funzione: si tratta di uno strumento di esecuzione musicale LASy controllabile in tempo reale con comandi MIDI o con periferiche di ingresso (tastiera alfa numerica e mouse); inoltre costituisce un ambiente di ricerca e di sperimentazione della sintesi LASy.

E' possibile passare in modo trasparente da l'uno all'altro modo di operazione, che generalmente vengono utilizzati in successione (prima ricerca e costruzione timbrica, poi utilizzo musicale degli strumenti elaborati). Una delle motivazioni alla base della concezione di LW é la speranza che il musicista, abituatosi senza difficoltà alle funzioni di "strumento" del LW, si avvicini alla comprensione della sintesi e alla costruzione completa dei propri strumenti. A questo scopo il LW contempla diversi livelli di sperimentazione e di elaborazione timbrica adatti ad altrettanti livelli di familiarità con l'algoritmo LASy.

1.2 Requisiti hardware per il corretto funzionamento

- Gli appartenenti alla famiglia Macintosh II con almeno due Mbyte di memoria RAM
- Digidesign Sound Accelerator Card
- Digidesign Audiomedia
- Un interfaccia MIDI compatibile con il Macintosh é necessaria per il controllo del LW da parte di apparecchiature Midi esterne (tastiera musicale, sequencer).

Il LW funziona con una scheda nella configurazione minima di memoria, ma certe funzioni vengono disabilitate in questo caso (in particolare la sintesi polifonica). E' fortemente consigliato una configurazione della scheda con RAM dati di 32 KWord o più per godere della totale potenzialità del LW.

1.3 Requisiti software per il corretto funzionamento

- System versione T1-6.0.7, Finder versione T1-6.1.7

1.4 Requisiti firmware per il corretto funzionamento

-Tutte le ROM dalla versione a 128 kbyte in poi.

1.5 Note per installazione

1.5.1 Hardware

Per l'installazione della scheda DSP, consultare il manuale della Digidesign. Le uscite audio della scheda devono essere collegate ad un dispositivo di amplificazione.

Per controllare il LW via Midi, collegare la presa MIDI_In dell'interfaccia MIDI alla presa MIDI Out dell'apparecchiatura MIDI utilizzata

1.5.2 Software

L'applicazione comporta un solo archivio che può essere copiato su hard disk o altri supporti magnetici.

2 Struttura dell'applicazione

2.1 Le linee direttrici della realizzazione del LASy Workbench

La tecnica LASy é molto economica dal punto di vista computazionale. Infatti il LW prevede una versione dell'algoritmo che utilizza solo il microprocessore 680x0 del Macintosh per la sintesi e la visualizzazione della forma d'onda generata in tempo reale. La realizzazione di una versione polifonica (su 4 voci) ottenuta semplicemente usando un DSP generico come il Motorola 56000.

Possono sorgere invece problemi causati dalla dimensione delle tabelle numeriche richieste dall'algoritmo di sintesi. Finché queste tabelle sono memorizzate nella memoria principale dell'elaboratore non sorgono grosse difficoltà, ma la situazione diventa critica se le tabelle devono essere presenti nella memoria della scheda direttamente indirizzabile dal DSP, limitata a 64 Kword. Le schede della Digidesign non prevedono trasferimenti DMA tra il calcolatore ospite e il DSP; l'unico veicolo tra scheda e calcolatore é una porta a 8 bit che sopporta un carico molto limitato.

Questo collo di bottiglia, aggiunto alla scarsa disponibilità di memoria sulla scheda, rappresenta il principale limite nella realizzazione dell'algoritmo LASy.

Nonostante i limiti appena citati, la realizzazione dell'algoritmo LASy sulla piattaforma utilizzata può essere considerata soddisfacente e propone le funzionalità previste dall'uso senza compromessi di rilievo.

Una caratteristica essenziale del LW é la disponibilità di controlli sulla sintesi in tempo reale. Questi controlli devono:

- * aggiungere reali capacità espressivi al suono sintetizzato;
- * essere di uso semplice e intuitivo;
- * non aggiungere un carico di lavoro eccessivo all'elaboratore per non influire negativamente sull'efficienza dell'algoritmo di sintesi.

Nella versione attuale del LW sono stati implementati un numero limitato di controlli in tempo reale che rispondono alle esigenze appena elencate. Una parte corrisponde a controlli di sintesi classici (modulazione della

frequenza e dell'ampiezza del suono,...), l'altra parte permette di modificare elementi specifici alla sintesi LASy (modifica della regola di transizione attiva,...). L'utente interagisce con essi in modo intuitivo usando il mouse o un controllo presente su un dispositivo Midi.

I controlli presenti nella versione attuale offrono varietà e espressività al suono generato. Non sono stati sfruttate però tutte le possibilità di controllo legate alle caratteristiche dell'algoritmo LASy, quest'ultimo sarà un punto principale da esaminare per la realizzazione di ulteriori versioni del LW.

Le componenti del LW non legate direttamente alla sintesi riguardano da una parte la costruzione di nuovi strumenti, forme d'onde, involucri e regole di transizione, dall'altra la gestione e l'archiviazione di questi componenti.

I moduli relativi alla costruzione di uno strumento devono adattarsi alle particolarità dell'algoritmo di sintesi. La costruzione di una forma d'onda é un'operazione del tutto classica, offerta nel LW in due versioni: scelta delle armoniche (ampiezza e fase) presenti nel segnale o costruzione diretta della forma d'onda con segmenti consecutivi. La costruzione di una regola di transizione é invece un problema nuovo, é richiede l'ideazione di schemi nuovi per semplificare un processo non intuitivo. Una volta disegnati e programmati in modo modulare questi schemi l'aggiunta di nuovi algoritmi di costruzione di regole di transizione diventa una procedura abbastanza agevole; la difficoltà rimane quella dell'ideazione di algoritmi semplici e potenti.

La gestione delle tabelle di strumenti, forme d'onde e regole di transizione presenta diversi problemi:

- * memorizzazione delle tabelle e dei loro elementi sotto una forma efficiente e maneggevole;
- * ottimizzazione dell'occupazione di memoria;
- * realizzazione di un'interfaccia utente veloce ed intuitiva.

Queste necessità hanno richiesto la creazione di strutture di dati e di procedure di gestione abbastanza complesse; questa complessità viene ripagata da una grande flessibilità e scioltezza nell'uso. L'utente ha dimestichezza con l'interfaccia Macintosh non ha difficoltà a manipolare i dati usati dal LW.

2.2 Scelte implementative

2.2.1 Precisione dei dati usati nell'algoritmo di sintesi

L'hardware a disposizione permette una conversione digitale-analogica a 16 bit, di conseguenza è logico usare tale dimensione durante l'elaborazione della sintesi LAsy. La dimensione delle tabelle numeriche che rappresentano le regole di transizione aumenta però esponenzialmente rispetto al numero di bit di ogni elemento. Nel caso di una regola basata sulla somma dei valori dei tre elementi di un vicinato, tale tabelle contiene 768 word nel caso di una definizione di 8 bit, 12288 word nel caso di una definizione di 12 bit, e ben 192K word nel caso di una definizione di 16 bit! Mantenere una grande scelta di regole di transizione nella memoria del calcolatore può essere problematico quando si usa la definizione massima. La situazione peggiora se per motivi di efficienza si decide di mantenere la tabella nella memoria direttamente indirizzabile dal DSP, che non può superare i 64K word da 24 bit.

Durante la realizzazione del LW é stato scelto di eseguire la computazione della nuova forma d'onda su parole di 12 bit, e di effettuare un'interpolazione lineare al momento della lettura della tabella secondo la frequenza corrente per ottenere un valore codificato su 16 bit da mandare al convertitore. Questa soluzione rappresenta il massimo consentito se si vuole memorizzare un'intera tabella di regole di transizione nella memoria della scheda, e mantiene il livello di rumore introdotto dalla minore precisione dei calcoli ad un livello accettabile per applicazioni musicali.

2.2.2 Variazioni in frequenza del suono sintetizzato

Nella versione di base dell'algoritmo di sintesi, un nuovo campione viene mandato al DAC ogni volta che viene aggiornato lo stato di una cella. La frequenza f del suono emesso é fissa e dipende di due parametri:

- * la lunghezza della forma d'onda;
- * la frequenza di campionamento F , secondo la formula $f = F/l$.

Per applicazioni musicali, la frequenza del suono deve poter essere controllata nell'arco delle altezze percettibili dall'orecchio umano. Ci sono principalmente tre metodi per modificare la frequenza del suono emesso:

1) Variazione della lunghezza della forma d'onda.

Questo metodo presenta il grosso inconveniente di limitare il numero di frequenze disponibili (la lunghezza della tabella é un numero intero). La precisione ottenibile di é molto scarsa e diminuisce con l'altezza della nota. Questa restrizione impedisce ogni impiego nel campo musicale.

2) Variazione della frequenza di campionamento.

E' possibile modificare via software la frequenza di campionamento del DAC della scheda DSP con una buona precisione. La componente analogica della scheda (in particolare i filtri) consentono però una qualità ottimale del segnale per la frequenza di campionamento di default pari a 44.1 MHz. L'inconveniente maggiore di questa soluzione risiede nell'impossibilità di realizzare una vera sintesi polifonica, con l'emissione contemporanea di suoni con frequenze differenti.

3) Separazione delle operazioni di calcolo della nuova forma d'onda e della conversione digitale-analogica.

Questa soluzione richiede l'esecuzione asincrona dei due processi. La forma d'onda viene aggiornata ciclicamente secondo la regola di transizione vigente. Parallelamente, un campione viene prelevato dalla tabella della forma d'onda per ogni periodo di campionamento. La posizione del campione da prelevare viene calcolata in funzione della frequenza desiderata. Questo metodo consente la generazione con grande precisione di qualsiasi frequenza, ma presenta una differenza dall'algoritmo di base: il campione mandato al DAC non é più quello che é stato appena calcolato. L'assenza di sincronismo tra le due operazioni non produce però modifiche delle componenti spettrali del segnale generato, e le esperienze uditive non hanno messo in evidenza una differenza significativa tra i risultati dei due tipi di elaborazione.

La terza soluzione si é rivelata l'unica accettabile per applicazioni musicali, e non ha comportato l'introduzione di inconvenienti maggiori. Bisogna notare però che la velocità dell'evoluzione del suono generato dipende dalla velocità del calcolo di una nuova forma d'onda, introducendo una differenza percettiva tra il risultato dell'algoritmo che utilizza il processore del calcolatore e quello che utilizza il DSP della scheda, tra la sintesi monofonica e quella polifonica.

2.2.3 Lunghezza della tabella della forma d'onda

Più lunga è la tabella che rappresenta la forma d'onda, minore è il rumore introdotto nella generazione di frequenze che non corrispondono ad un intervallo di lunghezza fissa tra la posizione di due campioni letti in successione. Ma l'aumento della lunghezza di questa tabella aumenta proporzionalmente il lavoro computazionale necessario al calcolo di una nuova generazione dell'automa. Per limitare la lunghezza della tabella mantenendo il rumore aggiunto ad un livello minimo si può calcolare il valore del campione corrente a mezzo di un'interpolazione lineare. In pratica si tratta di leggere i valori di due campioni successivi nella tabella e di calcolare il valore intermedio che corrisponde al campione virtuale determinato dalla frequenza esatta del suono. Questo metodo dà buoni risultati anche con tabelle di lunghezza relativamente limitata e aggiunge un carico di lavoro limitato al processo di sintesi. La lunghezza della tabella usata nel LW è di 512 word, e sembra un buono compromesso tra le esigenze segnalate in questo paragrafo.

2.3 I moduli del LASy Workbench

L'applicazione LW può essere intesa come composizione di diversi moduli rappresentati nella pagina seguente.

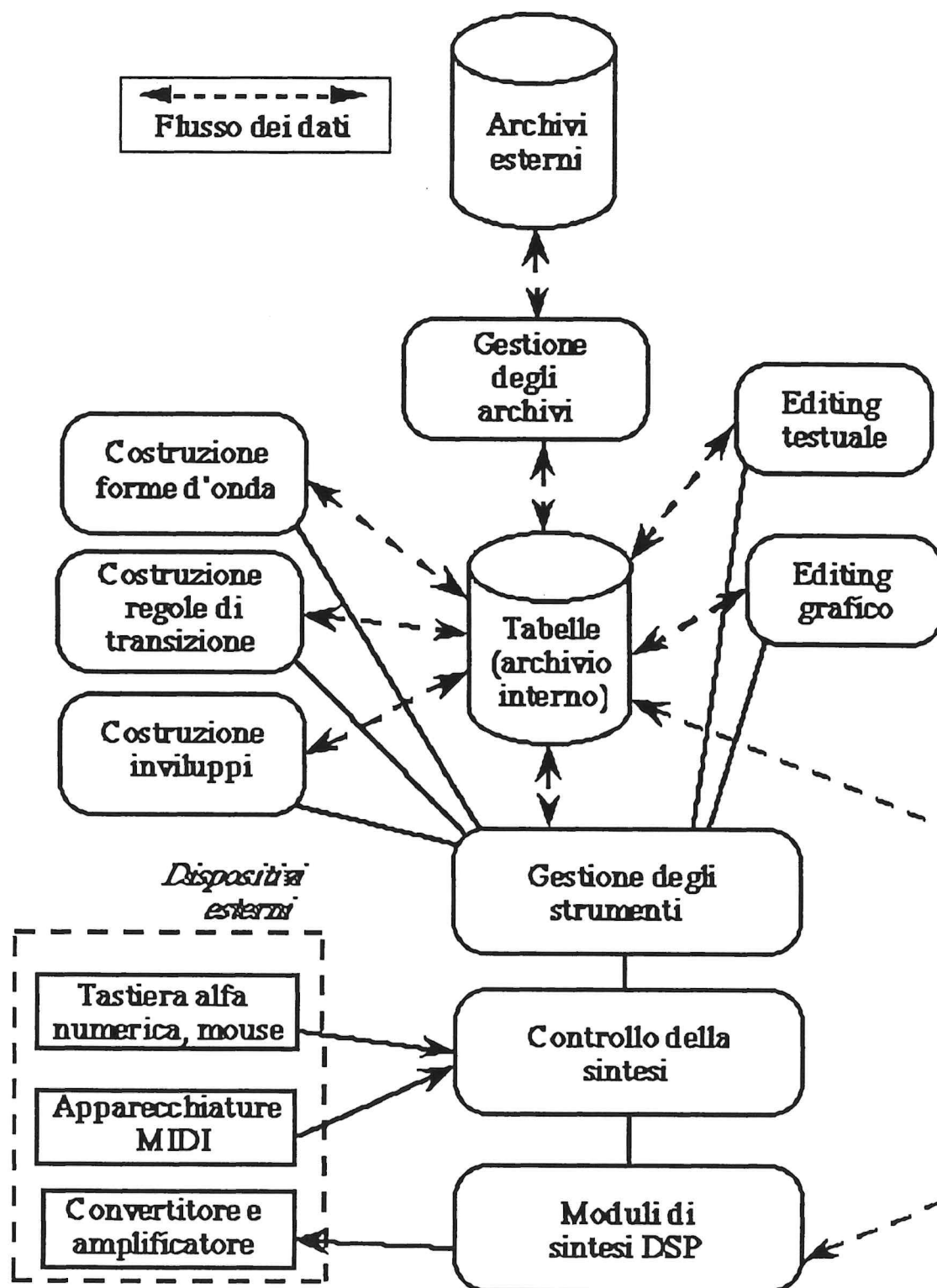
* I moduli di sintesi:

Tali moduli sono codificati in linguaggio Assembler del DSP Motorola 56000 (nel modulo "Synthesis 68k" il calcolo della nuova generazione dell'automa è codificato in assembler 680x0, e il modulo in assembler 56000 ha solo il carico di gestire l'invio del campione giusto a secondo della frequenza corrente). Questi moduli sono archiviati come resource di tipo SCD nell'applicazione LW, e caricati nella memoria di programma della scheda ogni volta che viene scelta una nuova opzione del menu "Synthesis".

* Il modulo di esecuzione in tempo reale e di controllo della sintesi.

Tale modulo esegue la sintesi del suono secondo i parametri precisati dall'utente e gestisce la corretta interpretazione dei controlli in tempo reale. Il modulo è codificato in Assembler 680x0 le cui procedure sono chiamate dal programma principale codificato in linguaggio "C".

A questo modulo si aggiungono le routine di visualizzazione in tempo reale del grafico della forma d'onda.



* Il modulo di gestione delle librerie di dati.

I compiti di archiviazione delle tabelle di strumenti, forme d'onde e regole di transizione vengono devolute a questo modulo, che gestisce anche in modo trasparente l'organizzazione in memoria dei dati.

* Il modulo di costruzione di strumenti, forme d'onda e regole di transizione.

Questo modulo contiene gli algoritmi di costruzione dei diversi tipi di dati usati dal LW e gestisce i dialoghi collegati a questa costruzione.

La struttura del modulo permette di aggiungere facilmente altri algoritmi di costruzione.

* I moduli di editing di dati

Sono presenti due moduli per l'editing delle tabelle di forme d'onda e regole di transizione. Il primo agisce sui dati in modo testuale (l'utente modifica i dati numerici introducendo nuovi valori), il secondo in modo grafico (i dati vengono modificati con l'aiuto del mouse).

* Il modulo Midi

Trovano posto in questo modulo le routine relative al controllo della sintesi da parte di un'apparecchiatura Midi esterna. Questo modulo utilizza il Midi Manager vers. 2.0 ed é compatibile con il software che rispetta lo stesso standard.

Durante il funzionamento del LW sono presenti contemporaneamente il programma principale (che utilizza la CPU 680x0) e un microprogramma caricato sulla scheda Digidesign (che utilizza il DSP 56000).

* Il programma principale.

L'esecuzione segue un ciclo principale nel quale vengono generati eventi a secondo delle azioni dell'utente (sulla tastiera, il mouse,...) e delle necessità interne del sistema. Questi eventi vengono trasmessi ai moduli software incaricati di gestirli.

Quando viene richiesta la sintesi sonora il programma entra in un loop secondario a dove vengono presi in considerazione solo gli eventi specifici al controllo in tempo reale della sintesi, in modo di assicurare un responso immediato all'azione dell'utente.

Fanno anche parte del programma principale le routine di interrupt chiamate su domanda del microprogramma DSP.

* Il microprogramma DSP

Ogni opzione di sintesi del menu "Synthesis" é associata al proprio microprogramma. Il microprogramma é caricato nella memoria di

programma della scheda quando l'utente sceglie una di queste opzioni e inizia subito la sua esecuzione.

I microprogrammi hanno tutti la stessa struttura e comportano una routine di inizializzazione, un ciclo principale e tre tipi di routine di interrupt, SSInterrupt, UnderrunInterrupt e HostInterrupt.

Il ciclo di inizializzazione deve solo accertarsi che la CPU é pronta a comunicare con la scheda, dopodiché si entra nel ciclo principale.

Il ciclo principale controlla se una nuova forma d'onda deve essere caricata a partire dalla RAM del calcolatore e calcola la nuova forma d'onda per ognuno degli oscillatore attivi nel tipo di sintesi usata (queste ultime routine non esistono nel microprogramma "Synthesis 68k", nel quale il calcolo della nuova forma d'onda viene eseguito dal processore 680x0).

La routine SSInterrupt viene fornisce la frequenza di campionamento. Un nuovo campione viene calcolato a partire dei valori della forma d'onda, tenendo conto della frequenza e del volume correnti, e viene poi mandato al convertitore digitale-analogico.

La routine di UnderrunInterrupt viene chiamata per segnalare che non é stato trasmesso un campione al DAC in tempo utile. Questa routine non viene mai chiamata nel funzionamento normale del LW e si accontentata di segnalare l'errore al programma principale.

Le routine di HostInterrupt viene chiamata ogni volta che un interrupt viene generato dal CPU (vedi sotto).

* Comunicazione tra CPU e DSP

La comunicazione tra il programma principale e i microprogrammi avviene in due modi: passaggio di dati attraverso l'unico canale di comunicazione disponibile e le chiamate di interrupt.

Lo scambio dei dati (un byte alla volta) avviene scrivendo in una particolare locazione dello spazio di memoria gestito dal processore (680x0 o 56000) dopo la verifica (leggendo un'altra locazione) che il ricevente é pronto a accettare il messaggio.

Nel senso CPU-DSP vengono trasmesse le tabelle di forme d'onda e di regole di transizione e i valori dei parametri usati nella sintesi (volume, frequenza, modo,...).

Nel senso DSP-CPU vengono trasmessi i codici che indicano l'azione da effettuare dopo un interrupt causato dalla scheda.

Il meccanismo di interrupt funziona nelle due direzioni: il programma principale può causare un HostInterrupt che interrompe il

microprogramma, e il microprogramma può causare uno SlotInterrupt che interrompe il programma principale.

Ci sono 14 tipi di HostInterrupt che vengono dirottati verso altrettante routine del microprogramma. Gli HostInterrupt usati nel LW sono i seguenti (non tutti gli HostInterrupt vengono utilizzati in tutti i modi di sintesi):

- * HostInterrupt 0: trasmette un nuovo valore del volume.
- * HostInterrupt 1: inizializza i parametri della sintesi.
- * HostInterrupt 2: trasmette un nuovo valore della frequenza.
- * HostInterrupt 3: trasmette un nuovo valore del modo di sintesi.
- * HostInterrupt 4: usato durante il periodo di debugging dell'applicazione
- * HostInterrupt 5: segnale di Note On (solo in modo Mono).
- * HostInterrupt 6: segnale di Note Off (solo in modo Mono).
- * HostInterrupt 7: precede la trasmissione di una regola di transizione
- * HostInterrupt 8: cambia il rango di una regola di transizione.
- * HostInterrupt 10: segnale di Note On (solo in modo Poly).
- * HostInterrupt 11: segnale di Note Off (solo in modo Poly).
- * HostInterrupt 13: trasmette un parametro generico.

L'emissione di un HostInterrupt può essere associata alla trasmissione di uno o più parametri (valore del volume, della frequenza,...).

Esiste un solo tipo di SlotInterrupt, ma le diverse routine da eseguire vengono identificate da un codice trasmesso dal microprogramma immediatamente dopo il segnale di interrupt. Il microprogramma interrompe il programma principale per:

- * richiedere la trasmissione di una nuova forma d'onda;
- * richiedere la trasmissione di un nuovo inviluppo;
- * richiedere la trasmissione di una nuova regola di trasmissione;
- * segnalare un errore nella trasmissione seriale.

3 Strutture dati

3.1 Generalità

Le tabelle di strumenti, forme d'onda, inviluppi e regole di transizione sono ricoverate in memoria durante l'esecuzione. A questi dati dinamici (che vengono memorizzate all'apertura della finestra corrispondente e cancellati alla sua chiusura) si aggiungono un insieme di dati statici che occupano una zona fissa della memoria del calcolatore:

- * La tabella numerica che rappresenta la forma d'onda attiva.
- * Le tabelle numeriche che rappresentano le forme d'onda utilizzate nei tre tipi di modulazione.
- * Le tabelle numeriche che rappresentano le quattro regole di transizione attive.
- * Le tabelle numeriche che rappresentano i tre inviluppi attivi.

Tali strutture di dati statici sono referenziate per mezzo di puntatori e occupano una posizione fissa in memoria. Per non frammentare lo spazio di memoria, tali strutture vengono raggruppate nella parte bassa dello heap.

Tutti gli altri dati vengono referenziati per mezzo di handle gestiti dal Memory Manager del Macintosh. Se la memoria disponibile é insufficiente alla creazione o l'ampliamento di una tabella di dati, un dialogo informa l'utente, il quale può archiviare e chiudere una tabella presente in memoria prima di ripetere l'operazione.

3.2 Strutture di base

3.2.1 Strumenti

Una struttura **instrument** é una tabella numerica che contiene i valori dei parametri che caratterizzano uno strumento software usato dall'algoritmo LASy. Questa struttura é rappresentata nella figura. Segue una breve spiegazione del contenuto di ogni campo.

0	Length	32	Control 1 number
1	Mode	33	Control 2 number
2	EnvTabA	34	
3	EnvA	35	
4	EnvTabF	36	
5	EnvF	37	
6	EnvTabWM	38	
7	EnvWM	39	
8	WaveTabFM	40	WaveTab
9	WaveFM	41	Wave
10	IndFM	42	Rule1Tab
		43	Rule1
12	FreqFM	44	Rule1Alg
		45	Rule2Tab
14	WaveTabAM	46	Rule2
15	WaveAM	47	Rule2Alg
16	IndAM	48	Rule3Tab
		49	Rule3
18	FreqAM	50	Rule3Alg
		51	Rule4Tab
20	WaveTabWM	52	Rule4
21	WaveWM	53	Rule4Alg
22	IndWM	54	Attack
		55	PropA
24	FreqWM	56	PropB
		57	
26	Midi mode + Channel	58	
27	Velocity	59	
28	Pitch Bend	60	
29	After Touch	61	
30	Control 1	62	
31	Control 2	63	

0) Length:

lunghezza totale (in word da 16 bit) della struttura che rappresenta il presente strumento.

1) Mode:

modo di sintesi.

2) EnvTabA:

numero d'ordine della tabella d'involuppo d'ampiezza.

3) EnvA:

numero d'ordine dell'involuppo d'ampiezza.

4) EnvTabF:

numero d'ordine della tabella d'involuppo di frequenza.

5) EnvF:

numero d'ordine dell'involuppo di frequenza.

6) EnvTabWM:

numero d'ordine della tabella d'involuppo della Wave Modulation.

7) EnvWM:

numero d'ordine dell'involuppo della Wave Modulation.

8) WaveTabFM:

numero della tabella di forme d'onda utilizzata per l'applicazione della modulazione in frequenza.

9) WaveFM:

numero d'ordine della forma d'onda utilizzata per l'applicazione della modulazione in frequenza.

10) IndFM:

indice utilizzato per l'applicazione della modulazione in frequenza. La parte intera dell'indice viene memorizzata nel word alto di IndF, la parte decimale nel word basso.

12) FreqFM:

frequenza utilizzata per l'applicazione della modulazione in frequenza. La parte intera della frequenza viene memorizzata nel word alto di FreqF, la parte decimale nel word basso.

14) WaveTabAM:

numero della tabella di forme d'onda utilizzata per l'applicazione della modulazione in ampiezza.

15) WaveAM:

numero d'ordine della forma d'onda utilizzata per l'applicazione della modulazione in ampiezza.

16) IndAM:

indice utilizzato per l'applicazione della modulazione in ampiezza. La parte intera dell'indice viene memorizzata nel word alto di IndA, la parte decimale nel word basso.

18) FreqAM:

frequenza utilizzata per l'applicazione della modulazione in ampiezza. La parte intera della frequenza viene memorizzata nel word alto di FreqA, la parte decimale nel word basso.

20) WaveTabWM:

numero della tabella di forme d'onda utilizzata per l'applicazione della wave modulazione.

21) WaveWM:

numero d'ordine della forma d'onda utilizzata per l'applicazione della wave modulazione.

22) IndWM:

indice utilizzato per l'applicazione della wave modulazione. La parte intera dell'indice viene memorizzata nel word alto di IndWM, la parte decimale nel word basso.

24) FreqWM:

frequenza utilizzata per l'applicazione della wave modulazione. La parte intera della frequenza viene memorizzata nel word alto di FreqWM, la parte decimale nel word basso.

26) Midi mode + Channel: il byte basso caratterizza il modo di ricezione Midi (0 = Omni, 1 = Poly), il byte alto è il numero del canale Midi attivo in modo Poly.

27) Velocity:

interpretazione del parametro di Velocity.

28) Pitch Bend:

interpretazione dei messaggi di Pitch Bend ricevuti durante l'emissione di una nota.

29) After Touch:

interpretazione dei messaggi di After Touch ricevuti durante l'emissione di una nota.

30) Pitch Bend:

interpretazione dei messaggi del controller numero uno ricevuti durante l'emissione di una nota.

31) Pitch Bend:

interpretazione dei messaggi del controller numero due ricevuti durante l'emissione di una nota.

- 32) **Control 1 number:**
numero reale del controller visto dal LW come Control1.
- 33) **Control 2 number:**
numero reale del controller visto dal LW come Control2.
- 33-39) Riservati.
- 40) **WaveTab:**
numero della tabella di forme d'onda utilizzata.
- 41) **Wave:**
numero d'ordine della forma d'onda utilizzata.
- 42) **Rule1Tab:**
numero della prima tabella di regole di transizione utilizzata .
- 43) **Rule1:**
numero d'ordine della prima regola di transizione utilizzata.
- 44) **Rule1Alg:**
numero del tipo di algoritmo richiesto nell'applicazione della prima regola di transizione.
- 45) **Rule2Tab:**
numero della seconda tabella di regole di transizione utilizzata (questo campo é significativo solo nel caso di un modo di sintesi 2, 3 o 4).
- 46) **Rule2:**
numero d'ordine della seconda regola di transizione utilizzata (questo campo é significativo solo nel caso di un modo di sintesi 2, 3 o 4).
- 47) **Rule2Alg:**
numero del tipo di algoritmo richiesto nell'applicazione della seconda regola di transizione (questo campo é significativo solo nel caso di un modo di sintesi 2, 3 o 4).
- 48) **Rule3Tab:**
numero della terza tabella di regole di transizione utilizzata (questo campo é significativo solo nel modo di sintesi 4).
- 49) **Rule3:**
numero d'ordine della terza regola di transizione utilizzata (questo campo é significativo solo nel modo di sintesi 4).
- 50) **Rule3Alg:**
numero del tipo di algoritmo richiesto nell'applicazione della terza regola di transizione (questo campo é significativo solo nel modo di sintesi 4).
- 51) **Rule4Tab:**
numero della quarta tabella di regole di transizione utilizzata (questo campo é significativo solo nel caso di un modo di sintesi 1, 2, 3 o 4).

52) Rule4:

numero d'ordine della quarta regola di transizione utilizzata (questo campo é significativo solo nel caso di un modo di sintesi 1, 2, 3 o 4).

53) Rule4Alg:

numero del tipo di algoritmo richiesto nell'applicazione della quarta regola di transizione (questo campo é significativo solo nel caso di un modo di sintesi 1, 2, 3 o 4).

54) Attack:

valore del parametro di attacco (questo campo é significativo solo nel caso di un modo di sintesi 2 o 4).

55) PropA:

valore del parametro prop1 (questo campo é significativo solo nel caso di un modo di sintesi 3 o 4).

56) PropB:

valore del parametro prop2 (questo campo é significativo solo nel caso di un modo di sintesi 3 o 4).

57-63) Riservati.

3.2.2 Forme d'onda

Una struttura **wave** é una tabella numerica (di lunghezza variabile) formata dagli elementi seguenti (il primo numero è l'offset in word dall'inizio della tabella):

0) lunghezza della struttura (in word da 16 bit);

1 a 512) la tabella dei valori che descrivono la forma d'onda;

513) il byte basso indica il modo di costruzione della forma d'onda (0 = costruzione per armoniche; 1 = costruzione per segmenti). Il byte alto contiene nel bit più significativo un flag settato quando la forma d'onda ha subito operazioni di editing dopo la sua costruzione.

513 a 572) trenta coppie di parametri utilizzati nella costruzione della forma d'onda. Se la forma d'onda è stata costruita per armoniche, ogni coppia contiene l'ampiezza e la fase di una delle trenta prime armoniche presenti nel segnale rappresentato dalla forma d'onda. Se la forma d'onda è stata costruita per segmenti, ogni coppia contiene le coordinate degli estremi di uno dei ventinove segmenti (al massimo) che compongono la forma d'onda.

a partire da 573) è codifica delle modifiche portate alla forma d'onda in fase di editing.

3.2.3 Regole di transizione

Una struttura **rule** é una tabella numerica (di lunghezza variabile) che contiene i valori dei parametri che caratterizzano una regola di transizione usata dall'algoritmo LASy, seguiti eventualmente dalle modifiche subite dalla regola di transizione in fase di editing. La parte ripetitiva di questa struttura é rappresentata nelle figure. Segue una breve spiegazione del contenuto di ogni campo.

1	IndSup
2	Prop
3	
4	Rand
5	F1 + MinMode
6	a1
8	b1
9	F2 + MaxMode
10	a2
12	b2
13	Min
14	LimInf
15	Max
16	LimSup

1	IndSup
2	LimInf2
3	LimSup2
4	Rand
5	F1
6	a1
8	b1
9	F2
10	a2
12	b2
13	Min
14	LimInf
15	Max
16	LimSup

0) Length:

lunghezza totale (in word da 16 bit) della struttura che rappresenta la regola di transizione.

1) LPar:

nel caso di una regola non espansa, lunghezza dell'insieme dei parametri usati nella costruzione della regola di transizione.

2) **Type:**

indicatore del tipo di regola di transizione. Il byte basso di questo campo indica il tipo di algoritmo di costruzione usato per la regola di transizione. Il bit 8 indica se la regola di transizione é stata memorizzata in modo espanso (tabella dei valori utilizzati direttamente dall'algoritmo di sintesi) o in modo compatto (tabella dei parametri utilizzati per costruire la tabella dei valori). Il bit 12 é settato quando la regola di transizione é simmetrica. Il bit 15 è settato quando la regola di transizione ha subito modifiche dopo la sua costruzione.

3) **Size:**

parametro che indica la lunghezza della regola di transizione (una volta espansa). La lunghezza reale si ottiene moltiplicando il valore del campo Size per 4096.

4 e 5) **Riservati.**

Dopo questi, presenti in ogni struttura di regola di transizione, troviamo:

- * la tabella dei valori che rappresentano la regola di transizione (se questa è memorizzata in modo "espanso")

- * la tabella dei parametri usati durante l'operazione di costruzione della regola di transizione (nel caso contrario). Questi parametri si presentano come una successione di sottostrutture identiche; i parametri che formano ogni sottostruttura dipendono del tipo di regola di transizione (parametro fisso di offset 2).

Ogni sottostruttura è composta dei seguenti parametri (la ripetizione dello stesso valore di offset indica che il parametro corrispondente ha un significato diverso a secondo del tipo di regola di transizione):

0) **IndSup :**

limite superiore dell'intervallo sul quale devono essere usati i parametri che seguono (fino al prossimo campo IndSup o alla fine della struttura).

1) **LimInf2:**

secondo valore al di sotto del quale viene verificata la condizione di underflow nel caso di una regola calcolata con l'algoritmo 1.

2) **LimSup2 :**

secondo valore al di sopra del quale viene verificata la condizione di overflow nel caso di una regola calcolata con l'algoritmo 1.

3) **Rand :**

grado di casualità applicato al risultato del calcolo di ogni elemento della tabella.

4) **F1:**

il byte basso é l'indice della funzione f1 utilizzata nel calcolo di ogni elemento della tabella numerica che caratterizza la regola di transizione. Il byte alto é interpretato come un flag; quando questo flag é settato, vengono usati valori assoluti nel calcolo della regola di transizione in caso di underflow; nel caso contrario vengono usati valori relativi.

Per una regola di transizione di tipo 0, questo campo contiene anche nel byte alto il parametro **MinMode** che il comportamento dell'automa quando il valore calcolato scende sotto il valore minimo consentito.

5) **a1:**

valore del parametro a1 utilizzato nel calcolo della regola di transizione. La parte intera del parametro viene memorizzata nel word alto di A1, la parte decimale nel word basso.

7) **B1:**

valore del parametro b1 utilizzato nel calcolo della regola di transizione.

8) **F2 :**

il byte basso é l'indice della funzione f2 utilizzata nel calcolo di ogni elemento della tabella numerica che caratterizza la regola di transizione. Il byte alto é interpretato come un flag; quando questo flag é settato, vengono usati valori assoluti nel calcolo della regola di transizione in caso di overflow; nel caso contrario vengono usati valori relativi.

Per una regola di transizione di tipo 0, questo campo contiene anche nel byte alto il parametro **MaxMode** che il comportamento dell'automa quando il valore calcolato supera il valore massimo consentito.

9) **a2 :**

valore del parametro a2 utilizzato nel calcolo della regola di transizione. La parte intera del parametro viene memorizzata nel word alto di A2, la parte decimale nel word basso.

11) **B2 :**

valore del parametro b2 utilizzato nel calcolo della regola di transizione.

12) **Min :**

valore utilizzato per il calcolo degli elementi della tabella che caratterizza la regola di transizione in caso di underflow.

13) **LimInf:**

valore al di sotto del quale è verificata la condizione di underflow.

14) **Max:**

valore utilizzato per il calcolo degli elementi della tabella che caratterizza la regola di transizione in caso di overflow.

15) LimSup:

valore al di sopra del quale viene verificata la condizione di overflow.

3.2.4 Inviluppi

Una struttura **Envelope** é una tabella numerica (di lunghezza fissa uguale a 193 word) formata dagli elementi seguenti (il primo numero è l'offset in word dall'inizio della tabella):

0) lunghezza della struttura (in word da 16 bit);

1 a 128) la tabella dei valori utilizzati dal modulo di sintesi per applicare l'inviluppo al segnale generato. Questa tabella comporta due gruppi formati dei 32 valori codificati su 4 byte (o due word) che corrispondono ai 32 segmenti autorizzati per l'inviluppo. Il primo gruppo contiene i valori dell'incremento che subisce il valore corrente dell'inviluppo ad ogni iterazione, il secondo gruppo il valore finale raggiunto dall'inviluppo alla fine del segmento.

129 a 193) la tabella dei valori dei parametri precisati dall'utente nella fase di costruzione dell'inviluppo. Questa tabella comporta due gruppi formati dei 32 valori codificati su un word che corrispondono ai 32 segmenti autorizzati per l'inviluppo. Il primo gruppo contiene i valori della lunghezza del segmento, il secondo gruppo il valore finale raggiunto dall'inviluppo alla fine del segmento.

3.3 Tabelle di strumenti, forme d'onde, regole di transizione e inviluppi

Una tabella di strumenti, forme d'onde, regole di transizione e inviluppi raccoglie un gruppo di elementi della stessa natura (fino ad un massimo teorico di 32768). Nella memoria del calcolatore una tabella di strumenti, forme d'onde, regole di transizione e inviluppi é costituita da una tabella di **Handle** (puntatori a puntatori). Tali handle sono memorizzati (come long word da 32 bit) in posizioni contigue della memoria. Ognuno di questi handle punta (dopo doppia deferenzazione) a uno elemento memorizzato secondo una delle strutture descritte nelle sezioni 3.2.1, 3.2.2, 3.2.3 e 3.2.4. L'uso di una tabella di handle (al posto di semplici puntatori) permette di ottimizzare l'uso della memoria evitando la sua frammentazione.

Una tabella di strumenti, forme d'onde, regole di transizione e involucri ha una lunghezza variabile, che dipende dalla creazione o la cancellazione di nuovi elementi. Per evitare il moltiplicarsi dei spostamenti di strutture in memoria, lo spazio per gli handle viene attribuito o soppresso a gruppi di dieci.

Una tabella di strumenti, forme d'onde, regole di transizione e involucri non occupa una posizione fissa in memoria. La posizione di ogni tabella viene ricordata in quattro array di handle di tabelle, collocati nella zona delle variabili globali. La dimensione di questi array autorizza la presenza in memoria di dieci tabelle per ogni tipo di elemento oltre alla tabella di default.

3.4 Archivi

3.4.1 File di strumenti

Un file di strumenti é un file sequenziale che contiene le informazioni necessarie alla memorizzazione su supporto magnetico di una tabella di strumenti. Al contrario di un file completo di strumenti, un file di strumenti **non** contiene le tabelle di forme d'onda e di regole di transizione usate da questi strumenti. Queste tabelle vengono memorizzate in file di forme d'onda e di regole di transizione e devono essere usate insieme al file di strumenti per ricostituire tutti i componenti di uno strumento.

Un file di strumenti contiene i seguenti campi:

* **NumIns** (1 word):

numero di strumenti memorizzati nel file.

* **NameIns1, NameIns2,...** (NumIns * 32 word):

nome degli strumenti memorizzati nel file. Ognuno dei **NumIns** nomi é una tabella di trentadue caratteri (compreso lo zero che segnala la fine di una stringa di caratteri "C").

* **Ins1, Ins2,...** (NumIns * 64 word):

descrizione degli strumenti. Ognuno dei **NumIns** elementi é una tabella dei parametri che definisce uno strumento, come descritto nella sezione 3.2.1.

3.4.2 File di forme d'onda

Un file di forme d'onda é un file sequenziale che contiene le informazioni necessarie alla memorizzazione su supporto magnetico di una tabella di forme d'onda.

Un file di forme d'onda contiene i seguenti campi:

* **NumWave** (1 word):

numero di forme d'onda memorizzati nel file.

* **NameWave1, NameWave2,...** (NumWave * 32 word):

nome delle forme d'onda memorizzate nel file. Ognuno dei **NumWave** nomi é una tabella di trentadue caratteri (compreso lo zero che segnala la fine di una stringa di caratteri "C").

* **Wave1, Wave2,...** (NumWave elementi di lunghezza variabile): descrizione delle forme d'onda. Ognuno dei **NumWave** elementi é una tabella dei parametri che definisce una forma d'onda, come descritto nella sezione 3.2.2.

3.4.3 File di regole di transizione

Un file di regole di transizione é un file sequenziale che contiene le informazioni necessarie alla memorizzazione su supporto magnetico di una tabella di regole di transizione.

Un file di regole di transizione contiene i seguenti campi:

* **NumRule** (1 word):

numero di regole di transizione memorizzate nel file.

* **NameRule1, NameRule2,...** (NumRule * 32 word):

nome delle regole di transizione memorizzate nel file. Ognuno dei **NumRule** nomi é una tabella di trentadue caratteri (compreso lo zero che segnala la fine di una stringa di caratteri "C").

* **Rule1, Rule2,...** (somma delle lunghezza del NumRule regole di transizione):

descrizione delle regole di transizione. Ognuno dei **NumRule** elementi é una tabella dei parametri che definiscono una regola di transizione, come descritto nella sezione 3.2.2.

3.4.4 File di involucri

Un file di strumenti é un file sequenziale che contiene le informazioni necessarie alla memorizzazione su supporto magnetico di una tabella di involucri.

Un file di involucri contiene i seguenti campi:

* **NumEnv** (1 word):

numero di strumenti memorizzati nel file.

* **NameEnv1, NameEnv2,...** (NumEnv * 32 word):

nome degli involucri memorizzati nel file. Ognuno dei **NumEnv** nomi é una tabella di trentadue caratteri (compreso lo zero che segnala la fine di una stringa di caratteri "C").

* **Env1, Env2,...** (NumEnv * 193 word):

descrizione degli involucri. Ognuno dei **NumEnv** elementi é una tabella dei parametri che definiscono un involucro, come descritto nella sezione 3.2.4.

3.4.5 File completo di strumenti

Un file completo di strumenti é un file sequenziale che contiene tutte le informazioni necessarie alla memorizzazione su supporto magnetico di una tabella di strumenti. Al contrario di un file di strumenti, un file completo di strumenti contiene le tabelle di forme d'onda e di regole di transizione usate da questi strumenti. Basta caricare un file completo di strumenti per poter ricostruire completamente gli strumenti memorizzati in questo file.

Un file completo di strumenti contiene i seguenti campi:

* **NumIns** (1 word):

numero di strumenti memorizzati nel file.

* **NameIns1, NameIns2,...** (NumIns * 32 word):

nome degli strumenti memorizzati nel file. Ognuno dei **NumIns** nomi é una tabella di trentadue caratteri (compreso lo zero che segnala la fine di una stringa di caratteri "C").

* **Ins1, Ins2,...** (NumIns * 64 word):

descrizione degli strumenti. Ognuno dei **NumIns** elementi é una tabella dei parametri che definiscono uno strumento, come descritto nella sezione 3.2.1.

* **NumWave** (1 word):

numero di forme d'onda usate dagli strumenti memorizzati nel file.

* **NameWave1, NameWave2,...** (NumWave * 32 word):

nome delle forme d'onda usate dagli strumenti memorizzati nel file. Ognuno dei **NumWave** nomi é una tabella di trentadue caratteri (compreso lo zero che segnala la fine di una stringa di caratteri "C").

* **Wave1, Wave2,...** (somma delle lunghezze del NumWave forme d'onda):

descrizione delle forme d'onda usate dagli strumenti memorizzati nel file. Ognuno dei **NumWave** elementi é una tabella dei parametri che definisce una forma d'onda, come descritto nella sezione 3.2.2.

* **NumRule** (1 word):

numero di regole di transizione usate dagli strumenti memorizzati nel file.

* **NameRule1, NameRule2,...** (NumRule * 32 word):

nome delle regole di transizione usate dagli strumenti memorizzati nel file. Ognuno dei **NumRule** nomi é una tabella di trentadue caratteri (compreso lo zero che segnala la fine di una stringa di caratteri "C").

* **Rule1, Rule2,...** (somma delle lunghezze del NumRule regole di transizione):

descrizione delle regole di transizione usate dagli strumenti memorizzati nel file. Ognuno dei **NumRule** elementi é una tabella dei parametri che definiscono una regola di transizione, come descritto nella sezione 3.2.3.

* **NumEnv** (1 word):

numero di involucri usati dagli strumenti memorizzati nel file.

* **NameEnv1, NameEnv2,...** (NumEnv * 32 word):

nome degli involucri usati dagli strumenti memorizzati nel file. Ognuno dei **NumEnv** nomi é una tabella di trentadue caratteri (compreso lo zero che segnala la fine di una stringa di caratteri "C").

* **Env1, Env2,...** (NumEnv * 193 word):

descrizione degli involucri usati dagli strumenti memorizzati nel file. Ognuno dei **NumEnv** elementi é una tabella dei parametri che definiscono un involucro, come descritto nella sezione 3.2.4.

4 Algoritmi

Nei paragrafi seguenti vengono esaminati gli algoritmi di calcolo implementati nel microcodice presente sulla scheda DSP durante la generazione dei campioni e le loro conseguenze implementative.

4.1 Modo "68k"

4.1.1 Struttura del microprogramma

* Il programma principale.

Nel tipo di sintesi 68k il calcolo della nuova forma d'onda é presa in carica dal programma eseguito dal microprocessore 680x0 del Macintosh. Il programma principale del microcodice risulta perciò particolarmente semplice. Dopo la fase di inizializzazione, il loop principale esegue due tipi di compiti: il controllo dei flag che segnalano l'inizio e la fine di una nota e la verifica della necessità del caricamento di una nuova forma d'onda nella memoria dati della scheda DSP. A secondo del risultato di questi controlli vengono eseguite le operazioni necessarie: chiamata alla routine che genera uno slot interrupt nel Macintosh per richiede il caricamento di una nuova forma d'onda, modulazione del volume del segnale mandato al DAC, ecc.

* Le routine di interrupt.

La più importante è la SSInterrupt, chiamata ogni volta che viene richiesto l'invio di un campione alla porta seriale. Questa routine calcola la posizione nella tabella del campione da usare a seconda della frequenza richiesta (e delle eventuali modulazioni da applicare), calcola il valore del campione da utilizzare per interpolazione lineare e moltiplica il valore calcolato per il volume corrente dell'oscillatore.

La seconda routine viene chiamata solo in caso di underrun e segnala la situazione al LW. Questa routine non viene mai chiamata nel funzionamento normale del microprogramma.

* Le host command.

Ognuna di queste viene chiamata dal Macintosh per trasmettere dati o comandi al microprogramma. Vedere sezione 2.2.

4.1.2 Utilizzo della memoria

* La parte bassa della memoria dati X: (da \$0000 a \$0FFF) viene utilizzata per memorizzare le variabili usate nella sintesi: frequenza e volume dei diversi oscillatori, parametri dell'algoritmo di sintesi, indirizzi delle tabelle delle regole di transizione, flag vari. Troviamo anche in questa zona di memoria le variabili dove vengono salvaguardati i registri durante l'esecuzione delle routine di interrupt.

* La parte bassa della memoria dati Y: (da \$0000 a \$03FF) contiene la tabella della forma d'onda corrente.

I registri X0 e Y0 vengono usati unicamente all'interno della routine di interrupt. Gli altri registri sono liberamente utilizzati in diverse parti del microcodice, e vengono adeguatamente salvati e ripristinati all'inizio e alla fine delle routine di interrupt.

4.2 Modo "56k"

4.2.1 Struttura del microprogramma

* Il programma principale.

Dopo la fase di inizializzazione, il loop principale realizza l'aggiornamento delle forme d'onda secondo la regola di transizione corrente e i parametri dell'algoritmo scelto. Viene prima eseguito un controllo sul modo di sintesi corrente per decidere quale delle regole di transizione presenti in memoria deve essere utilizzata. Poi viene scandita la tabella, presente nella memoria dati, che corrisponde alla forma d'onda corrente. Per ogni campione viene chiamata la subroutine che corrisponde al tipo di algoritmo di sintesi corrente incaricata dell'aggiornamento del campione. Ad ogni ripetizione del loop principale vengono anche svolti compiti secondari. In particolare vengono controllati i flag che segnalano l'inizio e la fine di una nota. Nel primo caso viene chiamato una routine che genera uno slot interrupt nel Macintosh e richiede il caricamento di una nuova forma d'onda. Nel secondo caso il flusso del programma viene modificato (generalmente per modificare la regola di transizione usata).

* Le routine di interrupt.

La routine di SSI Interrupt determina la posizione nella tabella del campione da usare a seconda della frequenza richiesta, calcola il valore del campione da utilizzare per interpolazione lineare e moltiplica il valore calcolato per il volume corrente prima di mandarlo all'uscita seriale.

La seconda routine viene chiamata solo in caso di underrun e segnala la situazione al LW. Questa routine non viene mai chiamata nel funzionamento normale del microprogramma.

* Le host command. Vedere sezione 2.2.

4.2.2 Utilizzo della memoria

* La parte bassa della memoria dati X: (da \$0000 a \$0FFF) viene utilizzata per memorizzare le variabili usate nella sintesi: frequenza e volume dei diversi oscillatori, parametri dell'algoritmo di sintesi, indirizzi delle tabelle delle regole di transizione, flag vari. Troviamo anche in questa zona di memoria le variabili dove vengono salvaguardati i registri durante l'esecuzione delle routine di interrupt.

* La parte bassa della memoria dati Y: (da \$0000 a \$05FF) contiene le tabelle di forme d'onde. Due tabelle contengono due generazioni successive dell'automa (da \$0200 a \$05FF), la terza contiene la tabella della forma d'onda corrente nel formato adeguato all'applicazione dell'interpolazione lineare (da \$0000 a \$01FF).

* La parte alta della memoria dati X: e Y: (da \$1000 a \$3FFF per una memoria complessiva di 32k x 24 bit, da \$1000 a \$7FFF per una memoria complessiva di 64k x 24 bit) contiene le tabelle delle regole di transizione. Con una regola di transizione "classica", la configurazione minima consente la memorizzazione contemporanea di quattro tabelle combinabili in diversi modi per consentire la sintesi di un singolo suono. I registri X0 e Y0 vengono usati unicamente all'interno della routine di interrupt. Gli altri registri sono liberamente utilizzati in diverse parti del microcodice, e vengono adeguatamente salvati e ripristinati all'inizio e alla fine delle routine di interrupt.

4.3 Modo "56ks"

4.3.1 Struttura del microprogramma

* Il programma principale.

Lo schema presentato nella sezione precedente rimane valido per questo modo di sintesi, con una serie di restrizioni dovute alle restrizioni dello spazio di memoria (questo modo è stato concepito per autorizzare la generazione sonora internamente ad una scheda DSP con la dotazione minima di memoria dati che ammonta a 8 KWord). La limitazione della memoria disponibile consente il caricamento di una sola regola di transizione (vengono eliminati tutti i modi di funzionamento dello strumento tranne il modo 0) e non c'è spazio per il caricamento delle tabelle di inviluppo e di forme d'onda usate nelle diverse modulazioni, con la soppressione di queste funzioni della sintesi.

4.3.2 Utilizzo della memoria

* La parte bassa della memoria dati X: (da \$0000 a \$03FF) viene utilizzata per memorizzare le variabili usate nella sintesi: frequenza e volume dei diversi oscillatori, parametri dell'algoritmo di sintesi, indirizzi delle tabelle delle regole di transizione, flag vari. Troviamo anche in questa zona di memoria le variabili dove vengono salvaguardati i registri durante l'esecuzione delle routine di interrupt.

* La parte bassa della memoria dati Y: (da \$0000 a \$03FF) contiene le tabelle di forme d'onde che corrispondono a due generazioni successive dell'automa.

* La parte alta della memoria dati X: e Y: (da \$0400 a \$0FFF) contiene la tabella della regola di transizione corrente.

I registri X0 e Y0 vengono usati unicamente all'interno della routine di interrupt. Gli altri registri sono liberamente utilizzati in diverse parti del microcodice, e vengono adeguatamente salvati e ripristinati all'inizio e alla fine delle routine di interrupt.

4.4 Poly

4.4.1 Struttura del microprogramma

* Il programma principale.

Dopo la fase di inizializzazione, il loop principale realizza l'aggiornamento delle forme d'onda (una per ogni oscillatore virtuale) secondo la regola di transizione corrente e i parametri dell'algoritmo scelto. Viene prima eseguito un controllo sul modo di sintesi corrente per decidere quale delle regole di transizione presenti in memoria deve essere utilizzata. Poi vengono scandite in successione le quattro tabelle presenti nella memoria dati che corrispondono alla forma d'onda corrente di ognuno dei quattro oscillatori. Per ogni campione viene chiamata la subroutine che corrisponde al tipo di algoritmo di sintesi corrente incaricata dell'aggiornamento del campione. Ad ogni ripetizione del loop principale vengono anche svolti compiti secondari. In particolare vengono controllati i flag che segnalano l'inizio e la fine di una nota. Nel primo caso viene chiamata una routine che genera uno slot interrupt nel Macintosh e richiede il caricamento di una nuova forma d'onda. Nel secondo caso il flusso del programma viene modificato (generalmente per modificare la regola di transizione usata).

* Le routine di interrupt.

La SSInterrupt viene chiamata ogni volta che è necessario l'invio di un campione alla porta seriale. Si tratta di una routine abbastanza complessa nel caso della sintesi polifonica. Per ognuno dei quattro oscillatori virtuali, questa routine calcola la posizione nella tabella del campione da usare a seconda della frequenza richiesta (e delle eventuali modulazioni da applicare), calcola il valore del campione da utilizzare per questo oscillatore per interpolazione lineare e moltiplica il valore calcolato per il volume corrente dell'oscillatore. I quattro risultati vengono poi sommati e mandati all'uscita seriale.

La seconda routine di interrupt viene chiamata solo in caso di underrun e segnala questa situazione al LW. Questa routine non viene mai chiamata nel funzionamento normale del microprogramma.

* Le host command. Vedere sezione 2.2.

4.4.2 Utilizzo della memoria

* La parte bassa della memoria dati X: (da \$0000 a \$0FFF) viene utilizzata per memorizzare le variabili usate nella sintesi: frequenza e volume dei diversi oscillatori, parametri dell'algoritmo di sintesi, indirizzi delle tabelle delle regole di transizione, flag vari, variabili dove vengono salvaguardati i registri durante l'esecuzione delle routine di interrupt. Troviamo anche le tabelle dei tre tipi di involucri e le tabelle delle forme d'onda utilizzate nelle diverse modulazioni.

* La parte bassa della memoria dati Y: (da \$0000 a \$0FFF) contiene le tabelle di forme d'onde (due per oscillatore, che contengono due generazioni successive dell'automa).

* La parte alta della memoria dati X: e Y: (da \$1000 a \$3FFF per una memoria complessiva di 32k x 24 bit, da \$1000 a \$7FFF per una memoria complessiva di 64k x 24 bit) contiene le tabelle delle regole di transizione. Con una regola di transizione "classica", la configurazione minima consente la memorizzazione contemporanea di quattro tabelle combinabili in diversi modi per consentire la sintesi di un singolo suono. I registri X0 e Y0 vengono usati unicamente all'interno della routine di interrupt. Gli altri registri sono liberamente utilizzati in diverse parti del microcodice, e vengono adeguatamente salvati e ripristinati all'inizio e alla fine delle routine di interrupt.

