Sistemi di Elaborazione per la Musica

Ch.mo Prof. G.B.Debiasi
Ing. N.Orio

Controllo del Mixer
"Sound Engineer II" General Music
mediante la programmazione
del Controller Midi Peavey PC-1600.

Fabio Parise

Anno Accademico 1995/96

INDICE DEGLI ARGOMENTI

INTRODUZIONE

Scopo di questo lavoro è di interfacciare e programmare il Controller Midi Peavey 1600 (**PC-1600**) tramite un Personal Computer. Il controller permette di memorizzare al suo interno 50 differenti configurazioni (**Presets**) per i suoi elementi programmabili.
L'interfaccia offerta dal controller per la definizione delle varie funzioni è semplice ed abbastanza intuitiva, ma la mancanza di una tastiera alfanumerica ed il display LCD molto ridotto, rendono la programmazione molto lunga e noiosa. Per programmare uno dei 50 Preset si impiega circa un'ora.
La possibilità di interfacciare il controller con un PC è offerta dallo stesso controller, che a fronte di un messaggio Midi di System Exclusive: *'Require All Presets*', invia su Midi-OUT tutti i dati concernenti i 50 Presets memorizzati al suo interno.
La casa costruttrice non specifica però il formato di memorizzazione dei dati inviati, si deve perciò procedere alla decodifica del dump esadecimale inviato dal controller. Una volta decodificato il formato dei dati e realizzato il software di decodifica automatica delle informazioni, sarà possibile impostare da PC i parametri di programmazione del Peavey 1600 e poi spedirli al controller.
Sarà possibile anche creare copie di backup su disco dei Presets da ripristinare al momento opportuno, oppure creare e salvare su disco i parametri dei singoli Presets, realizzando un piccolo database di Preset residente nella memoria di massa del PC.

La seconda parte del lavoro consiste nel programmare il PC-1600 tramite Pc, in modo da poter controllare il Mixer Midi 'Sound Engineer II' (SE II) della General Music. Il 'SE II' è un Mixer Midi analogico a controllo completamente digitale a 64 ingressi e 16 uscite, dotato di numerosi controlli gestibili solamente via software, mediante la connessione con un Pc.
Il Mixer ha una duplice possibilità di collegamento con l'esterno: attraverso l'interfaccia seriale o attraverso l'interfaccia Midi (quest'ultima permette il collegamento con la maggior parte degli strumenti e dispositivi nell'ambito musicale).

Il controllo Midi può avvenire mediante messaggi Midi standard oppure mediante messaggi di System Exclusive.
Il Peavey deve essere impostato in modo da inviare questi comandi al Mixer, in modo da gestire le sue numerose funzioni.
La programmazione del Peavey avviene tramite un programma interprete di Script, che genera un file contenente la descrizione di un particolare Preset. Il Preset contiene i messaggi Midi impostati per il controllo del Mixer. Questo file viene poi inviato al PC-1600 tramite un apposito messaggio di SysEx e quindi memorizzato al suo interno.

E' stato scelto l'uso del controller Midi PC-1600 per controllare il mixer, in quanto si tratta di un prodotto di semplice utilizzo da parte di chiunque voglia utilizzarlo. Il "SE II" è fornito di un apposito software di controllo, ma questo oltre ad essere assai complesso da utilizzare, in alcuni casi presenta dei malfunzionamenti dovuti ai numerosi bug del software.
La necessità di effettuare conferenze all'esterno del C.S.C. inoltre costringerebbe al trasporto di un Pc al seguito del mixer, mentre utilizzare il Peavey per il suo controllo comporta pochissimi problemi di trasporto viste le sue ridotte dimensioni.
Il software è inoltre impostato in modo da poter essere facilmente ampliato. La strutturazione in due programmi separati dell'interprete di Script e del software di comunicazione con il PC-1600 rende semplice poter programmare il Peavey per il controllo di altri dispositivi Midi. Sarà sufficiente scrivere un programma (interprete di Script o meno) in grado di produrre un file .PRS con le impostazioni del Preset da programmare.
L'uso degli Script per la descrizione dei comandi diventa uno strumento semplice e potente se rivolto ad una utenza ristretta ed esperta.

NOZIONI BASE

L'Hardware

L'hardware necessario per la realizzazione del lavoro si trova al CSC (Centro di Sonologia Computazionale) presso il Centro di Calcolo di Ateneo dell'Università di Padova e consiste in:

· un Personal Computer IBM compatibile (CPU i80x86)
· un Controller Midi Peavey 1600
· una scheda Roland MIF-PC
· una Roland MPU-401
· un cavo di connessione a 25 pin
· due cavetti per l'interconnessione Midi (5 pin)

· il Mixer Midi "SE II" della General Music

Cos'è MIDI

MIDI (Musical Instrument Digital Interface) è uno standard di comunicazione adottato da tutti i costruttori di strumenti musicali, computer e apparecchi audio/video in generale. Esso in pratica stabilisce le specifiche hardware e software che tutti i dispositivi devono soddisfare per poter essere considerati MIDI-compatibili.
Le specifiche hardware stabiliscono le modalità di costruzione delle porte MIDI, dei circuiti di trasmissione e ricezione dei messaggi e dei cavi di connessione.
Le specifiche software definiscono invece, il linguaggio (il codice MIDI appunto), stabilendo la sintassi e le modalità con cui devono essere inviati i messaggi.

Lo standard di comunicazione Midi mette a disposizione 16 canali diversi (Midi Channel), su cui ogni dispositivo può ricevere e trasmettere dati. Si tratta di

'canali virtuali' messi a disposizione dall'interfaccia Midi, ma fisicamente i dati fluiscono tutti per lo stesso cavo di connessione e sulla stessa linea. Il funzionamento dei canali Midi può essere paragonato a quello dei canali televisivi, in cui per accedere ad una determinata trasmissione è sufficiente sintonizzarsi sul rispettivo canale. Allo stesso modo, ogni dispositivo Midi riceve e trasmette i dati di sua competenza solamente su di un ben definito canale Midi. Chiunque volesse operare con il dispositivo deve occuparsi di leggere e scrivere i dati nel canale su cui è sintonizzato il dispositivo scelto. Questo permette a numerosi dispositivi Midi interconnessi di colloquiare tra di loro senza problemi di interferenze.
Il numero di canale deve essere specificato in ogni messaggio Midi inviato. Nei messaggi Midi standard il canale è specificato nel primo byte trasmesso. Nei messaggi System Exclusive il numero di canale viene specificato all'interno del messaggio.

Messaggi Midi standard

Questo tipo di messaggi è il più comune ed il comando è identificato dal primo byte del messaggio, detto **Status Byte**. Lo Status Byte contiene le seguenti informazioni:

- i 4 bit più significativi contengono il tipo di comando
- i 4 bit meno significativi specificano il canale Midi usato.

Nota: il bit più significativo dello Status Byte deve sempre essere posto uguale ad 1. Serve per discernere i bytes dati dai caratteri di controllo dello standard Midi.

I messaggi standard definiti sono:

Messaggio Midi	Status Byte
Nota OFF	8*c*

Nota ON	9*c*
Poly Pressure	A*c*
Control Change	B*c*
Program Change	C*c*
Channel Pressure/after touch	D*c*
Pitch Wheel Change	E*c*

c = canale Midi (0..F)

Allo Status Byte seguono uno o due bytes dati, a seconda del comando, che specificano i valori dei parametri da impostare. I byte dati sono invece vincolati dal fatto di dover sempre avere il bit più significativo impostato a zero.

ES. Messaggio: Nota ON.
Parametri: valore della nota, velocità nota.

Codifica Midi: 9*c* 30 56.

Significato dei codici esadecimali:

9*c* : comando di Nota ON, sul canale Midi specificato da *c*.
30 : corrisponde alla nota Do3. (30 hex=48=12*4)
56 : velocità 56 hex = 86 dec.

Midi System-Exclusive

I System Exclusive sono dei messaggi Midi speciali, specifici per un determinato Hardware. Sono definiti dalla casa costruttrice e riportati nei manuali delle apparecchiature Midi. Permettono ad un qualsiasi dispositivo o programma che li conosca, di inviare alle apparecchiature connesse con l'interfaccia Midi, dei comandi che consentono di selezionare alcune funzioni od impostare i valori di alcuni parametri. E' possibile ad esempio alzare il volume di un amplificatore o modificare le impostazioni di un mixer tramite un messaggio System Exclusive.

Permettono inoltre di colloquiare con queste apparecchiature, chiedendo i valori di stato o facendosi spedire in risposta i dati desiderati.
Ogni dispositivo può ricevere messaggi Midi di ogni tipo, anche messaggi destinati ad altre apparecchiature e che quindi non riconosce. Il comportamento seguito in questo caso è semplice: tutti i messaggi che entrano da Midi-IN vengono automaticamente copiati su Midi-THRU, se il dispositivo riconosce il messaggio (dalla sua intestazione), lo preleva ed esegue ciò che è indicato al suo interno. In questo modo tutti i messaggi non riconosciuti da dispositivo, lo attraversano senza procurare alcun effetto, procedendo verso gli ulteriori dispositivi connessi.

Ogni messaggio SysEx inizia con il valore F0 hex (SysEx Status) e termina con F7 hex (End of Exclusive).
Questi messaggi non hanno vincoli di lunghezza e contengono la seguente sequenza di informazioni:

· Codice di identificazione della casa proprietaria
· Tipo di dispositivo
· Tipo di comando
· Ulteriori dati

Per esempio il messaggio: F0 00 00 1B 0B 00 00 pp F7

è un SysEx del PC-1600 che ordina al controller di selezionare il Preset indicato dal byte 'pp' (Recall Preset).

Nota: I bytes spediti tra i due marker F0 ed F7 devono tutti avere il bit più significativo (MSb) posto a zero, poiché il MSb = 1 è riservato per i caratteri di controllo della trasmissione Midi, quali appunto F0, F7, FF, ... etc.

Connessione Pc-dispositivo Midi

Per la comunicazione tra il Personal Computer e i dispositivi Midi, utilizziamo una scheda Roland MIF-PC da inserire in uno slot del Pc ed una Roland MPU-401 (Midi Processor Unit) disponibile su box esterno.
Le due schede vengono collegate fisicamente tramite il cavo a 25 pin. La MPU-401 così collegata al Pc è dotata delle connessioni Midi-IN e Midi-OUT, con le quali ci si può collegare ad un qualsiasi dispositivo dotato di un'interfaccia Midi.

Per la connessione di un Pc con più dispositivi Midi usiamo la porta Midi-THRU dell'interfaccia Midi.
Ogni messaggio che perviene al dispositivo attraverso la porta Midi-IN viene copiato sulla porta di uscita Midi-THRU. In questo modo connettendo la porta Midi-IN del secondo dispositivo alla Midi-THRU del primo, il secondo riceve una copia esatta di tutti i messaggi Midi che pervengono al primo.

Primitive di comunicazione Midi

A questo punto l'elaboratore è dotato dell'interfaccia Midi, ma non sono presenti alcune primitive per lo scambio dati con l'interfaccia.
Ho perciò realizzato la libreria IO401.PAS contenente tre primitive scritte in linguaggio Assembly per i80x86, che permettono l'utilizzo della MPU-401 per l'invio e la ricezione di dati Midi.

Le primitive sono:

· Function **GetMIDIdata** : INTEGER

Riceve un byte di dati Midi dalla MPU-401 e lo ritorna come valore della funzione. Ritorna il valore -1 se non ci sono dati disponibili o in caso di errore.

· Function **PutMIDIdata**(x : INTEGER) : INTEGER

Invia un byte di dati Midi all'interfaccia MPU-401. Ritorna -1 in caso di errore.

· Function **PutMIDIcmd**(x : INTEGER) : INTEGER

Invia un comando alla MPU-401 e ritorna il byte di acknowledgement FE hex se il comando è stato ricevuto, -1 in caso di errore.

Verranno usati solamente due comandi della MPU-401 : **UART** (Universal Asinchronous Receive and Transmit) che disabilita le funzioni intelligenti del dispositivo e passa i dati ricevuti, senza apportarvi alcuna modifica, da e verso i dispositivi cui è connesso.
Questa modalità serve per utilizzare la MPU-401 come mezzo di collegamento fra Pc e dispositivi Midi.
Al termine della connessione il comando **SYSRESET** (System Reset) riporta alla modalità di funzionamento normale dell'unità per l'uso da parte di qualche altro programma.

I DISPOSITIVI

Il Controller Midi : PEAVEY 1600.

Il PC-1600 è un Controller Midi molto versatile, progettato sia per applicazioni dal vivo che per l'utilizzo in studio. E' fornito di una discreta varietà di dispositivi programmabili essenziali per l'editing, aggiustamenti di precisione e mixing di studio.
Sono disponibili:

- 16 cursori programmabili (Faders)
- 16 pulsanti programmabili (Buttons)
- 2 ingressi controllati in tensione programmabili (CV1, CV2) per le pedaliere.
- una Data Wheel programmabile
- una Setup String programmabile in grado di inviare ai dispositivi collegati una qualsiasi serie di comandi Midi, ogni volta che un Preset viene richiamato. Serve per impostare dei valori iniziali nei vari dispositivi controllati.

Nota: mentre tutti gli altri sono elementi legati a dispositivi hardware, la Setup String è una funzionalità prettamente software, legata alla fase di inizializzazione del Preset di appartenenza.

Tutte queste impostazioni vengono memorizzate in un Preset del PC-1600. Il PC-1600 offre la possibilità di memorizzare 50 differenti Preset, corrispondenti a 50 diverse configurazioni del controller.

Ogni Preset consente di memorizzare in una stringa alfanumerica di 16 caratteri il proprio nome, utile per caratterizzare la sua particolare funzione di utilizzo.
Per esempio il preset 00 fornito dalla casa: "Volume with Mute" assegna ai cursori i codici per la regolazione dei volumi ed ai pulsanti la funzione di Mute (disabilitazione) del rispettivo cursore.

Oltre alla possibilità di programmare ogni controllo del PC-1600, è anche possibile assegnare ad ognuno di questi una stringa di valori contenente un comando di System Exclusive (**SysEx**), che verrà spedita su Midi-OUT ad ogni azione sul controllore. Ad esempio il movimento di un cursore o la pressione di un pulsante.
E' anche possibile raggruppare un insieme di cursori sotto il controllo di un altro cursore, detto **Master**. Questa funzionalità permette un facile sub-mixing.

Il Mixer Midi : "Sound Engineer II"

Il "Sound Engineer II" (SE II) della General Music è un sofisticato mixer analogico a controllo completamente digitale. Le sue numerose funzionalità possono essere selezionate solamente via software, mediante la connessione con un Personal Computer.
Il Mixer ha una duplice possibilità di collegamento con l'esterno: attraverso l'interfaccia seriale (mediante il codice SMPTE) o attraverso l'interfaccia Midi. Quest'ultima permette il collegamento con la maggior parte degli strumenti e dispositivi nell'ambito musicale.
Il codice SMPTE consente di sincronizzare "SE II" con registratori multitraccia, sequencers ed apparecchiature di post-produzione audio e video.
Il collegamento Midi consente il controllo integrale di tutti i parametri audio del mixer. Questo è reso possibile attraverso una serie di codici di System Exclusive riconosciuti da "SE II". Il mixer è anche controllabile, tramite comandi Midi standard quali Note-ON, Note-OFF, Poly Press, Control Change, etc ..., che vengono opportunamente interpretati secondo una codifica interna del mixer. Questi non danno il controllo globale di tutte le funzionalità del mixer, ma hanno il pregio di essere messaggi Midi più corti dei SysEx: servono solo 3 bytes per specificare un comando Midi standard contro i minimo 7 per un messaggio System Exclusive.
Viene perciò preferito, dove ciò sia possibile, l'utilizzo di comandi Midi di tipo standard, limitando l'uso dei System Exclusive ai casi realmente necessari. Lo scopo è quello di salvaguardare il sistema da problemi di intasamento delle linee

di comunicazione. I comandi Midi standard sono i più adatti anche per essere generati da un controller Midi programmabile (quale il PC-1600), data la sua predisposizione all'invio di messaggi Midi standard e le sue ridotte capacità di memoria.
Il "SE II" riceve messaggi attraverso il canale Midi memorizzato nel file di start-up 'START.JOB' presente nell'unità a dischi del mixer e caricato automaticamente ogni volta che il mixer viene acceso con il disco inserito nel disk drive.
Il canale Midi può anche essere impostato via software tramite il messaggio SysEx: '*Custom Set Standard Midi Channel*', specificato dalla seguente sequenza esadecimale:

F0 2F 7ch 40 [MidiChannel] F7
|
Canale Midi da impostare

Si assume in questa sede come canale di default per il mixer il canale zero.
Il mixer è corredato da 8 schede di ingresso, ognuna corrispondente a 4 coppie di ingressi bilanciati (A-B), per un totale di 64 canali in linea, di cui 32 utilizzabili contemporaneamente. Ad ognuno dei 32 canali di ingresso del mixer è associata una coppia di ingressi bilanciati, referenziati rispettivamente con A e B. Nella programmazione del mixer possiamo scegliere in riferimento ad ogni canale, se usare l'ingresso A o l'ingresso B, oppure se commutare tra i due durante le operazioni. Tutte le impostazioni di canale rimangono inalterate dopo un'operazione di cambio dell'ingresso da A a B o viceversa (input switch).
Questo permette di aver disponibili in linea un notevole numero di dispositivi nello stesso momento, senza essere obbligati a cambiare i cablaggi del mixer ad ogni nuova configurazione.
Per quanto riguarda le uscite sono disponibili due schede:

· scheda Master: contiene due uscite stereo bilanciate: L, R, Sub_L, Sub_R e le 4 uscite bilanciate AUX1, AUX2, AUX3 ed AUX4.

· scheda Group: comprende 8 uscite bilanciate dette Group. Ogni uscita Group riunisce le uscite di diversi canali che necessitano di un controllo di volume comune.

Risultano perciò un totale di 16 uscite disponibili.

Funzionalità del Mixer

· Canali di ingresso

Per ogni canale di ingresso è possibile impostare numerosi parametri. Il modulo di ingresso del canale esegue l'elaborazione del segnale presente in linea dalla corrispondente presa abilitata.
E' possibile migliorare la sensibilità del canale attraverso controlli di pre-amplificazione (Gain), equalizzare il segnale ed assegnare il canale alle uscite generali Master, ai Group o agli AUX.

· Uscite

Per ogni canale di uscita è possibile controllare il livello del volume ad esso associato. Questo avviene attraverso i comandi Master.

Per l'elenco delle funzioni del mixer, attivabili con sequenze Midi, si veda il paragrafo riguardante i comandi dei file script di SCRIPT.EXE.

PROGRAMMAZIONE DEL PC-1600

Analizziamo nel dettaglio le singole funzioni che possono essere assegnate ai vari controlli del PC-1600:

■ **FADERS:**

Si tratta di cursori comunemente usati per inviare dati come controller continui, c'è tuttavia la possibilità di associarvi una stringa Midi o di definire un Fader come Master di un altro gruppo di Faders. Le possibili impostazioni sono:

· **No Message**: nessuna funzione.

· **Continuous Controller**: il Fader è programmato per spedire un messaggio come controller continuo quando viene mosso.

E' possibile selezionare il numero di canale Midi (**Chnl** = 1..16), il numero del controller da inviare (**Num** = 1..120) (ES. #7 per il controllo del Volume). **Min** specifica il valore minimo inviabile (cursore basso) e **Max** specifica il massimo valore inviabile (cursore tutto alto).

Per cambiare il verso del Fader è sufficiente scambiare i due valori di Min e Max.

Il parametro **Mode** determina come viene gestita la collisione tra i messaggi che entrano nel PC-1600 (da Midi-IN) e quelli generati dallo stesso PC-1600. Vi sono tre opzioni:

Merge = i messaggi entranti nel controller che combaciano con quelli di qualche Fader, vengono mantenuti inalterati all'uscita. I messaggi dovuti al movimento di qualche Fader vengono miscelati con quelli entranti.

Replace = funziona analogamente al modo Merge, ma quando un Fader viene mosso, i messaggi entranti vengono filtrati e rimpiazzati con i dati generati dal Fader.

Update = viene usato quando il PC-1600 è connesso tra due dispositivi ed uno di loro trasmette dati del tipo 'controller continuo' all'altro (i dati attraversano il PC-1600). Il PC-1600 è usato in questa modalità per poter editare i dati entranti, prima di passarli alla destinazione.

· **Master Fader**: con questa opzione è possibile raggruppare un insieme di cursori sotto il controllo di un altro cursore, detto il **Master Fader**. Si devono specificare tutti i cursori gestiti dal Master.

· **Midi String**: è possibile memorizzare un qualsiasi messaggio Midi, inserendo i valori esadecimale che lo compongono. E' anche possibile inserire la posizione corrente del cursore nel messaggio inserendo il carattere 'pr' nella stringa da inviare. Inoltre è possibile definire l'intervallo dei valori tra i quali il parametro può variare.

Nota: se mentre stiamo editando un Fader, il PC-1600 riceve un messaggio SysEx, il messaggio viene memorizzato come Midi String associata a quel Fader.

■ BUTTONS:

I pulsanti (o Buttons) vengono usati esclusivamente per funzioni di tipo on/off. Possono essere usati per interagire con i rispettivi Faders attraverso le funzioni di Muting (disabilitazione) e Soloing degli stessi. E' anche possibile assegnare al pulsante l'invio di un messaggio di Note ON/OFF, di un Program Change o di una prefissata stringa Midi.
Le possibili impostazioni sono:

· **No Message**: nessuna funzione.

· **Mute Fader**: viene associata al Button la funzione di Mute del rispettivo Fader, impostando il Fader al minimo valore possibile e poi disattivandolo. Quando il tasto è premuto una seconda volta il Fader viene riabilitato.

· **Solo Fader**: viene associata al Button la funzione di Solo del rispettivo Fader. Vengono mandati al minimo tutti gli altri Fader e poi disabilitati. Resta attivo solo il Fader associato al pulsante premuto. Per riattivare gli altri Fader è sufficiente muoverli.

· **Program Change**: viene spedito un messaggio di Program Change, ogni volta il pulsante viene premuto. Si possono impostare: il Canale Midi (1..16) ed il Program Number (0..127) che indica il Program Change da inviare. Scegliendo ChOut come Canale Midi, viene utilizzato il canale specificato da Main Midi Channel, nella sezione Utility del PC-1600.

· **Note On/Off**: è possibile assegnare il Button in modo che invii un messaggio di Note-ON quando premuto ed un messaggio di Note-OFF quando viene rilasciato.
Si devono impostare: il canale Midi (1..16), Note: il valore della nota da suonare (DO_{-1} => Note=0,...etc), Vel: valore della velocità per il comando di nota.

· **Midi String**: il Button è associato ad una stringa Midi, che viene spedita al momento della pressione dello stesso. Se poi durante la fase di editing della stringa, viene ricevuto dal PC-1600 un messaggio System Exclusive, il controller lo memorizza automaticamente al posto della stringa. (Funzione di apprendimento automatico.)

· **String Press/Release**: è possibile associare ad un Button due stringhe: una che viene inviata alla pressione del pulsante ed una al suo rilascio.

· **String Toggle**: al Button possiamo associare due stringhe che vengono inviate alternativamente una volta una, una volta l'altra. (Utile per azioni del tipo abilita/disabilita).

■ DATA WHEEL

E' possibile programmare il controllore Data Wheel, collegandolo con un Fader o con una pedaliera tramite l'ingresso CV1 o CV2. Si può scegliere di collegarla ad un determinato Fader oppure all'ultimo che è stato mosso. CV1 e CV2 sono degli ingressi esterni al Peavey controllabili in tensione (CV = Control Voltage). Mediante questi due ingressi è possibile associare un messaggio di controllo Midi ad un dispositivo esterno esattamente come per un controllore del Peavey. Un esempio di utilizzo di questa funzionalità, è la connessione del PC-1600 con una pedaliera esterna. La pedaliera ha un comportamento da controllore continuo.
Dal punto di vista del tipo di controllo, la programmazione degli ingressi CV1 e CV2 è identica a quella dei Faders.

A Data Wheel possono essere assegnati i seguenti controllori: Fader 1..16, CV1, CV2, Last Fader.

■ SETUP STRING

Ogni volta che un Preset viene selezionato c'è la possibilità di inviare ad ognuno dei 16 canali Midi rispettivamente un messaggio di selezione di banco, un Program Change e l'impostazione del volume. Inoltre è possibile inviare ulteriori messaggi Midi tramite una stringa di aggiuntiva di al più 80 caratteri. Tutte queste informazioni vengono spedite ogni volta che il Preset viene selezionato.
E' molto utile per impostare i valori iniziali dei parametri che la configurazione che stiamo caricando dovrà utilizzare.

STRUTTURA DI CODIFICA DEI DATI NELLA MEMORIZZAZIONE DEI PRESET DEL CONTROLLER MIDI: PEAVEY 1600.

PREMESSA

Il formato di memorizzazione dei dati inviati dal Peavey 1600, è frutto dell'analisi dei tabulati contenenti il dump esadecimale dei dati trasmessi via Midi.

Poiché nel manuale di riferimento del controller non vi è alcun riferimento al particolare formato di memorizzazione delle informazioni, ne' alla loro strutturazione nel file trasmesso, l'analisi è dovuta partire da zero. Numerosi tentativi e raffinamenti successivi hanno portato ad una interpretazione organica di ogni singolo byte dei dati trasmessi.

La decodifica del formato presentava delle difficoltà intrinseche dovute alla completa carenza di qualsiasi informazione riguardante i formati di memorizzazione dei dati, quali gli interi o le stringhe alfanumeriche (Ad Es. ASCII, EBCDIC, formato proprietario).

L'analisi di configurazioni di stringhe con strutture note e particolari (ad es. simmetriche 'ABCDEFFEDCBA', progressive 'ABCDEFGHIJKL' o con piccole variazioni nella struttura 'AAAAAABAAAAA'), ha permesso di identificare lo standard di codifica dei caratteeri alfanumerici come lo standard ASCII, oltre ad aver dato un contributo determinante alla decodifica della struttura del formato.

Un altro problema per la decodifica sorge dallo standard di comunicazione Midi, che permette di inserire solo 7 bit di informazione per ogni byte trasmesso, riservando il bit più significativo per distinguere tra caratteri di controllo Midi e dati.

Come avviene allora la codifica di un byte di informazione ?

La risposta a questa domanda la troveremo nelle pagine seguenti. Questo fattore impone uno stadio di decodifica delle informazioni ricevute dal Peavey per tradurle dal formato di spedizione (con max 7 bit di informazione trasmessa), alla rappresentazione finale con informazioni ad 8 bit pieni.

A complicare il lavoro di decodifica c'è anche la natura dinamica della memorizzazione dei dati, da cui anche la necessità di inserire all'interno degli stessi dati dei puntatori indicanti uno spiazzamento nel file.
Questo per consentire un facile accesso alle varie strutture del file, senza doverlo percorrere sequenzialmente dall'inizio ad ogni operazione.

Si tratta di un enorme cruciverba a schema libero in cui si deve trovare il significato di ogni singolo valore presente.
L'analisi ha reso necessario lo sviluppo di software aggiuntivo, da me realizzato 'ad hoc' per la manipolazione e conversione della grossa mole di dati (circa 120 Kb di codici esadecimali). Questo ha consentito operazioni di ricerca di configurazioni simili ad una data e la possibilità di aver numerose rappresentazioni degli stessi dati dalle quali estrapolare informazioni utili.

Questa parte del lavoro ha messo a dura prova le mie capacità di analisi e le mie conoscenze del settore ed ha avuto un ruolo fondamentale per l'esito dell'intero lavoro.

LA CODIFICA

I dati riguardanti le informazioni sui Preset del PC 1600 vengono inviati dallo stesso su Midi Out in risposta al messaggio System Exclusive:

Send All Presets: F0 00 00 1B 0B ch 11 F7

ch = numero di canale.

Formato di spedizione dati di Peavey 1600

Il controller Midi Peavey 1600 per spedire un byte di informazione (256 valori), spedisce 2 bytes, ognuno corrispondente ad una cifra esadecimale del byte da trasmettere. Ogni byte trasmesso porta perciò un'informazione di soli 4 bit corrispondente alla cifra esadecimale trasmessa.

ES. voglio trasmettere il valore 127 Dec
trasformo in Hex: 7F Hex
prendo le due cifre Hex e le trasmetto separatamente: 07 Hex, 0F Hex.

Due bytes trasmessi per un byte di informazione.

Infatti i bytes trasmessi devono avere il MSb (Most Significant bit) sempre uguale a zero, per le regole della trasmissione via Midi System Exclusive. Per trasmettere un byte sono allora necessari almeno 2 bytes. Questo metodo è spartano e poco efficiente, ma consente una semplice codifica e riconversione delle informazioni, sia a livello di hardware della macchina, che per il programmatore.

Nota: D'ora in poi faremo riferimento sempre ai bytes di informazione trasmessi e non alla loro codifica per la spedizione.

Struttura del dump dei dati sui Preset

La memorizzazione dinamica delle informazioni da parte di Peavey 1600 ha reso particolarmente ostica la decodifica dei codici esdecimali di descrizione dei Preset, spediti attraverso l'interfaccia Midi, in risposta al comando di Dump dei Preset.
Qui di seguito ne riporto la descrizione:

■ I primi due bytes spediti non soddisfano al formato di spedizione appena descritto. Si tratta di :

$32 = codifica esadecimale di 50 (numero di Preset memorizzati). Forse per la compatibilità con altri modelli della casa che utilizzano lo stesso formato.

$00 = nessun significato trovato.

Poiché il PC1600 invia oltre a tutti i Preset (50) anche il contenuto dell'attuale Edit Buffer, avremo 51 entità memorizzate che chiameremo "**Preset Descriptor**".
Questi vengono memorizzati nel seguente ordine:

Preset00 ... Preset49, Edit Buffer.

■ Elenco di tutti i nomi dei Preset, più quello dell' Edit Buffer, nell'ordine menzionato, memorizzati nel formato **PresetNAME**.

■ Elenco di 52 puntatori (2 bytes ciascuno) che indicano la posizione relativa dei singoli Preset nel file, a partire dal byte seguente all'ultimo puntatore (il 52°). L'ultimo puntatore sembrerebbe inutile, visto che memorizziamo solo 51 Preset, ma presumo serva ad identificare la fine dei dati. Punta al carattere F7 (End of System Exclusive).

■ 51 blocchi del tipo **Preset Descriptor**. (50 Preset + Edit Buffer)

PresetNAME = stringa di 16 caratteri codificati secondo lo standard ASCII indicante il nome del Preset memorizzato.

Struttura di un "PRESET DESCRIPTOR".

Ogni Preset contiene i dati di ciascun elemento programmabile del PEAVEY 1600.
Questo Controller Midi programmabile permette di programmare i seguenti dispositivi:

· 16 cursori (Fader #1..16)
· 16 pulsanti (Button #1..16)
· 2 ingressi controllati in tensione (CV1 e CV2)
· Data Wheel
· Possibilità di scegliere una Setup String inviata su Midi-OUT ad ogni richiamo di Preset.

Ogni Preset mantiene le impostazioni di questi dispositivi memorizzate nel seguente ordine:

Fader01 .. Fader16, CV1, CV2, Button01 .. Button16, Data Wheel, Setup String.

NOTA: Non vi è nessun identificatore che qualifica il dispositivo, ma solamente il suo ordine di memorizzazione.

Ogni singolo elemento programmabile è descritto nelle sue impostazioni da un blocco di dati che chiameremo **ControlDescriptor** o semplicemente blocco.

Dobbiamo identificare 4 tipologie di formato di memorizzazione:

· **FaderFORMAT** : descrive i 16 cursori e CV1, CV2.

· **ButtonFORMAT** : descrive i 16 pulsanti.
· **DataWheelFORMAT** : descrive i link di Data Wheel.
· **SetupStringFORMAT** : descrive la memorizzazione della Setup String.
Ognuno di questi formati di memorizzazione ha una organizzazione base, comune per tutti, che struttura ogni blocco nel seguente modo:

[1°byte del blocco] = lunghezza delle informazioni seguenti nel blocco, che chiameremo **LengthBYTE**.

Perciò la lunghezza del blocco è data da LengthBYTE+1.

[2°byte del blocco] = memorizza due informazioni:

1) nei 4 bit meno significativi troviamo l'identificatore del tipo di controllo memorizzato, che chiameremo **TypeNIBBLE**.

2) i 4 bit più significativi invece contengono la lunghezza della stringa (da 0 a 15) che descrive il nome del controllo associato a quel dispositivo. Questa stringa verrà in seguito riferita come **CommandNAME** e la sua lunghezza come **NameLENGTH**. La codifica della stringa avviene secondo lo standard ASCII.

Seguono poi gli altri dati, la cui dimensione e significato dipendono dal valore di TypeNIBBLE.

■ **Descrizione di : FaderFORMAT.**

[LengthBYTE] + [NameLENGTH|TypeNIBBLE] + [DATI]

TypeNIBBLE = $00 = No Message
$01 = Controller
$02 = Master Fader
$03 = String

Vediamo il formato del campo [DATI] nei vari casi:

1) NO MESSAGE

Nessun dato. [ES. $01 $00]

LengthBYTE NameLEGNTH=0 + No Message

2) CONTROLLER

[DATI] = Min, Max, CHNL, Num, MODE, [CommandNAME].

byte byte byte byte byte

Min, Max = valori variabili tra 0 e 127 che indicano l'intervallo di variazione del controllo.

CHNL = valore variabile tra 0 e 15 che descrive il numero di canale. Al canale 1 associo il valore CHNL = 0 e così via ...

Num = numero del controller da inviare (0..120).

MODE = $00 : Merge
$01 : Replace
$02 : Update

3) MASTER FADER

[DATI] = LSB, MSB, cvB, [CommandNAME].

| | |

byte byte byte

La rappreeesentazione binaria dell'intero a 16 bit formato dalla giustapposizione: MSB,LSB identifica le selezioni per i Fader da 1 a 16.
Il bit 0 di cvB identifica la selezione di CV1 ed il bit 1 identifica la selezione di CV2.

4) STRING

[DATI] = Formato, Min, Max, String, [CommandNAME].

| | |

byte 2byte 2byte

Formato = $01 per "Single Byte"

String = codifica di una stringa a lunghezza variabile (max 256 caratteri) in formato ASCII, secondo la struttura seguente:

[1°byte] = lunghezza della stringa (n)
[successivi n bytes] = codifica ASCII della stringa.

Nota: Nella codifica della stringa il vaolre $FF identifica il carattere "pr" che dice al Controller Midi di inserirvi il valore attuale della posizione del dispositivo.

■ **Descrizione di : ButtonFORMAT.**

[LengthBYTE] + [NameLENGTH|TypeNIBBLE] + [DATI]

TypeNIBBLE =
$00 = No Message
$01 = Mute Fader
$02 = Solo Fader
$03 = Program Change
$04 = Note ON/OFF
$05 = String
$06 = String Press/Release
$07 = String Toggle

Vediamo il formato del campo [DATI] nei vari casi:

1) NO MESSAGE

Nessun dato. [ES. $01 $00]

LengthBYTE NameLEGNTH=0 + No Message

2) MUTE FADER

[DATI] = [CommandNAME].

Il campo CommandNAME è presente solo se NameLENGTH è diverso da zero.

3) SOLO FADER

[DATI] = [CommandNAME].

Il campo CommandNAME è presente solo se NameLENGTH è diverso da zero.

4) PROGRAM CHANGE

[DATI] = CHNL, Prog.Num., [CommandNAME].
| |
byte byte

5) NOTE ON/OFF

[DATI] = CHNL, Note, Volume, [CommandNAME].
| | |
byte byte byte

Note = byte che esprime il valore di una nota. Associa il valore 0 alla nota Do_{-1} (Do a 16 Hz) ed ogni incremento di una unità aumenta di un semitono il valore della nota.

6) STRING

[DATI] = StringToSend, [CommandNAME].
|
String

7) STRING PRESS/RELEASE

[DATI] = PressString, ReleaseString, [CommandNAME].
| |
String String

8) STRING TOGGLE

[DATI] = ToggleString1, ToggleString2, [CommandNAME].
| |
String String

■ **Descrizione di : DataWheelFORMAT.**

[LengthBYTE] + [TypeNIBBLE] + [ControlLINK]

TypeNIBBLE = $01

ControlLINK = valore che identifica il dispositivo del Controller collegato a Data Wheel.

ControlLINK =
$00 = FADER 1
$01 = FADER 2
$02 = FADER 3
. = .
. = .
. = .
$0F = FADER 16
$10 = CV1
$11 = CV2
$12 = LAST FADER

Per NON avere l'associazione rotellina-dispositivo (corrisponde alla scelta **NONE** nel menu) si specifica un blocco con la struttura di un "No Message":

LengthBYTE = $01 TypeNIBBLE = $00.

■ **Descrizione di : SetupStringFORMAT.**

[LengthBYTE] + [TypeNIBBLE] + [DATI]

Per ogni canale Ch=1..16 posso impostare i valori di:

Bank = off, 0..127.
Prog = off, 0..127.
Vol = off, 0..127.

TypeNIBBLE = $01

[DATI] = LSB, MSB, n*(Bank, Prog, Vol), String.
| | | | |
byte byte byte byte byte

Un intero a 16 bit, formato giustapponendo MSB e LSB, identifica i canali attivi con il bit=1 e disattivi con il bit=0. Il bit 0 identifica il canale 1, ... , ed il bit 15 il canale 16.

n = numero di bit uguali ad 1 nell'intero MSB,LSB. Cioè indica il numero di canali attivi (in cui cioè almeno uno fra Bank, Prog, Vol è diverso da off).

Ogni blocco di 3 bytes (Bank, Prog, Vol) descrive il canale, nell'ordine dal canale 1 al 16.

Metodo per memorizzare i valori 'off', 0..127 :

· 0..127 : i 7 bit meno significativi contengono il numero e l'ottavo bit è posto = 1.
· 'off' : valore $7F (01111111 binario).

IL SOFTWARE

Il programma PC-1600.EXE

Il programma PC-1600.EXE si occupa principalmente di gestire lo scambio di informazioni tra Personal Computer ed il Controller Midi PC-1600. Esso fornisce anche altre funzionalità, quali il salvataggio di dati su file e la possibilità di modificare i valori memorizzati.
Per capire appieno il funzionamento del programma è necessario essere a conoscenza delle strutture che gestisce e della loro organizzazione.

Il Peavey 1600 a fronte di una richiesta SysEx "Require All Presets", invia su Midi-IN tutte le informazioni riguardanti le impostazioni dei suoi 50 Preset. Il formato di queste informazioni è stato da me decodificato ed il programma PC1600.EXE contiene al suo interno tutte le routine di decodifica automatica di questi dati.
I dati sui Preset dopo essere stati ricevuti sequenzialmente, opportunatamente assemblati e codificati, vengono memorizzati in una struttura dati che chiameremo **PresetTable**.
La PresetTable contiene quindi la decodifica dell'intero blocco di dati inviato dal Peavey. Essa può essere vista in termini di struttura dati come un Array di 50 elementi, dove ogni elemento è un descrittore di Preset: **PresetTYPE**. Usando la notazione del Pascal possiamo scrivere:

```
PresetTable : ARRAY[1..50] of PresetTYPE;
```

PC-1600.EXE mantiene in memoria una PresetTable che può essere ricevuta dal Peavey, modificata, salvata e ricaricata da disco e rispedita al Peavey. La PresetTable è quindi la struttura su cui prepariamo le impostazioni che vogliamo memorizzare nel Peavey 1600.
All'inizio del programma la PresetTable viene inizializzata con tutti Preset del tipo "* Preset Vuoto *", i quali contengono per ogni Preset l'associazione a *'No Message'* (nessuna azione).

Per operare con questa struttura abbiamo bisogno di un elemento di appoggio su cui eseguire le operazioni. Questo elemento è denominato **Buffer** e la sua struttura è quella di un descrittore di Preset. Possiamo perciò così formalizzare la definizione di Buffer:

```
Buffer : PresetTYPE;
```

La PresetTable è perciò costituita da 50 descrittori di Preset, che possono essere singolarmente copiati nel Buffer per venire manipolati.
Il programma permette quattro operazioni sul Buffer:

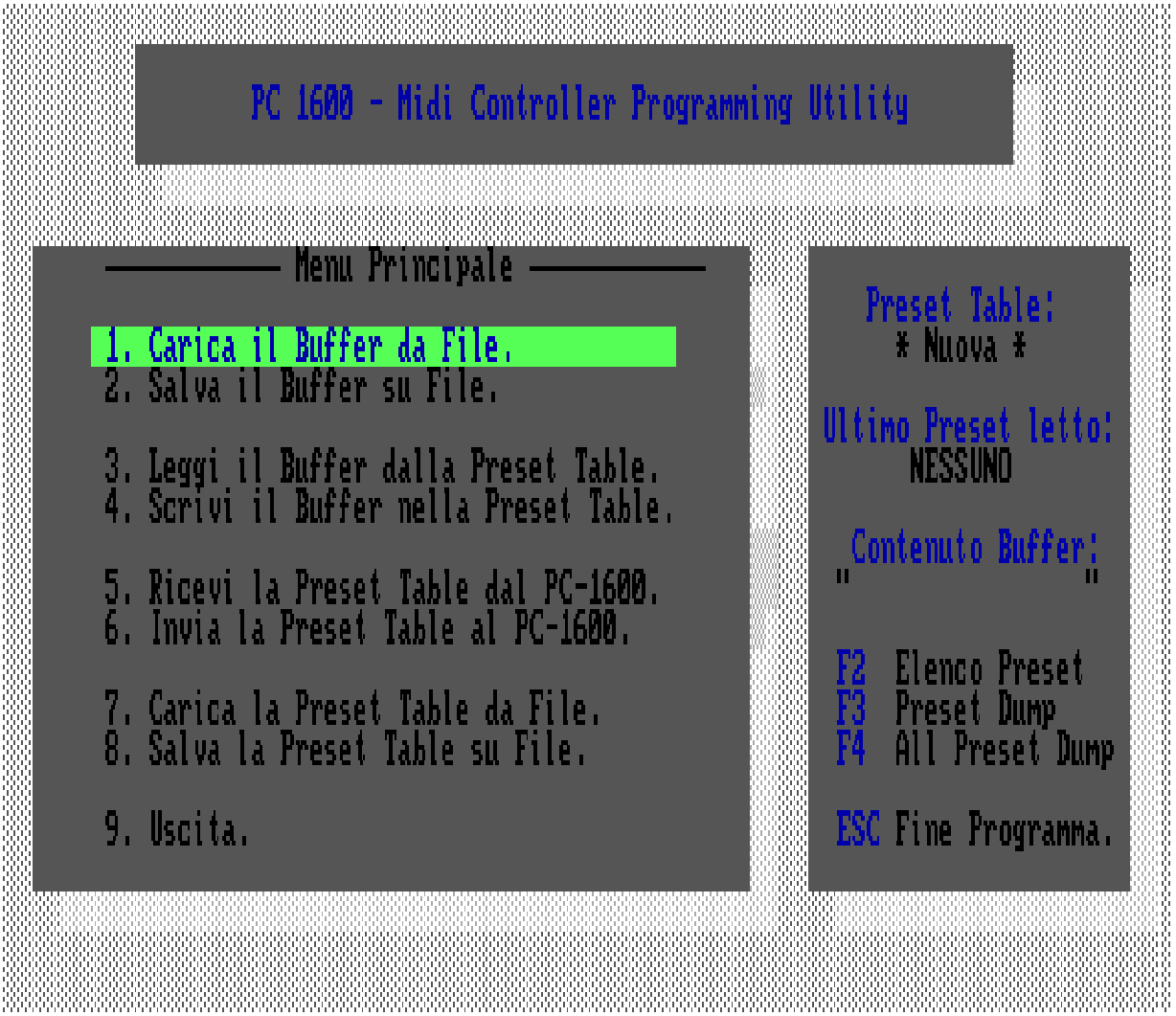
· caricamento del Buffer da file [.PRS]
· salvataggio del Buffer su file [.PRS]
· copia del Buffer in uno dei 50 elementi della PresetTable
· caricamento in uno dei 50 elementi della PresetTable nel Buffer

Queste operazioni, pur essendo molto spartane, permettono una notevole flessibilità e consentono, con un po' di pratica, una grande varietà di usi. Ad esempio si possono salvare su disco le informazioni di un Preset, oppure scambiare il contenuto di due Preset o ancora annullare il contenuto di un Preset. (Per annullare un Preset è sufficiente caricare nel Buffer il "* Preset Vuoto *" precedentemente salvato su disco dalla condizione iniziale, e copiarlo al posto dell'elemento della PresetTable scelto.)

Alle quattro funzionalità precedentemente descritte, il programma ne prevede altre quattro per la gestione della PresetTable. Esse consistono in:

· caricamento da file della PresetTable [.MID]
· salvataggio du file della PresetTable [.MID]
· invio della PresetTable al Peavey 1600
· ricezione della PresetTable dal Peavey 1600
La PresetTable viene memorizzata su disco esattamente nello stesso formato in cui essa viene inviata al controller Midi. Si tratta praticamente di una ridirezione su disco dei dati Midi da inviare; ottengo cioè su disco una copia esatta di ciò che devo inviare al controller. Questo risulta molto utile nel caso in cui si lavori su di

un elaboratore non connesso fisicamente all'interfaccia Midi, premettendo di costruire la propria PresetTable e memorizzarla su disco. Il programma si presenta con la seguente interfaccia utente:



Ciascuno degli 8 possibili comandi è impostabile selezionandolo con le frecce della tastiera e premendo 'Enter'. Sulla destra sono disponibili alcune informazioni sullo stato attuale del sistema. Si tratta rispettivamente di:

· da quale fonte è stata caricata l'ultima PresetTable
· il nome dell'ultimo file .PRS caricato nel Buffer
· il contenuto del Buffer

Sono inoltre disponibili alcune funzioni che allargano le potenzialità del software, da semplice gestore del colloquio tra Pc e Peavey 1600, a vero e proprio strumento di gestione remota dei dati tramite elaboratore.
Premendo il tasto funzine **F2** si ottiene in una finestra l'elenco di tutti i nomi dei Preset memorizzati nella PresetTable. Questo permette di conoscerne facilmente il contenuto e la composizione.
I tasti **F3** ed **F4** rendono disponibili all'utente una sorta di ricostruzione dei dati presenti nella PresetTable. Con la semplice pressione di un tasto il programma ricostruisce le operazioni che sono state eseguite nella programmazione di un determinato Preset del controller e genera la loro rappresentazione sotto forma di comandi specifici per il Peavey. Per la sintassi ed il significato dei comandi si faccia riferimento alla sezione: "Programmazione Diretta del Peavey 1600". L'output è generato sul file PR-TABLE.TXT, data la discreta quantità delle informazioni prodotte.
F3 produce la lista dei comandi associati ad un Preset scelto, mentre F4 produce la lista dei comandi di tutti i Preset contenuti nella PresetTable.

Un esempio di output del comando F3 è:

```
   PRESET N°49    PresetName = "DPM SXII Control"
  ================================================
FADER1 = String(1,0,127,"90 FF 7F ") "Note-On"
FADER2 = NoMesssage
FADER3 = NoMesssage
FADER4 = NoMesssage
FADER5 = NoMesssage
FADER6 = NoMesssage
FADER7 = NoMesssage
FADER8 = NoMesssage
FADER9 = NoMesssage
FADER10 = NoMesssage
FADER11 = NoMesssage
FADER12 = NoMesssage
FADER13 = NoMesssage
FADER14 = String(1,42,1,"F0 00 00 1B 02 03 00 0A FF F7 ") "Threshold"
FADER15 = String(1,0,127,"F0 00 00 1B 02 03 00 1C FF F7 ") "Pre-Trigger"
FADER16 = String(1,0,127,"F0 00 00 1B 02 03 00 0E FF F7 ") "Length"
FADER17 = NoMesssage
FADER18 = NoMesssage
BUTTON1 = String("F0 00 00 1B 02 03 00 06 05 0B 05 F7 ") "12kHz"
BUTTON2 = String("F0 00 00 1B 02 03 00 06 24 68 03 F7 ") "16kHz"
BUTTON3 = String("F0 00 00 1B 02 03 00 06 43 45 02 F7 ") "24kHz"
BUTTON4 = String("F0 00 00 1B 02 03 00 06 12 74 01 F7 ") "32kHz"
BUTTON5 = String("F0 00 00 1B 02 03 00 06 10 4B 01 F7 ") "38.461kHz"
```

```
BUTTON6 = String("F0 00 00 1B 02 03 00 06 14 31 01 F7 ") "44.1kHz"
BUTTON7 = String("F0 00 00 1B 02 03 00 06 61 22 01 F7 ") "48kHz"
BUTTON8 = NoMesssage
BUTTON9 = NoMesssage
BUTTON10 = StringToggle("F0 00 00 1B 02 03 00 1B F7 ","F0 00 00 1B 02 03 00
1A F7 ") "Stereo/Mono"
BUTTON11 = String("F0 00 00 1B 02 03 00 17 F7 ") "Monitor On"
BUTTON12 = String("F0 00 00 1B 02 03 00 16 F7 ") "Monitor Off"
BUTTON13 = String("F0 00 00 1B 02 03 00 18 F7 ") "Audition"
BUTTON14 = String("F0 00 00 1B 02 03 00 01 F7 ") "Arm"
BUTTON15 = String("F0 00 00 1B 02 03 00 02 F7 ") "Start"
BUTTON16 = String("F0 00 00 1B 02 03 00 03 F7 ") "Stop"
DATAWHEEL = ( Control#18 )
SETUPSTRING = NONE
```

Si può constatare come di sia cercato di riprodurre nella maniera più fedele possibile la sintassi per la programmazione diretta del Peavey.

Vediamo ora come avvengono nel dettaglio le operazioni di comunicazione e scambio dati con il Peavey.

L'Implementazione:

Particolare attenzione merita l'implementazione delle operazioni per lo scambio dati con il controller Peavey. Sono stati necessari alcuni accorgimenti per garantire un giusto dialogo fra Pc e controller.
Per salvaguardarci da perdite di dati nella comunicazione è stato necessario predisporre una struttura in memoria centrale sufficientemente grande da contenere interamente tutti i dati inviati dal PC-1600. Questa struttura consiste in un Array di bytes residente nello Heap, che referenzieremo in seguito con **MData**.

MData : ^ARRAY[1..BuffSize] of BYTE;

Nel nostro caso sono necessari un minimo di 42 Kb per contenere tutti i dati; valore che può aumentare vista la natura dinamica del formato.
Poiché i dati letti dall'interfaccia Midi vengono ricavati dalla lettura della porta del microprocessore relativa alla scheda MIF-PC collegata al computer, questa

operazione deve avvenire il più velocemente possibile, pena la perdita di parte dei dati.

Appena è stata inviata la richiesta di spedizione dati, si procede con il trasferire tutti i dati in arrivo nella struttura MData, senza effettuare alcuna elaborazione. Questo consente di ottenere integralmente tutti i dati. Successivamente si eseguono delle elaborazioni sui dati memorizzati in MData per tradurli dal formato di spedizione del Peavey al formato ad 8 bit pieni, necessario per la decodifica delle informazioni.
Innanzitutto si procede con l'eliminare l'intestazione del messaggio SysEx ricevuto, dopo aver controllato che corrisponda al messaggio che volevamo ricevere. Si raggruppano poi a due a due i bytes seguenti, che vengono assemblati per dar luogo ad un singolo byte di informazione; uno per ogni coppia considerata.
Tutte le operazioni avvengono esclusivamente agendo sulla struttura dati MData, per ragioni di risparmio della memoria disponibile. Alla fine di queste elaborazioni avremo su MData tutti i dati convertiti e pronti alla successiva decodifica.
Per la decodifica delle informazioni (e per averne anche un facile accesso) dobbiamo trasferire i dati memorizzati sequenzialmente in MData in un'altra struttura, che ci permette di distinguere ogni singolo Preset ed ogni suo elemento caratterizzato dagli specifici attributi.
Si tratta della PresetTable già descritta in precedenza, dove

PresetTable : ARRAY[1..50] of PresetTYPE;

e PresetTYPE è così definito:

PresetTYPE = ARRAY[1..36] of BlockTYPE;

PresetTYPE contiene 36 elementi uguali (con la stessa struttura) che rappresentano i 36 controlli del Peavey memorizzati in un Preset: 16 Fader + CV1 + CV2 + 16 Button + DataWheel + SetupString; organizzati nel seguente modo:

Elementi da 1 a 16: Fader 1..16,
Elemento 17: CV1,
Elemento 18: CV2,
Elementi da 19 a 34: Button 1..16,
Elemento 35: DataWheel,
Elemento 36: SetupString.

Questa struttura ci permette di modificare ogni singolo attributo di ogni controllo del Preset prescelto con estrema facilità. Ogni elemento di tipo BlockTYPE descrive quindi in maniera generale la definizione di un qualsiasi controllo del Peavey. Ad esso corrisponde la seguente struttura dati:

```
BlockTYPE = Record
    TypeBYTE : BYTE;
    ControlDATA : STRING[80];
    ControlNAME : STRING[15];
   End;
```

· TypeBYTE = contiene un intero che identifica il tipo di funzione assegnata al controllo. Per i possibili valori ci si riferisca alla descrizione del formato dati del PC-1600.

· ControlDATA = si tratta di una stringa che contiene i parametri che caratterizzano una determinata funzione associata (ad Es. la stringa Midi da spedire).

· ControlNAME = stringa di max.15 caratteri che contiene il nome da associare al controllo.

La manipolazione dei dati in questa struttura, è compito di un intero insieme di procedure del programma, le quali contenendo al loro interno le regole di codifica del formato, fungono da interfaccia tra programma e struttura dati. E' sufficiente passar loro, come parametri, i valori da memorizzare e loro si occupano di inserirli nella struttura dati secondo le varie regole della codifica.

E' facile constatare come chiunque abbia una media esperienza di programmazione possa estrapolarle assieme alla struttura dati ad esse associata, per poter gestire personalmente i Preset del Peavey 1600.

```
SetFader2NOMESSAGE(PresetNDX, FaderNDX : BYTE);
SetFader2CONTROLLER(PresetNDX, FaderNDX : BYTE;
  Min,Max,CHNL,Num,MODE : BYTE; Name : STRING);
SetFader2MASTERFADER(PresetNDX, FaderNDX : BYTE; FaderW : WORD;
  cvB : BYTE; Name : STRING);
SetFader2STRING(PresetNDX, FaderNDX : BYTE; Formato:BYTE; Min,
  Max:WORD; St : STRING; Name : STRING);

SetButton2NOMESSAGE(PresetNDX, ButtonNDX : BYTE);
SetButton2MUTEFADER(PresetNDX, ButtonNDX : BYTE; Name : STRING);
SetButton2SOLOFADER(PresetNDX, ButtonNDX : BYTE; Name : STRING);
SetButton2PRGCHANGE(PresetNDX, ButtonNDX : BYTE; CHNL, ProgNum
  :BYTE; Name : STRING);
SetButton2NOTEONOFF(PresetNDX, ButtonNDX : BYTE; CHNL,Note,Vol :
  BYTE; Name : STRING);
SetButton2STRING(PresetNDX, ButtonNDX : BYTE; St : STRING; Name :
  STRING);
SetButton2STRINGPRRE(PresetNDX, ButtonNDX : BYTE; St1, St2 :
  STRING; Name : STRING);
SetButton2STRINGTOGGLE(PresetNDX, ButtonNDX : BYTE; St1,St2 :
  STRING; Name : STRING);

SetDataWheel(PresetNDX : BYTE; Control : BYTE);

SetSetupString(PresetNDX : BYTE; St : STRING);
```

Tutte le operazioni di modifica/aggiornamento dei Preset devono essere eseguite sulla PresetTable per mezzo di queste funzioni. La struttura MData è solo una struttura di appoggio per la conversione dal formato sequenziale inviato dal Peavey a questo formato strutturato.
Una volta che tutte le impostazioni sono state eseguite e si voglia memorizzare queste informazioni all'interno del PC-1600, il programma segue le seguenti fasi:

· A partire dalla PresetTable si costruisce la struttura sequenziale dei dati da inviare e la si memorizza in MData in questo modo:

Partendo dalla prima locazione di MData ed inserendo i dati sequenzialmente:

1. Inserisco i nomi dei 50+1 Preset.

2. Inserisco i 52 valori dei puntatori, posti inizialmente tutti a 0 poiché non se ne conosce ancora il valore.

3. Inserisco i 51 blocchi di descrizione dei Preset e calcolo i valori assunti dai puntatori.

4. Mi riposiziono ed aggiorno i puntatori con il valore precedentemente calcolato.

Nota: I dati sono ancora memorizzati nel formato ad 8 bit pieni. Verranno convertiti nel formato spedizione al momento stesso dell'invio.

. La spedizione avviene inviando su Midi-OUT il messaggio SysEx che notifica l'arrivo dall'esterno dei dati riguardanti i Preset "*Send/Receive All Presets*". Segue al messaggio l'intero contenuto di MData che viene trasformato in formato spedizione al momento stesso dell'invio dei dati. Infine viene spedito il carattere che identifica la fine del messaggio SysEx: F7 hex (End of Exclusive).

Il programma SCRIPT.EXE

Il programma SCRIPT.EXE è un interprete di Script. Cos'è un interprete di Script ? Si tratta di un programma che legge dei comandi contenuti in un file e li elabora, producendo in uscita un file con i risultati dell'elaborazione. Nel nostro caso il file di partenza contenente l'elenco dei comandi da elaborare sarà un file

testo con estensione .MIX. Questo file è formato da una serie di comandi, ognuno dei quali specifica un'associazione:

dispositivo programmabile del Peavey	comando Midi per il Mixer "SE II"

Questo insieme di comandi (file script .MIX) contiene la specificazione di un Preset del Peavey 1600, che verrà opportunatamente interpretato e codificato.
Il risultato delle operazioni viene dato in uscita sotto forma di un descrittore di Preset, in un file .PRS.
Questo è il punto di contatto tra i due programmi: il formato di scambio .PRS. Da notare che solo il programma SCRIPT.EXE contiene i dati specifici per il Mixer "SE II", mentre il PC-1600.EXE risulta indipendente da suddetto mixer e si occupa solo di comunicare dei dati forniti da qualcun'altro al controller Peavey 1600.
Nel caso si volesse programmare il Peavey 1600 per gestire un altro dispositivo Midi, è sufficiente scrivere un altro programma che genera il file corrispondente .PRS. Questo può essere sia un altro interprete di Script, come un qualsiasi altro tipo di programma. E' sufficiente quindi conoscere il formato del file .PRS, di struttura molto più semplice di quella per la comunicazione con il Peavey.

Esempio di utilizzo del Programma:

File di input: FDR-A-16.MIX. Per compilare lo script digito:

```
C:\>SCRIPT FDR-A-16.MIX

 The "Sound Engineer II"  Script Interpreter  v.1.1  - 29/10/96 -  Fabio Parise
--------------------------------------------------------------------------------
  Compiling ....
  Writing data to FDR-A-16.PRS ....
  Compile successful.
```

SCRIPT.EXE genera su disco il file FDR-A-16.PRS, compatibile con il programma per la comunicazione col Peavey: PC-1600.EXE.

REDAZIONE DI FILE COMANDO [.MIX] PER SCRIPT.EXE:

Segue l'insieme delle regole e dei comandi necessari per scrivere file Script [.MIX] interpretabili da SCRIPT.EXE.

· Tutti i comandi possono essere scritti sia con lettere maiuscole che minuscole, fatta eccezione per le stringhe alfanumeriche il cui contenuto è Case Sensitive.
· Le stringhe si specificano fra una coppia di doppi apici. ES. "Volume with Mute".

· E' possibile inserire righe vuote tra i comandi.

· I commenti si specificano inserendo un punto e virgola come primo carattere della riga.
· E' possibile (consigliato) specificare il nome del Preset con una stringa di 16 caratteri. Il comando è:

PRESET_NAME = "* Nome Preset *"

· La stringa specificata dopo il comando rappresenta il nome che si vuole assegnare al rispettivo controllo del Peavey. In caso questa venga omessa il programma provvede ad inserirne una di default (una per ogni comando del Mixer).

I comandi di questo file assegnano ad un controllo del Peavey, un comando del Mixer "SE II". L'assegnazione avviene tramite l'operatore: '='.

Ogni comando segue quindi la seguente struttura:

[Controllo Peavey] = [Comando Mixer]

A [Controllo Peavey] corrispondono i seguenti possibili valori:

FADER1, FADER2, FADER3, FADER4, FADER5, FADER6, FADER7, FADER8, FADER9, FADER10, FADER11, FADER12, FADER13, FADER14, FADER15, FADER16, CV1, CV2,
BUTTON1, BUTTON2, BUTTON3, BUTTON4, BUTTON5, BUTTON6, BUTTON7, BUTTON8, BUTTON9, BUTTON10, BUTTON11, BUTTON12, BUTTON13, BUTTON14, BUTTON15, BUTTON16,
DATAWHEEL,
SETUPSTRING.

Nota: l'operatore di assegnazione '=' deve sempre essere delimitato da almeno uno spazio prima e dopo.

I controlli che non vengono assegnati non assumono alcuna funzione nel Preset generato.

MOLTO IMPORTANTE: se un controllo viene assegnato due volte, ad esso viene associata una stringa Midi corrispondente alla SOMMA dei comandi specificati e non l'ultimo comando incontrato.
Questo permette di conglobare in un controllo più di un comando, fino al limite fisico di 80 bytes totali per la stringa Midi associata.

COMANDI ASSEGNABILI AI FADER:

· Controllo delle funzioni sui 32 Ingressi

Sintassi: **FADER***x* **=** *Comando***(***ch***)** ["*CommandString*"]

x = numero da 1 a 16
ch = numero di canale di ingresso (1..32)
CommandString = stringa di max 15 caratteri con il nome della funzione.

Comando:	**Significato:**
PAN	Permette la regolazione del panorama stereo del segnale (bilanciamento). Per i gruppi PAN bilancia tra i pari e i dispari. Per le mandate AUX non ha effetto.
FADER	Definisce il livello del segnale inviato alle uscite Master ed ai Group.
TREBLE	Equalizzazione delle alte frequenze.
BASS	Equalizzazione delle basse frequenze.
MIDDLE	Equalizzazione delle frequenze medie.
MIDFREQ	Regola la frequenza di centro banda per la regolazione dei medi (Middle).
AUX1_LEVEL	Livello di entrata della mandata Aux1.
AUX2_LEVEL	Livello di entrata della mandata Aux2.
AUX3_LEVEL	Livello di entrata della mandata Aux3.
AUX4_LEVEL	Livello di entrata della mandata Aux4.
GAIN	Definisce il livello di pre-amplificazione del canale.

· **Controllo delle Uscite**

Sintassi: **FADER***x* = *Comando* [**"***CommandString***"**]

x = numero da 1 a 16
CommandString = stringa di max 15 caratteri con il nome della funzione.

Comando:	**Significato:**
MASTER_L	Livello del Volume uscita Left
MASTER_R	Livello del Volume uscita Right
MASTER	Livello dei Volumi uscite Left-Right
MASTER_SUB_L	Livello del Volume uscita Sub_Left
MASTER_SUB_R	Livello del Volume uscita Sub_Right
MASTER_SUB	Livello dei Volumi uscite Sub_LR
MASTER_AUX1	Livello del Volume uscita AUX1
MASTER_AUX2	Livello del Volume uscita AUX2
MASTER_AUX3	Livello del Volume uscita AUX3
MASTER_AUX4	Livello del Volume uscita AUX4
MASTER_GROUP1	Livello del Volume uscita Group1
MASTER_GROUP2	Livello del Volume uscita Group2
MASTER_GROUP3	Livello del Volume uscita Group3
MASTER_GROUP4	" " " " Group4
MASTER_GROUP5	" " " " Group5
MASTER_GROUP6	" " " " Group6
MASTER_GROUP7	" " " " Group7
MASTER_GROUP8	" " " " Group8

COMANDI ASSEGNABILI AI BUTTON:

· Controllo delle funzioni sui 32 Ingressi

Sintassi: **BUTTON***x* **=** *Comando***(***ch***)** [**"***CommandString***"**]

x = numero da 1 a 16
ch = numero di canale di ingresso (1..32)
CommandString = stringa di max 15 caratteri con il nome della funzione.

Comando:	**Significato:**
RESET	Reset del mixer (nessun parametro).
INPUT-A	Seleziona l'ingresso A del canale.
INPUT-B	Seleziona l'ingresso B del canale.
INPUT_SWITCH	Seleziona alternativamente l'ingresso A e l'ingresso B del canale, ad ogni pressione del pulsante.
EQU-ON	Abilita l'equalizzazione del segnale per i comandi TREBLE, MIDDLE, BASS.
EQU-OFF	Disabilita l'equalizzazione del segnale.
EQU-ON/OFF	Alternativamente abilita/disabilita l'equalizzazione del segnale.
MUTE-ON	Abilita il muting del canale.
MUTE-OFF	Disabilita il muting del canale.
MUTE	Alternativamente disabilita/abilita il muting del canale.
SOLO	Soloing del canale. (Non ha effetto sulle mandate AUX.)
LEFT/RIGHT	Assegna al canale l'uscita Master LR.
LEFT/RIGHT-OFF	Leva l'assegnazione dell'uscita Master LR al canale.
GROUP1	Assegna al canale l'uscita Group1.
GROUP1-OFF	Leva l'assegnazione dell'uscita Group1 al canale.
GROUP2	Assegna al canale l'uscita Group2.

GROUP2-OFF	Leva l'assegnazione dell'uscita Group2 al canale.
GROUP3	Assegna al canale l'uscita Group3.
GROUP3-OFF	Leva l'assegnazione dell'uscita Group3 al canale.
GROUP4	Assegna al canale l'uscita Group4.
GROUP4-OFF	Leva l'assegnazione dell'uscita Group4 al canale.
GROUP5	Assegna al canale l'uscita Group5.
GROUP5-OFF	Leva l'assegnazione dell'uscita Group5 al canale.
GROUP6	Assegna al canale l'uscita Group6.
GROUP6-OFF	Leva l'assegnazione dell'uscita Group6 al canale.
GROUP7	Assegna al canale l'uscita Group7.
GROUP7-OFF	Leva l'assegnazione dell'uscita Group7 al canale.
GROUP8	Assegna al canale l'uscita Group8.
GROUP8-OFF	Leva l'assegnazione dell'uscita Group8 al canale.

· **Assegnazione di un valore ad un comando continuo**

Con questo insieme di comandi si vuole imporre un valore iniziale ad un controllo che può assumere un valore continuo. E' indispensabile nel caso si voglia inizializzare un determinato valore di un controllo.

Sintassi: **BUTTON***x* **=** *Comando***(***ch***):***Valore* [**"***CommandString***"**]

x = numero da 1 a 16
ch = numero di canale di ingresso (1..32)
Valore = numero compreso fra 0 e 127
CommandString = stringa di max 15 caratteri con il nome della funzione.

Comando:	**Significato:**
PAN	Permette la regolazione del panorama stereo del segnale (bilanciamento). Per i gruppi PAN bilancia tra i pari e i dispari. Per le mandate AUX non ha effetto.

FADER	Livello di ingresso del canale.
TREBLE	Equalizzazione delle alte frequenze.
BASS	Equalizzazione delle alte frequenze.
MIDDLE	Equalizzazione delle frequenze medie.
MIDFREQ	Regola la frequenza di centro banda per la regolazione dei medi (Middle).
AUX1_LEVEL	Livello di entrata della mandata Aux1.
AUX2_LEVEL	Livello di entrata della mandata Aux2.
AUX3_LEVEL	Livello di entrata della mandata Aux3.
AUX4_LEVEL	Livello di entrata della mandata Aux4.
GAIN	Livello di preamplificazione del canale.

· **Assegnazione di un valore ai volumi di uscita**

Sintassi: **BUTTON***x* = *Comando***:***Valore* [**"***CommandString***"**]

x = numero da 1 a 16
Valore = numero compreso fra 0 e 127
CommandString = stringa di max 15 caratteri con il nome della funzione.

Comando:	**Significato:**
MASTER_L	Livello del Volume uscita Left
MASTER_R	Livello del Volume uscita Rigth
MASTER	Livello dei Volumi uscite Left-Right
MASTER_SUB_L	Livello del Volume uscita Sub_Left
MASTER_SUB_R	Livello del Volume uscita Sub_Right
MASTER_SUB	Livello dei Volumi uscite Sub_LR
MASTER_AUX1	Livello del Volume uscita AUX1

MASTER_AUX2	Livello del Volume uscita AUX2
MASTER_AUX3	Livello del Volume uscita AUX3
MASTER_AUX4	Livello del Volume uscita AUX4
MASTER_GROUP1	Livello del Volume uscita Group1
MASTER_GROUP2	Livello del Volume uscita Group2
MASTER_GROUP3	Livello del Volume uscita Group3
MASTER_GROUP4	Livello del Volume uscita Group4
MASTER_GROUP5	Livello del Volume uscita Group5
MASTER_GROUP6	Livello del Volume uscita Group6
MASTER_GROUP7	Livello del Volume uscita Group7
MASTER_GROUP8	Livello del Volume uscita Group8

COMANDI ASSEGNABILI A DATAWHEEL:

· **Assegnazione di un controllo del PC-1600**

Sintassi: **DATAWHEEL =** *Controllo*

Controllo:	**Significato:**
FADER1	Assegna il Fader1 a DataWheel.
FADER2	Assegna il Fader2 a DataWheel.
FADER3	Assegna il Fader3 a DataWheel.
FADER4	Assegna il Fader4 a DataWheel.
FADER5	Assegna il Fader5 a DataWheel.
FADER6	Assegna il Fader6 a DataWheel.
FADER7	Assegna il Fader7 a DataWheel.
FADER8	Assegna il Fader8 a DataWheel.
FADER9	Assegna il Fader9 a DataWheel.
FADER10	Assegna il Fader10 a DataWheel.
FADER11	Assegna il Fader11 a DataWheel.
FADER12	Assegna il Fader12 a DataWheel.

FADER13	Assegna il Fader13 a DataWheel.
FADER14	Assegna il Fader14 a DataWheel.
FADER15	Assegna il Fader15 a DataWheel.
FADER16	Assegna il Fader16 a DataWheel.
CV1	Assegna l'ingresso CV1 a DataWheel.
CV2	Assegna l'ingresso CV2 a DataWheel.
LASTFADER	Assegna l'ultimo Fader mosso a DataWheel.
NONE	Nessuna assegnazione.

COMANDI ASSEGNABILI ALLA SETUPSTRING:

Tutti i comandi assegnabili ad un BUTTON.

ESEMPIO: IL FILE FDR-A-16.MIX

```
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
; *    Demo MIX file : Controllo dei Volumi dei canali da 1 a 16.   *
; * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *


;              0123456789ABCDEF = 16 caratteri
Preset_Name = "Faders ch: 1..16"

FADER1  = Fader(01) "Fader ch:01"
FADER2  = Fader(02) "Fader ch:02"
FADER3  = Fader(03) "Fader ch:03"
FADER4  = Fader(04) "Fader ch:04"
FADER5  = Fader(05) "Fader ch:05"
FADER6  = Fader(06) "Fader ch:06"
FADER7  = Fader(07) "Fader ch:07"
FADER8  = Fader(08) "Fader ch:08"
FADER9  = Fader(09) "Fader ch:09"
FADER10 = Fader(10) "Fader ch:10"
FADER11 = Fader(11) "Fader ch:11"
FADER12 = Fader(12) "Fader ch:12"
```

```
FADER13 = Fader(13) "Fader ch:13"
FADER14 = Fader(14) "Fader ch:14"
FADER15 = Fader(15) "Fader ch:15"
FADER16 = Fader(16) "Fader ch:16"

BUTTON1  = Mute(01) "Mute + "
BUTTON2  = Mute(02) "Mute + "
BUTTON3  = Mute(03) "Mute + "
BUTTON4  = Mute(04) "Mute + "
BUTTON5  = Mute(05) "Mute + "
BUTTON6  = Mute(06) "Mute + "
BUTTON7  = Mute(07) "Mute + "
BUTTON8  = Mute(08) "Mute + "
BUTTON9  = Mute(09) "Mute + "
BUTTON10 = Mute(10) "Mute + "
BUTTON11 = Mute(11) "Mute + "
BUTTON12 = Mute(12) "Mute + "
BUTTON13 = Mute(13) "Mute + "
BUTTON14 = Mute(14) "Mute + "
BUTTON15 = Mute(15) "Mute + "
BUTTON16 = Mute(16) "Mute + "

BUTTON1  = Input-A(01) "In:A"
BUTTON2  = Input-A(02) "In:A"
BUTTON3  = Input-A(03) "In:A"
BUTTON4  = Input-A(04) "In:A"
BUTTON5  = Input-A(05) "In:A"
BUTTON6  = Input-A(06) "In:A"
BUTTON7  = Input-A(07) "In:A"
BUTTON8  = Input-A(08) "In:A"
BUTTON9  = Input-A(09) "In:A"
BUTTON10 = Input-A(10) "In:A"
BUTTON11 = Input-A(11) "In:A"
BUTTON12 = Input-A(12) "In:A"
BUTTON13 = Input-A(13) "In:A"
BUTTON14 = Input-A(14) "In:A"
BUTTON15 = Input-A(15) "In:A"
BUTTON16 = Input-A(16) "In:A"

DATAWHEEL = LASTFADER

SETUPSTRING = Master_Group1:80
SETUPSTRING = Master_Group2:80
SETUPSTRING = Master_Group3:80
SETUPSTRING = Master_Group4:80
```

LA PROGRAMMAZIONE DIRETTA DEL PEAVEY 1600:

Questa sezione non rientrava inizialmente negli scopi di questo lavoro. E' stata introdotta solamente in seguito per ragioni di completezza del lavoro svolto. Tutto lo studio per la decodifca del formato di memorizzazione dei dati di Peavey 1600 rimane celato nei dettagli implementativi del software, permettendo solamente l'utilizzo di comandi specifici per il Mixer "SE II". L'implementazione dei comandi del Mixer è effetuata all'interno del generatore di script, usando come base una serie di primitive che mettono a disposizione comandi anaolghi a quelli dell'editor del PC-1600. Queste primitive realizzano una sorta di macchina virtuale all'interno del programma, in grado di rappresentare un Preset esattamente come avviene nella memoria del Peavey.
Perchè allora non realizzare un'interfaccia che renda disponibie all'esterno questa macchina virtuale?
Si tratta di rendere disponibili dei comandi aggiuntivi che permettano la chiamata diretta a queste funzioni, essendo queste già presenti nel software.

Questi nuovi comandi danno la possibilità di eseguire tutte le operazioni di impostazione dei controlli del Peavey tramite uno script da Pc. Questo evita di dover introdurre tutte le impostazioni tramite la scomoda interfaccia utente del PC-1600.

L'implementazione di questo insieme di funzioni dà la possibilità a qualsiasi utente di poter programmare il Peavey 1600 per contollare qualsiasi tipo di dispositivo Midi. E' sufficiente impostare i controlli in modo da inviare messaggi System Exclusive compatibili con quel particolare dispositivo.
L'interprete di script diventa quindi un programmatore di Peavey del tutto generale, in grado di comandare una grande varietà di dispositivi Midi, previa la conoscenza dei loro codici specifici di SysEx.

Per quanto riguarda il significato dei parametri, si fa riferimento alla sezione sulla programmazione del Peavey o al manuale per l'utente del PC-1600.

Programmazione dei FADER:

· **NoMessage**

Sintassi: **FADER***x* **= NoMessage**
x = numero da 1 a 16

· **Controller**

Sintassi: **FADER***x* **= Controller(** *Min***,** *Max***,** *CHNL***,** *Num***,** *MODE***,** *CommandNAME***)**

x = numero da 1 a 16
Min = limite inferiore di variabilità del controllo [0..127]
Max = limite superiore di variabilità del controllo [0..127].
CHNL = numero di canale Midi [0..15].
Num = numero del controller da inviare (0..120).
MODE = metodo di filtraggio dei dati entranti [0=Merge, 1=Replace, 2=Update]
CommandNAME = stringa di alfanumerica di max. 15 caratteri indicante il nome del controllo.

· **MasterFader**

Sintassi: **FADER***x* **= MasterFader(** *MSB***,** *LSB***,** *cvB***,** *CommandNAME* **)**
x = numero da 1 a 16
MSB, LSB, cvB = indicano i Fader di qui essere il Master.

· **String**

Sintassi: **FADER***x* **= String(** *Format***,** *Min***,** *Max***,** *HexString***,** *CommandNAME* **)**
x = numero da 1 a 16

HexString = stringa contenente la codifica esadecimale del messaggio Midi da inviare. Ogni byte della stringa Midi deve sempre essere rappresentato da due cifre esadecimali. Zero: 00, non 0. Uno: 01, non 1. Etc...

ES. "F0 00 01 F7"

Programmazione dei BUTTON:

· **NoMessage**

Sintassi: **BUTTON***x* **= NoMessage**

x = numero da 1 a 16

· **MuteFader**

Sintassi: **BUTTON***x* **= MuteFader(** *CommandName* **)**

x = numero da 1 a 16

CommandNAME = stringa di alfanumerica di max. 15 caratteri indicante il nome del controllo.

· **SoloFader**

Sintassi: **BUTTON***x* **= SoloFader(** *CommandName* **)**

x = numero da 1 a 16

CommandNAME = stringa di alfanumerica di max. 15 caratteri indicante il nome del controllo.

· **ProgramChange**

Sintassi: **BUTTON***x* **= ProgramChange(** *CHNL***,** *Num***,** *CommandName* **)**

x = numero da 1 a 16
CHNL = numero di canale Midi [0..15].
Num = numero del Program Change da inviare.
CommandNAME = stringa di alfanumerica di max. 15 caratteri indicante il nome del controllo.

· **NoteON/OFF**

Sintassi: **BUTTON***x* **= NoteON/OFF(** *CHNL***,** *Note***,** *Vol***,** *CommandName* **)**

x = numero da 1 a 16
CHNL = numero di canale Midi [0..15].
Note = valore della nota da inviare (Note=0 => Do_{-1}, ogni incremento di 1 corrisponde ad un semitono).
Vol = volume della nota da suonare [0..127].
CommandNAME = stringa di alfanumerica di max. 15 caratteri indicante il nome del controllo.

· **String**

Sintassi: **BUTTON***x* **= String(** *HexString***,** *CommandName* **)**

HexString = stringa contenente la codifica esadecimale del messaggio Midi da inviare. Ogni byte della stringa Midi deve sempre essere rappresentato da due cifre esadecimali. Zero: 00, non 0. Uno: 01, non 1. Etc...

CommandNAME = stringa di alfanumerica di max. 15 caratteri indicante il nome del controllo.

· **StringPrsRel**

Sintassi: **BUTTON***x* **= StringPrsRel(** *HexStrPrs***,** *HexStrRel***,** *CommandName***)**

HexStrPrs = codifica esadeciamle della PressString [Formato *HexString*].
HexStrRel = codifica esadeciamle della ReleaseString [Formato *HexString*].
CommandNAME = stringa di alfanumerica di max. 15 caratteri indicante il nome del controllo.

· StringToggle

Sintassi: **BUTTON***x* **= StringToggle(** *HexString1***,** *HexString2***,** *CommandName***)**

HexString1 = codifica esadeciamle della prima stringa da inviare [Formato *HexString*].
HexString2 = codifica esadeciamle della seconda stringa [Formato *HexString*].
CommandNAME = stringa di alfanumerica di max. 15 caratteri indicante il nome del controllo.

Programmazione di DATAWHEEL:

Sintassi: **DATAWHEEL =** *Controllo*

Controllo = può assumere uno dei seguenti valori:

FADER1, FADER2, FADER3, FADER4, FADER5, FADER6, FADER7, FADER8, FADER9, FADER10, FADER11, FADER12, FADER13, FADER14, FADER15, FADER16, CV1, CV2, LAST FADER, NONE.

Programmazione della SETUP STRING:

Sintassi: **SETUPSTRING = String(** *HexString* **)**

HexString = stringa contenente la codifica esadecimale del messaggio Midi da inviare. Ogni byte della stringa Midi deve sempre essere rappresentato da due cifre esadecimali. Zero: 00, non 0. Uno: 01, non 1. Etc...

ESEMPIO: IL FILE DIRECT.MIX.

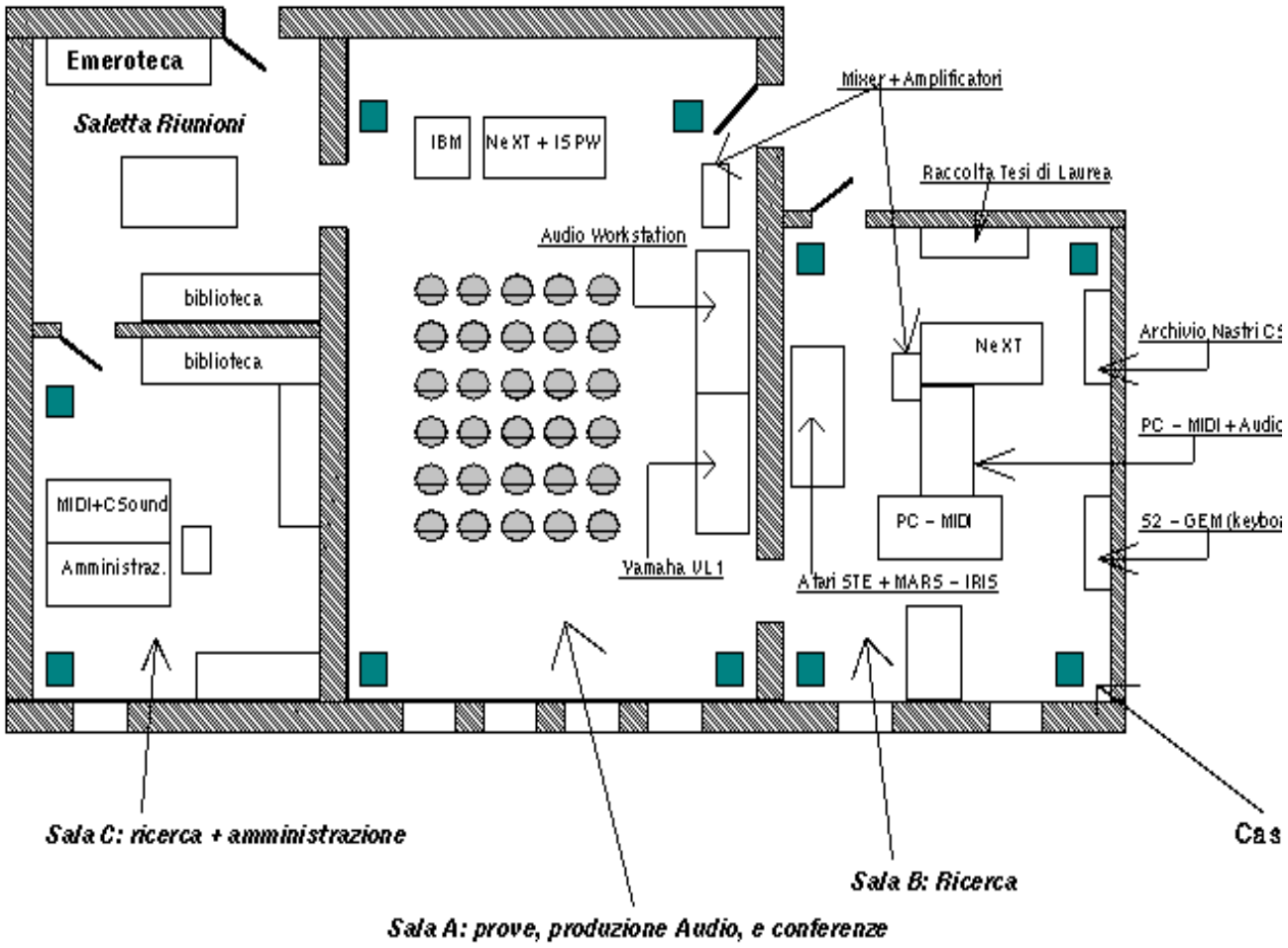
```
;  * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  *    Demo MIX file : Programmazione diretta del Peavey 1600.      *
;  *                                                                 *
;  *      Autore: Parise Fabio               Data: 29/11/96         *
;  * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

;              0123456789ABCDEF = 16 caratteri
Preset_Name = "Prova DirectPrg."

FADER1 = NoMessage
FADER2 = Controller(0,127,0,7,0) "Controller7"
FADER3 = Controller(0,127,0,9,0) "Controller9"
FADER4 = MasterFader(240,0,0) "Master 1-8"
FADER5 = MasterFader(16,0,0) "Master 9-16"
FADER6 = String(1,0,127,"00 01 02 03 04 05")
FADER7 = String(3,0,100,"F0 42 30 24 41 1E FF 00 F7") ""
FADER8 = String(1,0,50,"10 10 10")
FADER9 = String(1,0,100,"F0 A0 F7")

BUTTON1 = NoMessage
BUTTON2 = MuteFader "Mute Button"
BUTTON3 = SoloFader "Solo Button"
BUTTON4 = ProgramChange(0,7) "Volume"
BUTTON5 = ProgramChange(0,8) ""
BUTTON6 = NoteON/OFF(15,60,128) ""
BUTTON7 = NoteON/OFF(15,72,128) ""
BUTTON8 = String("F0 41 36 00 20 30 00 05 03 F7")
BUTTON9 = StringPrsRel("90 60 7F 5F 7F","90 5F 00 60 00") ""
BUTTON10 = StringPrsRel("90 3C 64","80 3C 00") ""
BUTTON11 = StringToggle("F0 41 36 00 20 30 00 05 03 F7","F0 41 36 00 20 30 00
05 04 F7") "SysEx Toggle"
BUTTON12 = StringToggle("F0 18 04 00 03 28 00 01 00 F7","F0 18 04 00 03 28 00
00 00 F7") "SysEx Toggle"

DATAWHEEL = LastFader
```

```
SETUPSTRING = String("F0 00 00 1B 0B 00 10 F7")
```

CONCLUSIONE DEL LAVORO:
Prova pratica delle funzionalità del Software.

A conclusione del lavoro svolto vediamo alcuni esempi di programmazione specifica del Peavey per il controllo del sistema di mixing della Sala Conferenze presente al Centro di Sonologia Computazionale dell'Università di Padova, in via San Francesco 11.

La sala è organizzata come in figura:



Si tratta di realizzare degli appositi files di script per il controllo del mixer "SE II", il quale gestisce le numerose apparecchiature di cui è dotata la sala:

[1] CONTROLLO LIVELLO INGRESSI

Funzionalità generale: Controllo del livello di ingresso dei primi 16 canali del mixer (canale A), con possibilità di muting del canale.

Implementazione: Per questa funzionalità associamo all'x-esimo cursore del controller (FADERx), il controllo del livello di ingresso dell'x-esimo canale entrante.

FADERx = Fader(x) per x = 1..16

Bisogna ora associare al pulsante relativo all'x-esimo cursore l'invio della stringa Midi per il muting del canale:

BUTTONx = Mute(x) per x = 1..16

Alla pressione del pulsante associamo anche la selezione del canale A per l'ingresso x-esimo.

BUTTONx = Input-A(x)

Bisogna però inviare al Mixer tutti i comandi per l'assegnazione degli ingressi ai gruppi di uscita (4 casse), come pure i controlli per il volume di uscita.

L'assegnazione dei gruppi (casse acustiche) può essere effettuata alla pressione del pulsante, perciò a canali dispari (uscita stereo: Left) associo i gruppi 4 ed 1 (cui corrispondono le casse acustiche di sinistra, come si può ben vedere dalla figura). Analogamente ai canali pari (uscita stereo: Right) associo i gruppi 2 e 3 (casse acustiche di destra).
Perciò ogni dispositivo, che ha sempre due canali di uscita adiacenti (per il suono stereofonico) uno pari ed uno dispari, avrà le seguenti impostazioni dei gruppi:

Detto x il canale dispari ed x+1 quello pari corrispondente, avremo per ogni dispositivo:

BUTTONx = Group1(x)
BUTTONx = Group4(x)
BUTTONx+1 = Group2(x+1)
BUTTONx+1 = Group3(x+1)

Il volume delle uscite può essere impostato assegnando i relativi controlli Midi alla SetupString, la quale viene inviata al mixer nel momento in cui il Preset viene selezionato.

Tra i valori di volume compresi fra 0 e 127, assegnamo un livello di volume ai 4 gruppi pari ad 80:

SETUPSTRING = Master_Group1:80
SETUPSTRING = Master_Group2:80
SETUPSTRING = Master_Group3:80
SETUPSTRING = Master_Group4:80

Data Wheel può essere utilizzata per aggiustamenti del tipo "Fine Tuning" del livello dell'ultimo Fader mosso, questo è stato previsto appositamente nel progetto del Peavey 1600 con l'impostazione LASTFADER. Perciò:

DATAWHEEL = LASTFADER

Sono stati realizzati 4 script allo scopo, per controllare i livelli di ingresso di tutti i 64 ingressi del mixer. 32 ingressi per canale A e 32 per il canale B.

FDR-A-16.MIX Ingressi da 1 a 16 del canale A.
FDR-A-32.MIX Ingressi da 17 a 32 del canale A.
FDR-B-16.MIX Ingressi da 1 a 16 del canale B.
FDR-B-32.MIX Ingressi da 17 a 32 del canale B.

CODICE SORGENTE DELLO SCRIPT REALIZZATO

```
;  * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  *    Demo MIX file : Controllo dei Volumi dei canali da 1 a 16.   *
;  *                    canale di ingresso: A                        *
;  * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *


;              0123456789ABCDEF = 16 caratteri
Preset_Name = "Faders InA 1..16"

FADER1  = Fader(01) "Mac#01 Fader"
FADER2  = Fader(02) "Mac#02 Fader"
FADER3  = Fader(03) "Mac#03 Fader"
FADER4  = Fader(04) "Mac#04 Fader"
FADER5  = Fader(05) "Workst.Left"
FADER6  = Fader(06) "Workst.Right"
FADER7  = Fader(07) "TC Left"
FADER8  = Fader(08) "TC Right"
FADER9  = Fader(09) "Next Left"
FADER10 = Fader(10) "Next Right"
FADER11 = Fader(11) "ISPW Left"
FADER12 = Fader(12) "ISPW Right"
FADER13 = Fader(13) "DAT Left"
FADER14 = Fader(14) "DAT Right"
FADER15 = Fader(15) "CD Left"
FADER16 = Fader(16) "CD Right"

BUTTON1  = Mute(01) "Mac"
BUTTON2  = Mute(02) "Mac"
BUTTON3  = Mute(03) "Mac"
BUTTON4  = Mute(04) "Mac"
BUTTON5  = Mute(05) "Workst."
BUTTON6  = Mute(06) "Workst."
BUTTON7  = Mute(07) "TC"
BUTTON8  = Mute(08) "TC"
BUTTON9  = Mute(09) "Next"
BUTTON10 = Mute(10) "Next"
BUTTON11 = Mute(11) "ISPW"
BUTTON12 = Mute(12) "ISPW"
BUTTON13 = Mute(13) "DAT"
BUTTON14 = Mute(14) "DAT"
BUTTON15 = Mute(15) "CD"
BUTTON16 = Mute(16) "CD"

BUTTON1  = Input-A(01) " "
BUTTON2  = Input-A(02) " "
BUTTON3  = Input-A(03) " "
BUTTON4  = Input-A(04) " "
BUTTON5  = Input-A(05) " "
BUTTON6  = Input-A(06) " "
BUTTON7  = Input-A(07) " "
BUTTON8  = Input-A(08) " "
BUTTON9  = Input-A(09) " "
BUTTON10 = Input-A(10) " "
```

```
BUTTON11 = Input-A(11) " "
BUTTON12 = Input-A(12) " "
BUTTON13 = Input-A(13) " "
BUTTON14 = Input-A(14) " "
BUTTON15 = Input-A(15) " "
BUTTON16 = Input-A(16) " "

; Mac (quadrifonico)
BUTTON1  = Group1(01) " "
BUTTON2  = Group2(02) " "
BUTTON3  = Group3(03) " "
BUTTON4  = Group4(04) " "
; Workstation
BUTTON5  = Group1(05) " "
BUTTON5  = Group4(05) " "
BUTTON6  = Group2(06) " "
BUTTON6  = Group3(06) " "
; TC (Transportable Computer)
BUTTON7  = Group1(07) " "
BUTTON7  = Group4(07) " "
BUTTON8  = Group2(08) " "
BUTTON8  = Group3(08) " "
; Next
BUTTON9  = Group1(09) " "
BUTTON9  = Group4(09) " "
BUTTON10 = Group2(10) " "
BUTTON10 = Group3(10) " "
; ISPW
BUTTON11 = Group1(11) " "
BUTTON11 = Group4(11) " "
BUTTON12 = Group2(12) " "
BUTTON12 = Group3(12) " "
; DAT
BUTTON13 = Group1(13) " "
BUTTON13 = Group4(13) " "
BUTTON14 = Group2(14) " "
BUTTON14 = Group3(14) " "
; CD
BUTTON15 = Group1(15) " "
BUTTON15 = Group4(15) " "
BUTTON16 = Group2(16) " "
BUTTON16 = Group3(16) " "

DATAWHEEL = LASTFADER

SETUPSTRING = Master_Group1:80
SETUPSTRING = Master_Group2:80
SETUPSTRING = Master_Group3:80
SETUPSTRING = Master_Group4:80
```

[2] REGISTRAZIONE SU DAT

Funzionalità generale: Impostazione dei cursori per una versatile registrazione su DAT, con controllo del volume di registrazione, del volume di ascolto e possibilità di mixing.

Implementazione: Consideriamo il caso specifico in cui si voglia suonare una melodia con RealPiano PRO1 della GEM su di una base musicale eseguita da PC. Si vuole poter ascoltare il tutto regolando il mixaggio tra le due sorgenti ed il volume finale d'ascolto. Deve esserci la possibilità di registrare l'uscita del mixer su DAT, regolando eventualmente il livello di ingresso per la registrazione.

I canali del mixer che ci interessano sono i seguenti:

- Canale A, ingressi 7 e 8: il PC che esegue la base.
- Canale A, ingressi 31 e 32: RealPiano PRO1 della GEM.
- Canale A, ingressi 13 e 14: uscita del DAT.
- L'uscita LeftRight del mixer va in ingresso al DAT.

Perciò:

- il livello degli ingressi 7 ed 8 controlla la base musicale.
- il livello degli ingressi 31 e 32 controlla la melodia suonata con RealPiano.
- l'uscita LeftRight (Master) controlla il volume di ingresso al DAT (e quindi il volume di registrazione).
- l'uscita del DAT entra nel mixer nei canali 13 e 14. Il livello di questi canali determina l'effettivo volume di uscita ai gruppi (cioè il volume d'ascolto finale).
- il volume dei gruppi viene mantenuto sempre costante (pari ad 80 in una scala da 0 a 127).

Associamo un cursore del Peavey al livello di ingresso di ciascun dispositivo interessato, perciò avremo un cursore che controlla entrambi i canali del dispositivo: Left e Right.

Al pulsante corrispondente associamo la funzione di muting del canale, l'assegnazione del canale A come ingresso, LeftRight come uscita ed il giusto bilanciamento del segnale.

Associamo al TC (Trasportable Computer) il Fader #1:

```
; Trasportable Computer
FADER1 = Fader(07) "TC Level"
FADER1 = Fader(08) " "

BUTTON1 = Mute(07) "TC"
BUTTON1 = Mute(08)
BUTTON1 = Input-A(07)
BUTTON1 = Input-A(08)
BUTTON1 = LeftRight(07)
BUTTON1 = LeftRight(08)
BUTTON1 = Pan(07):0
BUTTON1 = Pan(08):127
```

Associamo al RealPiano PRO1 il Fader #2:

```
; RealPiano PRO1
FADER2 = Fader(31) "RealPiano L."
FADER2 = Fader(32) " "

BUTTON2 = Mute(31) "RealPiano"
BUTTON2 = Mute(32)
BUTTON2 = Input-A(31)
BUTTON2 = Input-A(32)
BUTTON2 = LeftRight(31)
BUTTON2 = LeftRight(32)
BUTTON2 = Pan(31):0
BUTTON2 = Pan(32):127
```

Agli altri Fader è possibile associare altri dispositivi della Sala Conferenze, quali il Next (canali 9 e 10) oppure la Workstation (canali 5 e 6), inserendo parti di codice analoghe alle precedenti.

Gli ultimi due Fader vengono utilizzati per regolare i volumi di ingresso al DAT e quello di uscita del DAT.

```
; Volume di uscita
FADER15 = Master "Master LR"

; Volume di registrazione
FADER16 = Fader(13) "DAT rec Vol"
FADER16 = Fader(14) " "

BUTTON16 = Mute(13) "DAT rec"
BUTTON16 = Mute(14)
BUTTON16 = Input-A(13)
BUTTON16 = Input-A(14)
BUTTON16 = LeftRight(13)
BUTTON16 = LeftRight(14)
BUTTON16 = Pan(13):0
BUTTON16 = Pan(14):127
```

Rimagono le istruzioni per l'associazione dei gruppi e l'inizializzazione del volume dei gruppi, che possono venir demandate alla SetupString.

CODICE SORGENTE DELLO SCRIPT REALIZZATO

```
;  * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *
;  *    Demo MIX file : Registrazione su DAT al C.S.C.                     *
;  *                                                                       *
;  *    Autore: Fabio Parise                Data: 29/11/96                 *
;  * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * *

;              0123456789ABCDEF = 16 caratteri
Preset_Name = "DAT recording..."

; Transportable Computer
FADER1 = Fader(07) "TC Level"
FADER1 = Fader(08) " "

BUTTON1 = Mute(07) "TC"
BUTTON1 = Mute(08)
BUTTON1 = Input-A(07)
BUTTON1 = Input-A(08)
BUTTON1 = LeftRight(07)
BUTTON1 = LeftRight(08)
BUTTON1 = Pan(07):0
BUTTON1 = Pan(08):127

; RealPiano PRO1
FADER2 = Fader(31) "RealPiano L."
FADER2 = Fader(32) " "

BUTTON2 = Mute(31) "RealPiano"
BUTTON2 = Mute(32) " "
BUTTON2 = Input-A(31) " "
BUTTON2 = Input-A(32) " "
BUTTON2 = LeftRight(31) " "
BUTTON2 = LeftRight(32) " "
BUTTON2 = Pan(31):0 " "
BUTTON2 = Pan(32):127 " "

; Volume di uscita
FADER15 = Master "Master LR"

; Volume di registrazione
FADER16 = Fader(13) "DAT rec Vol"
FADER16 = Fader(14) " "

BUTTON16 = Mute(13) "DAT rec"
BUTTON16 = Mute(14) " "
BUTTON16 = Input-A(13) " "
BUTTON16 = Input-A(14) " "
BUTTON16 = LeftRight(13) " "
BUTTON16 = LeftRight(14) " "
BUTTON16 = Pan(13):0
BUTTON16 = Pan(14):127

; Workstation
FADER3  = Fader(05) "Workst.Vol"
```

```
FADER3  = Fader(06) " "

BUTTON3 = Mute(05) "Workst."
BUTTON3 = Mute(06) ""
BUTTON3 = LeftRight(05) ""
BUTTON3 = LeftRight(06) ""
BUTTON3 = Pan(05):0 ""
BUTTON3 = Pan(06):127 ""

; Next
FADER4  = Fader(09) "Next Vol."
FADER4  = Fader(10) " "
BUTTON4 = Mute(09) "Next"
BUTTON4 = Mute(10) " "
BUTTON4 = LeftRight(09) " "
BUTTON4 = LeftRight(10) " "
BUTTON4 = Pan(09):0 " "
BUTTON4 = Pan(10):127 " "


DATAWHEEL = LASTFADER

SETUPSTRING = Group1(07)
SETUPSTRING = Group4(07)
SETUPSTRING = Group2(08)
SETUPSTRING = Group3(08)
SETUPSTRING = Master_Group1:80
SETUPSTRING = Master_Group2:80
SETUPSTRING = Master_Group3:80
SETUPSTRING = Master_Group4:80
```

FASI NELLA REALIZZAZIONE DEL LAVORO

· Realizzazione dei seguenti programmi (scaffolding) per la comunicazione con le apparecchiature e l'analisi dei dati:

IO401.PAS Libreria Assembly 80x86 per lo scambio dati con il processore Roland MPU-401 (Midi Processor Unit).

READMIDI.EXE Programma per lo scambio di dati MIDI da e verso l'esterno, attraverso la Roland MPU-401. Produce un file ASCII contenente il dump esadecimale dei dati trasmessi. [.OUT]

MPURESET.EXE Resetta la MPU-401. Utilissimo.

OUT2MIDI.EXE Trasforma l'output di READMIDI (dump esadecimale) in un file binario Midi, esattamente come era stato trasmesso. [.MID]

OUT2ASC.EXE Trasforma l'output di READMIDI in un file ASCII decodificando il formato di spedizione del PEAVEY 1600. [.ASC]

MIDI2OUT.EXE Trasforma un file binario di comandi Midi, in un file ASCII contenente il suo dump esadecimale.

· Reverse Engineering del formato di memorizzazione dei dati del PEAVEY 1600, in base all'analisi dei dump esadecimali prodotti dal programma READMIDI.EXE.

· Realizzazione della documentazione sul formato del dump.

· Realizzazione del Software di codifica/decodifica dei dati spediti dal PEAVEY 1600. (PC-1600.EXE)

· Realizzaione dell'interprete di Script che produce un Preset per il PC-1600, contenente i comandi Midi specificati nel sorgente per pilotare il Mixer 'SE II'. (SCRIPT.EXE ver.1.0)

· Redazione del manuale utente per l'utilizzo dell'interprete di script: specifica del formato dati per i files .MIX.

· Realizzazione di tutta la documentazione sul software prodotto e sul lavoro eseguito.

· Aggiunta di nuove funzionalità all'interprete di script: Programmazione diretta del Peavey 1600. (Ver.1.1 di SCRIPT.EXE)

· Debugging di PC-1600.EXE ed aggiunta delle funzionalità di ricostruzione dei comandi in memoria.

· Redazione di script [.MIX] d'esempio per pilotare lo specifico sistema di mixing digitale del Centro di Sonologia Computazinale dell'Università di Padova.

· Produzione e redazione di tutta la documentazione.

· Prova su strada del software.

BIBLIOGRAFIA / RIFERIMENTI

[1] "C Programming for Midi", Jim Conger, M&T Books

[2] "Peavey PC-1600 Midi Controller USER'S GUIDE"

[3] Manuale per l'utente del mixer "SE II"

[4] Manuale di programmazione Midi di "SE II"

[5] Dispensa del corso di 'Sistemi di Elaborazione per la Musica' sullo Standard Midi.

[6] Peavey Electronics Corporation, Web Page:

URL: http://www.peavey.com

E-Mail: peavey@peavey.com

[7] Guida all’uso di “SE II” tramite il Peavey PC-1600, C.S.C.

IL CONTENUTO DEL DISCHETTO:

File:	Contenuto:
READMIDI.EXE	Legge ed invia dati sulla porta Midi.
READMIDI.PAS	Codice sorgente.
MPURESET.EXE	Resetta la MPU-401. (porta Midi)
MPURESET.PAS	Codice sorgente.
IO401 .ASM	Primitive di I/O con MPU-401.
IO401 .PAS	Codice sorgente.
IO401 .TPU	Libreria compilata.
MIDI2OUT.EXE	Converte dati binari (Midi) nella loro rappresentaz. esadecimale.
MIDI2OUT.PAS	Codice sorgente.
OUT2ASC .EXE	Converte la rappr. esadec. dei dati decodificando il formato di spedizione
OUT2ASC .PAS	dati del Peavey e rappresentando il risultato in ASCII.
OUT2MIDI.EXE	Converte la rappr. esadecimale nel rispettivo file binario.
OUT2MIDI.PAS	Codice sorgente.
PC-1600 .EXE	Programma di comunicazione con Peavey PC-1600
PC-1600 .PAS	Codice sorgente.
SCRIPT .EXE	Interprete di Script (ver 1.1) per programmare Peavey con controlli per "SE II".
SCRIPT .PAS	Codice sorgente.
DAT-REC .MIX	Sorgente Script: registrazione su DAT.
DAT-REC .PRS	Script compilato.
FDR-A-16.MIX	Sorgente Script: regolazione livello ingressi 1..16, canale A.
FDR-A-16.PRS	Script compilato.
FDR-A-32.MIX	Sorgente Script: regolazione livello ingressi 17..32, canale A.
FDR-A-32.PRS	Script compilato.
FDR-B-16.MIX	Sorgente Script: regolazione livello ingressi 1..16, canale B.

FDR-B-16.PRS	Script compilato.
FDR-B-32.MIX	Sorgente Script: regolazione livello ingressi 17..32, canale B.
FDR-B-32.PRS	Script compilato.
DIRECT .MIX	Sorgente Script: programmazione diretta del Peavey.
DIRECT .PRS	Script compilato.
INITIAL .MID	Preset Table: impostazioni fornite dalla casa al Peavey PC-1600.
MIDI-IN .MID	Preset Table: simula Midi-IN du disco.
MIDI-OUT.MID	Preset Table: simula Midi-OUT su disco.
NULL .MID	Preset Table: tutti i Preset annullati.
SE-II .MID	Preset Table: programmazione del Mixer "SE II".
VUOTO .PRS	Preset Vuoto.
PR-TABLE.TXT	Esempio di ricostruzione di un Preset eseguita da PC-1600.EXE.
TESINA .DOC	Testo della tesina in formato WinWord 7.0.
TITOLO .DOC	Copertina della tesina in formato WinWord 7.0
INDICE .DOC	Indice della tesina in formato WinWord 7.0.
README .TXT	Questo file.
REPORT7 .MID	Esempi di dati su qui si è lavorato per decodificare il formato
REPORT7 .OUT	dei dati inviati dal Peavey PC-1600.
REPORT7 .ASC	
BIOS .TPU	Libreria per la realizzazione dei programmi.
VIDEO .TPU	Libreria per la realizzazione dei programmi.
FRAMES .TPU	Libreria per la realizzazione dei programmi.
SMART_IO.TPU	Libreria per la realizzazione dei programmi.
OUTFORMC.TPU	Libreria per la realizzazione dei programmi.