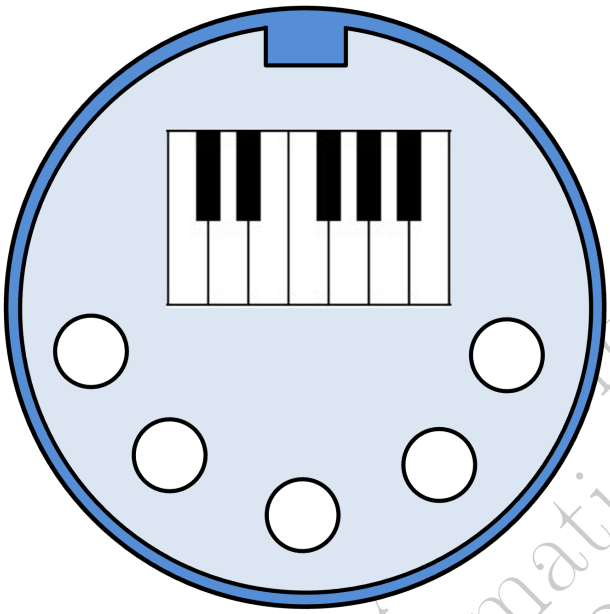
Capitolo 3

MIDI: Standard file e protocollo



In questo capitolo parleremo di Standard MIDI file e del protocollo MIDI. L'acronimo MIDI significa "Musical Instrument Digital Interface". Agli inizi degli anni '80 gli strumenti digitali erano ormai d'uso comune e nasceva l'esigenza di farli comunicare; spesso molti sintetizzatori venivano connessi fra di loro e quindi c'era l'esigenza di coordinarne l'esecuzione. Lo standard MIDI nasce proprio per questo motivo e le maggiori case produttrici di strumenti digitali hanno partecipato alla definizione dello standard. Nel 1984 nasce la "MIDI Manufacturers Association" che si occupa della pubblicazione delle specifiche MIDI. Lo standard MIDI è costituito da 3 componenti: l'interfaccia hardware, il protocollo, e il formato dei file.

3.1 Interfaccia MIDI

Lo standard MIDI stabilisce che il trasferimento dei dati avvenga alla velocità di 31.25 bps usando un collegamento seriale asincrono. Quindi i bit vengono spediti uno di seguito all'altro (trasmissione seriale) e il momento in cui vengono spediti viene deciso dal dispositivo che trasmette (trasmissione asincrona).

I connettori storicamente utilizzati per i dati MIDI sono connettori elettrici di tipo DIN, il cui formato è stato standardizzato dall'istituto tedesco per la standardizzazione (Deutsches Institut

für Normung). Un cavo DIN usato per trasmettere dati MIDI non deve superare la lunghezza di 6.6 metri in quanto con cavi più lunghi si potrebbero avere problemi di trasmissione dovuti all'attenuazione del segnale elettrico e di conseguenza anche a interferenze esterne. I connettori DIN prevedono 5 pin, di cui solo 3 vengono utilizzati per il protocollo MIDI.

Figura 3.1: Connettore e cavo MIDI

Gli strumenti digitali MIDI prevedono 3 porte fisiche: In, Out e Thru. La porta In serve a ricevere dati MIDI dall'esterno, la porta Out serve a spedire verso l'esterno i dati MIDI prodotti dallo strumento digitale, mentre la porta Thru replica in uscita i dati ricevuti sulla porta In senza l'intervento dello strumento stesso. Attraverso la porta Thru è possibile connettere più strumenti MIDI in cascata.

La figura 3.1 mostra le porte In, Out e Thru di un'interfaccia MIDI, un cavo MIDI maschio-maschio e i dettagli del connettore: i PIN 1 e 3 non sono utilizzati, il 4 porta la corrente, il 2 è collegato alla massa mentre il 5 trasporta i dati MIDI.

Il collegamento con cavi MIDI DIN, tuttavia, non viene più utilizzato in quanto i moderni computer riescono a comunicare molto velocemente con interfacce e cablaggi più recenti, come, ad esempio, lo standard USB: I dati MIDI vengono spediti tramite il cavo USB.

3.2 Protocollo MIDI

Come un qualsiasi altro protocollo, il protocollo MIDI è un insieme di regole che permette la comunicazione fra due strumenti MIDI. Le regole del protocollo sono ovviamente specifiche per gli strumenti digitali e riguardano la nota da suonare, lo strumento da usare e tante altre informazioni riguardanti la riproduzione elettronica del suono.

Gli strumenti MIDI sono sintetizzatori elettronici capaci di produrre suoni. Il protocollo MIDI nasce dalla necessità di far comunicare sintetizzatori elettronici commercializzati da diverse case produttrici. Quindi il protocollo MIDI si pone come linguaggio universale per i sintetizzatori elettronici.

Quali sono le informazioni di cui ha bisogno un sintetizzatore per produrre i suoni? Nel seguito daremo una descrizione di tali informazioni

3.2.1 Le note

Il protocollo MIDI rappresenta le note con i numeri interi da 0 a 127. Quindi in totale possiamo rappresentare 128 note diverse. Considerando che un pianoforte ha 88 tasti e quindi 88 note ed è lo strumento con l'estensione più grande, i 128 codici MIDI per le note sono più che sufficienti. La tabella 3.3 riporta i codici MIDI e le note corrispondenti. È uso comune indicare l'ottava di riferimento di una nota usando un pedice; ad esempio $\texttt{C}_1$ o $\texttt{Do}_1$ denota il `Do` della prima ottava del pianoforte, $\texttt{C}_2$ o $\texttt{Do}_2$ il `Do` della seconda ottava e così via. Il $\texttt{Do}_4$ che corrisponde al codice MIDI 60 è il `Do` centrale del pianoforte. Poichè la rappresentazione MIDI permette di distinguere 128 note, andiamo al di fuori dell'estensione di un pianoforte che tipicamente va dalla nota $\texttt{A}_0$, codice MIDI 21, alla nota $\texttt{C}_8$, codice MIDI 108.

Ottava	Codici MIDI											
	C	C#	D	D#	C	F	F#	G	G#	A	A#	B
-1	0	1	2	3	4	5	6	7	8	9	10	11
0	12	13	14	15	16	17	18	19	20	21	22	23
1	24	25	26	27	28	29	30	31	32	33	34	35
2	36	37	38	39	40	41	42	43	44	45	46	47
3	48	49	50	51	52	53	54	55	56	57	58	59
4	60	61	62	63	64	65	66	67	68	69	70	71
5	72	73	74	75	76	77	78	79	80	81	82	83
6	84	85	82	87	88	89	90	91	92	93	94	95
7	96	97	98	99	100	101	102	103	104	105	106	107
8	108	109	110	111	112	113	114	115	116	117	118	119
9	120	121	122	123	124	125	126	127				

Tabella 3.1: Codici MIDI. I tasti del pianoforte corrispondono ai codici dal 21 al 108.

3.2.2 Valore di attacco e di rilascio

Il codice MIDI di una nota individua la nota, cioè la frequenza del suono. In funzione dello strumento scelto oltre a specificare quale nota suonare, che, ad esempio, per un pianoforte significherebbe quale tasto premere mentre per un violino significherebbe in quale punto di quale corda applicare la pressione, è necessario specificare altri parametri che influenzano il suono prodotto. Ad esempio, se si preme un tasto di un pianoforte il suono sarà più forte e corposo se il tasto è premuto con forza e rapidità, mentre sarà più debole e raccolto se il tasto viene premuto con delicatezza. Sul violino dipende da come si usa l'arco, sul flauto da come si soffia. Per specificare il modo in cui si "attacca" il suono, il protocollo MIDI prevede un parametro che specifica la velocità di attacco. Anche in questo caso i valori possibili sono da 0 a 127.

Con funzione analoga (simmetrica) c'è il valore di rilascio che indica in che modo viene "terminata" una nota. Per alcuni strumenti, come ad esmepio il pianoforte, tale valore non ha nessun effetto. Per altri, come ad esempio un flauto, determina il modo in cui viene terminata una nota: un valore alto indica una terminazione brusca, un valore basso indica una terminazione più graduale. La reale implementazione spetta alla perifierica MIDI che riceve il messaggio.

3.2.3 Strumenti

I sintetizzatori posso produrre suoni di vario tipo "imitando" il timbro degli strumenti tradizionali. Ad esempio, possiamo suonare delle note MIDI con il timbro di un flauto o di un violino. Anche in questo caso il protocollo MIDI prevede 128 possibilità dando di fatto la possibilità di utilizzare contemporaneamente fino a 128 timbri diversi. La Tabella 3.2 riporta alcuni codici MIDI per i vari timbri. Nel gergo MIDI si utilizza la parola inglese *patch* per indicare un particolare timbro e l'espressione *program change* per indicare un cambiamento di timbro (quindi, in questa accezione *program* è sinonimo di *patch*).

3.2.4 Canali

Le informazioni MIDI vengono raggruppate in canali. Il protocollo MIDI prevede l'utilizzo di 16 canali. I canali costituiscono un mezzo per separare informazioni diverse fra loro da un punto di vista logico. Ad esempio, a ogni canale può essere associato un particolare strumento. Tutti i comandi che producono un suono relativo a un particolare canale utilizzeranno come timbro quello

Codice	Descrizione (inglese)	Descrizione (italiano)
1	Acoustic Grand	Pianoforte classico
3	Electric Grand	Pianoforte elettrico
7	Harpischord	Clavicembalo
19	Rock organ	Organo Rock
20	Church organ	Organo da chiesa (a canne)
23	Harmonica	Armonica
24	Accordion	Fisarmonica
33	Acustic bass	Basso acustico
34	Electric bass	Basso elettrico
41	Violin	Violino
42	Viola	Viola
43	Cello	Violoncello
44	Contrabass	Contrabbasso
49	String Ensemble 1	Sezione archi 1
50	String Ensemble 2	Sezione archi 2
53	Choir Aahs	Coro (aah)
54	Choir Oohs	Coro (ooh)
67	Tenor Sax	Sassofono tenore
72	Clarinet	Clarinetto
74	Flute	Flauto

Tabella 3.2: Alcuni timbri MIDI

associato al canale. Dunque di fatto il numero di timbri diversi utilizzabili contemporaneamente si riduce a 16 in quanto tanti sono i canali. I canali possono essere usati anche per fare in modo che alcuni strumenti MIDI rispondano solo a messaggi inviati su determinati canali.

3.2.5 Suonare una nota

Per suonare una nota occorre spedire a uno strumento MIDI un messaggio che indica allo strumento di suonare la nota e successivamente spedire un altro messaggio che indica allo strumento di smettere di suonare quella nota. Per un pianoforte questi due messaggi corrispondono alla pressione del particolare tasto che produce la nota e al rilascio del tasto precedentemente premuto. Parleremo più in dettaglio di questi e di altri messaggi MIDI nel seguito. Per il momento ci limitiamo a dire che esiste un messaggio di *Note On* che serve a far suonare la nota. Nel messaggio, oltre al canale di appartenenza, deve essere specificato il codice MIDI della nota da suonare e la velocità di attacco. In modo simile il messaggio di *Note Off* serve a dire allo strumento che lo riceve di smettere di suonare una particolare nota. Come il messaggio di Note On, anche per quello di Note Off occorre specificare il canale MIDI, la nota da non suonare più e la velocità di rilascio.

3.2.6 Informazioni sul tempo

Il protocollo MIDI è stato progettato per l'utilizzo in tempo reale. I messaggi relativi alle note non hanno una indicazione riguardo a "quando" l'azione deve essere eseguita. Il motivo per cui questa indicazione manca è che non serve: l'azione viene eseguita nel momento in cui arriva il messaggio.

Tuttavia alcune periferiche MIDI che registrano i dati MIDI, come ad esempio un sequencer, hanno bisogno di informazioni sul tempo per poter riprodurre successivamente le note. Affinchè tali periferiche possano eseguire la musica MIDI con il giusto tempo il protocollo MIDI prevede dei messaggi di Time Clock per la sincronizzazione. Una periferica MIDI master spedisce a intervalli regolari dei messaggi di Time Clock. Vengono spediti 24 messaggi di Time Clock per ogni semiminima. La durata di ogni semiminima dipende dal BPM (Beat Per Minute). Una semiminima corrisponde a un beat. Ad esempio, con un BPM=120 abbiamo 120 semiminime in un minuto, quindi ogni semiminima dura 0.5 secondi, o equivalentemente in ogni secondo ci sono 2 semiminime. Pertanto in questo caso il master MIDI spedisce 48 messaggi di Time Clock in ogni secondo. Le periferiche MIDI che ricevono questi messaggi possono sincronizzarsi per poter memorizzare i messaggi MIDI e successivamente suonare alla velocità corretta.

Per capire facciamo un esempio: consideriamo il semplice caso di due messaggi MIDI uno di Note On e l'altro di Note Off per la stessa nota.

Se la periferica MIDI che riceve tali messaggi è un sintetizzatore di suoni, deve semplicemente eseguire i comandi e li eseguirà nel momento in cui i messaggi arrivano. Assumiamo che il messaggio di Note On arrivi al tempo 0 sec e che il messaggio di Note Off arrivi al tempo 1 sec. La nota verrà suonata per 1 secondo dal tempo 0 sec al tempo 1 sec. Abbiamo completamente ignorato i messaggi di Time Clock.

Se invece la periferica MIDI che riceve i messaggi è un sequencer che deve memorizzare i messaggi per poi poterli suonari successivamente, allora diventano fondamentali i messaggi di Time Clock. Poichè dal momento in cui riceve il messaggio di Note On al momento in cui riceve il messaggio di Note Off il sequencer riceve 48 messaggi di Time Clock (assumiamo BPM=120) il sequencer sa che quando dovrà suonare la nota dovrà farla durare 48 Time Clock. La velocità di esecuzione potrà essere variata variando la velocità di spedizione dei messaggi di Time Clock.

3.3 File MIDI

I file MIDI servono per memorizzare dei dati MIDI in modo tale da poterli riutilizzare successivamente. Ovviamente, come per i sequencer, anche per i file MIDI occorre aggiungere delle informazioni relative ai tempi di esecuzione. In questa sezione descriveremo in dettaglio il fomrato dei file MIDI.

3.3.1 Big-Endian

I valori che vengono rappresentati con più byte vengono spediti/memorizzati nel formato Big-Endian, cioè partendo dal byte più significativo e proseguendo con i byte meno significativi.

3.3.2 Lunghezza variabile

I file MIDI utilizzano per alcune informazioni dei campi a lunghezza variabile. Un valore rappresentato con un campo a lunghezza variabile può essere rappresentato da uno o più byte. In ognuno dei byte utilizzati per la rappresentazione a lunghezza variabile il bit più significativo non fa parte dei dati ma viene utilizzato come segnalatore: se tale bit è 1 allora il prossimo byte fa ancora parte del dato a lunghezza variabile, se è 0 allora la rappresentazione termina con il byte in questione. Chiaramente solo 7 bit per ogni byte della rappresentazione possono essere sfruttati per i dati. Quindi se il valore da rappresentare è compreso fra 0 e 127, basterà un solo byte in cui il bit più significativo (che normalmente vale 128, ma in questo caso è solo un bit di segnalazione) vale 0. Per rappresentare 128 non basterà un singolo byte perché abbiamo bisogno di 8 bit di dati.

Ad esempio, consideriamo la rappresentazione a lunghezza variabile fornita dai seguenti byte:

Byte	Descrizione	Valore o range	Commento
1-4	Stringa di identificazione	"MThd" (0x4d546864)	costante
5-8	Grandezza del blocco	6 (0x00000006)	costante
9-10	Formato del file	0-2	utilizza solo 2 bit
11-12	Numero di tracce	1-255	utilizza solo 1 byte
13-14	PPQN o SMPTE	PPQN: 1-4096	PPQN utilizza solo 10 bit

Tabella 3.3: Blocco di intestazione

10000001**0**0000000

Leggendo il primo byte **1**0000001 vediamo che il bit più significativo è 1, il che significa che ci sarà almeno un altro byte da considerare. Gli altri 7 bit sono parte dei dati e sono i bit più significativi. Leggendo il secondo byte **0**0000000 vediamo che il bit più significativo è 0 e qiondi questo è l'ultimo byte della rappresentazione. Gli altri 7 bit di questo byte sono parte dei dati. Pertanto la rappresentazione binaria del dato, ignorando gli 0 alla sinistra dell'1 più significativo, è

10000000

che corrisponde a 128. Con 2 byte è possibile rappresentare valori di 14 bit, quindi fino a $2^{14} - 1$=16383. Per il valore successivo, 16384, occorrono 3 byte. Poiché con la rappresentazione a lunghezza variabile sfruttiamo per i dati solo 7 bit di ogni byte con k byte possiamo rappresentare i valori da 0 a $2^{7k} - 1$.

Nel protocollo MIDI i dati a lunghezza variabile possono usare al massimo 4 byte. Pertanto il valore massimo rappresentabile è $2^{28} - 1 = 268435455$ che in binario è rappresentato da un stringa di 28 bit pari a 1, in esadecimale da 0x0F FF FF FF, mentre nella rappresentazione a lunghezza variabile è

11111111 **1**1111111 **1**1111111 **0**1111111.

3.3.3 Struttura generale

Un file midi standard (SMF, Standard Midi File) è formato da una sequenza di blocchi, di cui il primo è un blocco di intestazione (header chunk) e quelli successivi sono dei blocchi di traccia (track-chunk). Quindi la struttura generale è:

```
SMF = <header-chunk> + <track-chunk> [+ <track-chunk> ...]
```

3.3.4 Header chunk (Blocco di intestazione)

Il blocco di intestazione (header chunk) contiene informazioni generali relative al brano musicale memorizzato nel file. Tali informazioni riguardono il formato del file MIDI, il numero di tracce e informazioni relative al tempo. Un file MIDI prevede un solo blocco di intestazione che deve essere inserito all'inizio del file.

```
header-chunk = "MThd" + <lunghezza> + <formato> + <tracce> + <tempo>
```

La struttura del blocco di intestazione è riportata nella seguente tabella.

Il blocco di intestazione (e quindi il file MIDI stesso) inizia sempre con la stringa di 4 caratteri "MThd". I successivi 4 byte contengono la lunghezza del blocco che per quello di intestazione è sempre 6; ciò significa che seguiranno altri 6 byte che fanno parte del blocco di intestazione.

I primi 2 di questi 6 byte specificano il formato del file MIDI. Esistono 3 formati identificati dai numeri 0, 1 e 2. Quindi dei due byte riservati per il formato vengono (almeno attualmente) utilizzati solo 2 bit. Nel formato 0 è prevista una sola traccia che contiene tutti gli eventi MIDI per l'intero brano musicale. Nel formato 1 ci sono due o più tracce; nella prima ci sono informazioni generali quali il titolo del brano, il tempo e altro, mentre nelle tracce successive ci sono dati relativi alle singole tracce, inclusi gli eventi MIDI relativi alle note da suonare. Il formato 2 prevede come per il formato 1 più tracce ma mentre nel formato 1 le tracce sono considerate simultanee nel formato 2 ogni traccia è indipendente dalle altre e la riproduzione della tracce non avviene necessariamente in modo simultaneo.

I successivi due byte specificano il numero di tracce. Ovviamente per i file in formato 0 tale numero è 1. Anche se sono previsti due byte, quindi si possono specificare fino a 65535 tracce, il numero massimo di tracce in un file MIDI è 255. In pratica il primo di questi due byte è sempre 0.

Gli ultimi 2 byte del blocco di intestazione specificano il numero di parti per semiminima (PPQN, Part Per Quarter Note) oppure il numero di frame per secondo. Il primo dei 16 bit indica se i successivi 15 bit specificano il PPQN (ticks).

Il PPQN specifica di quanti *tick* è fatta un semiminima. Si noti che questa indicazione temporale è relativa alla durata di una semiminima. La durata di una semiminima è stabilita dal BPM che può essere specificato attraverso un apposito meta-evento. Il BPM di default è pari a 120, e in questo caso una semiminima dura $0.5s = 500.000\mu s$. Stabilita la durata di una semiminima si può calcolare la durata di un tick dividendo la durata della semiminima per il PPQN. Ad esempio, per un BPM=120, se PPQN=100, allora ogni tick dura $5.000\mu s$.

3.3.5 Blocchi di traccia (track chunk)

Dopo il bloco di intestazione ci sono i blocchi di traccia. Un blocco di traccia contiene tutte le informazioni relative a un traccia.

```
track-chunk = "MTrk" + <lunghezza> + <evento> [+ <evento> ...]
```

Un blocco di traccia inizia con la stringa di 4 caratteri "MTrk" (0x4D54726B), seguita da altri 4 byte che specificano il numero totale di byte, a partire dal byte successivo, che fanno parte della traccia.

Gli eventi vengono usati per descrivere tutte le informazioni relative alla traccia a partire dagli eventi che descrivono le note per arrivare a tanti altri tipi di eventi (come ad esempio il nome della traccia o le parole del testo associato alla musica). Ogni evento è preceduto da un *delta time* che specifica dopo quanto tempo, rispetto all'evento precedente, l'attuale evento deve accadere. Il delta time viene specificato in ticks con un campo a lunghezza variabile. Se il delta time vale 0 vuol dire che l'evento che segue il delta time deve essere eseguito insieme all'evento precedente. Ci sono alcuni eventi per i quali il delta time non ha senso (ad esempio il nome della traccia). Poichè tutti gli eventi devono essere preceduti da un delta time il delta time esiste sempre. Per convenzione tutti gli eventi che non hanno un ordine temporale rispetto algi altri eventi vengono inseriti all'inizio del blocco di traccia ed usano un delta time pari a 0. Si rammenti che il delta time è specificato in ticks la cui durata dipende dal BPM e dal PPQN.

Il formato generale di un evento è il seguente:

```
evento = <delta-time> + <status byte> + (<dati-MIDI> | <dati-SYS> | <dati-META>)
```

Valore binario	Valore esadecimale	Valore decimale	Categoria	Messaggio
0xxx xxxx	0x00-0x7F	0-127	Running status	
1000 xxxx	0x80-0x8F	128-143	MIDI Voice message	Note Off
1001 xxxx	0x90-0x9F	144-159		Note On
1010 xxxx	0xA0-0xAF	160-175		Note Aftertouch
1011 xxxx	0xB0-0xBF	176-191		Controller
1100 xxxx	0xC0-0xCF	192-207		Program Change
1101 xxxx	0xD0-0xDF	208-223		Channel Aftertouch
1110 xxxx	0xE0-0xEF	224-239		Pitch Bend
1111 0000	0xF0	240	SYS messages	Inizio SysEx
1111 0001	0xF1	241		*undefined*
1111 0010	0xF2	242		Song position (LSB)
1111 0011	0xF3	243		Song position (MSB)
1111 0100	0xF4	244		Song select
1111 0101	0xF5	245		*undefined*
1111 0110	0xF6	246		*undefined*
1111 0111	0xF7	247		Fine SysEx
1111 1000	0xF8	248	Real Time	Timing Clock
1111 1110	0xF9	249		*undefined*
1111 1110	0xFA	250		Start
1111 1110	0xFB	251		Continue
1111 1110	0xFC	252		Stop
1111 1110	0xFD	253		*undefined*
1111 1110	0xFE	254		Active Sensing
1111 1111	0xFF	255	META evento	

Tabella 3.4: Singificato del byte di stato

Dopo il delta time c'è sempre un byte chiamato *byte di stato* (in realtà per risparmiare spazio spesso si usa un trucchetto per omettere questo byte quando il suo valore è lo stesso del precedente evento – descriveremo in seguito questo trucchetto noto con il nome di *running status*). Gli eventi sono di 3 tipi: eventi MIDI, eventi di sistema e meta eventi. Il byte di stato permette di capire il tipo di evento, come specificato Tabella 3.4.

Eventi MIDI

Gli eventi MIDI sono quelli che contengono le informazioni "musicali", come, ad esempio quelle relative all'esecuzione di una particolare nota. Un singolo evento MIDI consiste di

```
<dati-MIDI> = <parametro1> [+ <parametro2>]
```

Il primo elemento di un evento MIDI è lo status byte che specifica il tipo di messagio MIDI (con i 4 bit più significativi) e il canale (con i 4 bit meno significativi). Dopo lo status byte possono essere presenti uno o due parametri in funzione del tipo di messaggio MIDI. Gli eventi MIDI costituiscono la maggior parte di un file MIDI. Ci sono 7 tipi di eventi MIDI. La Tabella 3.5 descrive questi eventi specificando anche il significato dei 2 parametri. In alcuni casi viene usato solo il primo parametro (quando il secondo parametro non è usato esso non è presente nel file).

Descrizione	Status byte		Parametro 1	Parametro 2
	4bit (MSbits)	4bit (LSbits)	1 byte (0-127) usa solo 7 bit (LSbits)	1 byte (0-127) usa solo 7 bit (LSbits)
Note Off	0x8	canale (0-15)	codice MIDI nota	velocità
Note On	0x9	canale (0-15)	codice MIDI nota	velocità
Note Aftertouch	0xA	canale (0-15)	codice MIDI nota	volore di aftertouch
Controller	0xB	canale (0-15)	controller number	controller value
Program Change	0xC	canale (0-15)	program number	non usato
Channel Aftertouch	0xD	canale (0-15)	valore di aftertouch	non usato
Pitch Bend	0xE	canale (0-15)	pitch value (LSB)	pitch value (MSB)

Tabella 3.5: Eventi MIDI

Note Off. L'evento Note Off viene usato per rappresentare la fine di una particolare nota. Il codice MIDI della nota specifica a quale nota si riferisce l'evento, mentre il secondo parametro specifica la velocità con cui viene rilasciato la nota. In molti casi la velocità di rilascio non è molto importante (ad esempio per il pianoforte), ma in alcuni casi può essere utile. Il reale effetto dipende da come la periferica MIDI interpreta questo parametro.

Note On. L'evento Note On viene usato per rappresentare l'inizio di una particolare nota. Il codice MIDI specifica la nota da suonare mentre il secondo parametro rappresenta la velocità di attacco.

Aftertouch. Questo evento viene utilizzato per specificare un cambio di pressione relativo a una nota in esecuzione. Il primo parametro specifica la nota, il secondo la pressione (0 = nessuna pressione, 127, massima pressione).

Controller. Questo evento permette di specificare un cambiamento di stato relativo a un canale MIDI. É possibile specificare fino a 128 tipi di controller utilizzando il primo parametro. Il secondo parametro specifica il valore del controller. Un esempio di controller è il "Foot controller" che permette di specificare l'eventuale pressione di un pedale. La Tabella 3.6 mostra i controller definiti dal proocollo MIDI.

Program Change. L'evento Program Change permette di cambiare lo strumento MIDI da utilizzare per la riproduzione dei suoni. Questo evento ha bisogno solo del primo parametro che specifica lo strumento MIDI.

Channel Aftertouch. Questo evento è simile all'evento Aftertouch, con la differenza che opera su tutte le note del canale MIDI. Ha bisogno di un solo parametro che specifica la pressione da applicare a tutte le note che sono attive sul canale MIDI specificato.

Pitch Bend. Questo evento è utilizzato per specificare una eventuale modifica dell'altezza della nota. In realtà questo evento è molto simile a un controller con la differenza che il tipo di controller è intrinseco (è appunto il controller dell'altezza della nota) e il valore viene specificato con 14 bit (mentre un normale controller può usare solo 7 bit per il valore). Il valore del Pitch Bend è specificato dai 14 byte yyyyyyyxxxxxxx dove i caratteri y sono i 7 bit meno significativi del secondo parametro ed i caratteri x sono i 7 bit più significativi del primo parametro. Pertanto i possibili valori per il Pitch Bend sono 0-16383. L'evento ha effetto su tutte le note attive relative al canale MIDI specificato dall'evento. Il valore 8192 è il punto centrale che rappresenta l'altezza reale della nota. Valori al di sotto di 8192 fanno decrescere l'altezza della nota, mentre valori al di sopra fanno crescere l'altezza della nota. L'intervallo di riferimento varia in funzione dello strumento selezionato, ma tipicamente è di 1 tono sia a scendere che a salire. Quindi un Pitch

Valore	Descrizione del controller
0 (0x00)	Bank Select
1 (0x01)	Modulation
2 (0x02)	Breath Controller
4 (0x04)	Foot Controller
5 (0x05)	Portamento Time
6 (0x06)	Data Entry (MSB)
7 (0x07)	Main Volume
8 (0x08)	Balance
10 (0x0A)	Pan
11 (0x0B)	Expression Controller
12 (0x0C)	Effect Control 1
13 (0x0D)	Effect Control 2
16-19 (0x10-0x13)	General-Purpose Controllers 1-4
32-63 (0x20-0x3F)	LSB for controllers 0-31
64 (0x40)	Damper pedal (sustain)
65 (0x41)	Portamento
66 (0x42)	Sostenuto
67 (0x43)	Soft Pedal
68 (0x44)	Legato Footswitch
69 (0x45)	Hold 2
70 (0x46)	Sound Controller 1 (default: Timber Variation)
71 (0x47)	Sound Controller 2 (default: Timber/Harmonic Content)
72 (0x48)	Sound Controller 3 (default: Release Time)
73 (0x49)	Sound Controller 4 (default: Attack Time)
74-79 (0x4A-0x4F)	Sound Controller 6-10
80-83 (0x50-0x53)	General-Purpose Controllers 5-8
84 (0x54)	Portamento Control
91 (0x5B)	Effects 1 Depth (formerly External Effects Depth)
92 (0x5C)	Effects 2 Depth (formerly Tremolo Depth)
93 (0x5D)	Effects 3 Depth (formerly Chorus Depth)
94 (0x5E)	Effects 4 Depth (formerly Celeste Detune)
95 (0x5F)	Effects 5 Depth (formerly Phaser Depth)
96 (0x60)	Data Increment
97 (0x61)	Data Decrement
98 (0x62)	Non-Registered Parameter Number (LSB)
99 (0x63)	Non-Registered Parameter Number (MSB)
100 (0x64)	Registered Parameter Number (LSB)
101 (0x65)	Registered Parameter Number (MSB)
121-127 (0x79-0x7F)	Mode Messages

Tabella 3.6: MIDI Controllers

Bend di 0 farà scendere tutte le note attive di un tono, mentre un Pitch Bend di 16383 le farà salire di un tono.

Running status

Il protocollo MIDI prevede la possibilità, per gli eventi MIDI, (non per gli eventi di sistema ed i metaeventi, né tantomeno per i messaggi real-time) di non spedire il byte di stato nel caso in cui il valore di questo byte sia lo stesso di quello dell'evento precedente. Questa situazione viene segnalata da uno status byte minore o uguale a 127, cioè uno status byte con l'MSB pari a 0. Infatti i valori dello status byte che hanno significato in termini di eventi sono quelli con valori maggiori o uguali a 128, cioè con l'MSB pari a 1. Pertanto se il valore dello status byte è minore di 128 allora lo status byte è fittizio e in realtà il suo valore verrà usato come valore del primo parametro dell'evento. Il valore reale dello status byte in questo caso è uguale a quello dell'evento precedente. Quindi di fatto lo status byte c'è sempre, solo che con questo trucco riusciamo a codificare con un solo byte sia lo satus byte (che è uguale a quello dell'evento precedente) sia il valore del primo parametro.

Per rendere ancora più chiaro il trucco facciamo un esempio. Supponiamo di avere la seguente sequenza di 3 messaggi MIDI:

```
0x00 0x93 0x3C 0x7F 0x00 0x93 0x40 0x7F 0x00 0x93 0x43 0x7F
```

Questi 3 messaggi, che appaiono in sequenza nel flusso di dati midi, hanno come status byte lo stesso valore (0x90) che corrispondente al messaggio MIDI di Note On sul canale 3. Questa sequenza di 3 messaggi MIDI necessita di 9 byte. Usando il trucco del running status è possibile ridurre la lunghezza della sequenza a 7 byte, semplicemente omettendo gli status byte del secondo e del terzo messaggio. Infatti il valore del primo parametro (come pure quello del secondo) è sempre minore di 128. Pertanto la sequenza di byte

```
0x00 0x93 0x3C 0x7F 0x00 0x40 0x7F 0x00 0x43 0x7F
```

è equivalente alla precendente ma occupa solo 7 byte.

Il running status può essere utilizzato solo per i messaggi MIDI (status byte da 0x80 a 0xEF). I messaggi di tipo real time (status byte da 0xF8 a 0xFF) non hanno nessun effetto sul running status (cioè, è come se non fossero stati trasmessi). Si noti che 0xFF è in realtà il codice per i meta eventi.

I messaggi di sistema, invece, cancellano il running status, cioè dopo un messaggio di sistema è obbligatorio spedire il byte di stato.

Il trucco del Note Off

Omettere lo status byte è possibile solo quando esso è uguale a quello precedente. La maggiore parte dei messaggi MIDI è costituita messaggi di Note On e Note Off che si alternano o si ripetono in funzione della musica che codificano. Quando 2 o più messaggi dello stesso tipo (sia di Note On che di Note Off) si ripetono si può usare il running status. Esiste un trucchetto che permette di evitare completamente i messaggi di Note Off ed usare solo messaggi di Note On. Il trucchetto consiste nel sostituire i messaggi di Note Off con messaggi di Note On la cui velocità di attacco (secondo parametro) è 0. Di fatto un tale messaggio fa si che la nota venga "spenta" in quanto viene suonata con un volume pari a 0. Usando questo trucco per tutti i messaggi di Note Off si ha che la sequenza di messaggi MIDI diventi una sequenza di sole messaggi di Note On. Questo permette di utilizzare per tutti i messaggi il running status.

Tipo	Descrizione
0x00	Sequence number
0x01	Text event
0x02	Copyright notice
0x03	Sequence or track name
0x04	Instrument name
0x05	Lyric text
0x06	Marker text
0x07	Cue point
0x20	MIDI channel prefix assignment
0x2F	End of track
0x51	Tempo setting
0x54	SMPTE offset
0x58	Time signature
0x59	Key signature
0x7F	Sequencer specific event

Tabella 3.7: Alcuni meta-eventi

Meta Eventi

I meta-eventi servono a inserire informazioni addizionali non prettamente collegate alla musica codificata dagli eventi MIDI, come ad esempio il nome della traccia. Essi hanno la sguente forma:

```
dati-META = 255 + <tipo> + <lunghezza> + [dati-evento]
```

Quindi un meta evento inizia sempre con un byte che vale 255, seguito da altri 2 byte ed eventualmente da ulteriori byte di dati, il cui numero e significato dipende dallo specifico meta-evento. Il secondo byte specifica il tipo di meta evento. La lunghezza dell'evento è un campo a dimensione variabile e specifica quanti byte fanno parte dell'evento a partire da quello successivo. Se il meta-evento non prevede dati la lunghezza è 0.

La Tabella 3.7 riporta alcuni meta eventi. A titolo di esempio descriviamo in dettaglio alcuni di questi meta-eventi.

Text. I meta-eventi 0x01-0x07, hanno tutti la stessa struttura in quanto permettono di inserire informazioni testuali. La struttura è `FF c n` b_1 b_2 ... b_n, dove `c` è il codice del meta-evento — ad esempio 5 per il testo della canzone — `n` è il numero di caratteri di cui è costituito il testo ed i successivi `n` byte sono i caratteri del testo in codifica ASCII, anche se tutti i valori da 0 a 255 sono ammessi. La differenza è solo la categorizzazione del tipo di informazione contenuta nel testo. Il meta evento Text Event è per un testo generico (es. dei commenti). Il meta evento Copyright notice è per le informazioni sui diritti d'autore. Il meta evento Sequence or track name è per specificare un nome della traccia. Il meta evento Instrument name è per specificare il nome dello strumento utilizzato. Il meta evento Lyrics è per le parole della canzone. Il meta evento Marker text è per indicare delle specifiche posizioni (es. l'inizio di un verso o del ritornello).

Il meta-evento MIDI Channel Prefix serve a specificare un canale MIDI solitamente in riferimento ai successivi meta-eventi; ad esempio prima di un Instrument Name Event in modo da associare lil nome dello strumento al canale MIDI. Il formato è `255 (0xFF) 32 (0x20) 1 c`, dove `c` è un valore da 0 a 15.

Il meta event End of track serve a segnalare la fine di una traccia MIDI. Tale evento deve sempre essere l'ultimo evento di una traccia MIDI. Il formato è `255 47 0`.

Tempo (Set Tempo). Questo meta evento permette di cambiare il tempo specificando un nuovo valore del BPM. Il BPM viene specificato indicando il numero di microsecondi per semiminima. Quindi se si vuole un BPM di 120 il numero di microsecondi per semiminima è 500.000 Questo meta evento prevede 3 byte di dati. Il formato è `255 51 3` b_1 b_2 b_3, con b_1 b_2 b_3 che rappresentano il numero di microsencodi. Ad esempio, il meta-evento `FF 51 03 07 A1 20` specifica un BPM di 120 in quanto `07 A1 20` codificano il valore 500.000. La specifica del valore del BPM viene di solito messa nel primo blocco di traccia, spesso allineata temporalmente con un messaggio di MIDI clock. Se non viene specificato si assume un valore standard di 500.000μs (120 BPM).

Time Signature. Questo meta evento specifica la metrica del brano. Il formato è `FF 88 4` b_1 b_2 b_3 b_4. Il primo byte b_1 è il "numeratore", il secondo byte b_2 rappresenta il "denumeratore", il terzo b_3 le pulsazioni del metronomo e, infine, il quarto B_4 un numero di biscrome. Il numeratore viene specificato in modo diretto (valore letterale), mentre il denominatore si ottiene elevando 2 a b_2 e indica la durata: ad esempio, se b_2=1, il denominatore (che vale 2) è una minima, se b_2, il denominatore (che vale 4) indica una semiminima. Le pulsazioni b_3 specificano il numero di messaggi MIDI clock per clik (il protocollo MIDI prevede 24 click per semiminima). Infine b_4 specifica l'unità di tempo in termini di numero di biscrome; solitamente questo valore è 8 perchè l'unità di tempo è spesso la semiminima che equivale a 8 biscrome.

Key Signature. Questo meta evento specifica la tonalità. Il formato è `255 89 2` b_1 b_2 con b_1 che specifica il numero di alterazioni (da -7 a +7) e b_2 che specifica il modo (0 maggiore, 1 minore).

3.4 References

La documentazione ufficiale dello standard MIDI può essere consultata, previa registrazione gratuita, nel sito `https://www.midi.org/specifications`.

Esercizi

1. Come vengono rappresentate le note con il protocollo MIDI? E i diversi timbri? Quante note e quanti timbri diversi possono essere codificati?
2. Quali sono i messaggi MIDI che permettono di iniziare a suonare una nota e di terminare il suono?
3. Si consideri il seguente valore binario in notazione a lunghezza variabile: 10000010 (primo byte) 000000010 (secondo byte). Quale valore (decimale) codifica?
4. Con quali byte inizia un file MIDI? E con quali byte inizia una traccia all'interno di un file MIDI?
5. Che cosa è il delta time in un file MIDI? In che unità viene specificato?
6. Che cosa è lo status byte nel protocollo MIDI? Si fornisca una spiegazione dettagliata.
7. Quanti parametri prevedono i vari messaggi MIDI. Si faccia qualche esempio.
8. Si spieghi come funziona il metodo detto *running status*.
9. In cosa consiste il "trucco" del Note Off per i file MIDI?
10. Si consideri i sequenti di 4 eventi MIDI

    (a) 0x01 0x93 0x40 0x7F
    (b) 0x02 0x93 0x41 0x7F
    (c) 0x03 0x93 0x42 0x7F
    (d) 0x04 0x93 0x43 0x7F

    Si scriva la relativa sequenza di byte sfruttando il running status, spiegando come la si è ottenuta. (Si ricordi che il formato di un evento MIDI prevede il delta-time, il byte di stato, e uno o due parametri in funzione dell'evento.)
11. Si faccia qualche esempio di meta-evento MIDI fornendo dettagli.