The MIDI File Format

MIDI files contain one or more MIDI streams, with time information for each event. Song, sequence, and track structures, tempo and time signature information, are all supported. Track names and other descriptive information may be stored with the MIDI data. This format supports multiple tracks and multiple sequences so that if the user of a program which supports multiple tracks intends to move a file to another one, this format can allow that to happen.

The MIDI files are block oriented files. Currently only 2 block types are defined, header and track data. As opposed to the IFF and RIFF formats, no global header is given, so that the validation must be done by adding the different block sizes.

A MIDI file always starts with a **header block**, and is followed by one or more **track blocks.**

The format of the header block

OFFSET	*Count*	*TYPE*	*Description*
0000h	4	char	ID='MThd'
0004h	1	dword	Length of header data (=6)
0008h	1	word	Format specification: 0: one, single multi-channel track 1: one or more simultaneous tracks 2: one or more sequentially independent single-track patterns
000Ah	1	word	Number of track blocks in the file
000Ch	1	int	Unit of delta-time values: If negative: Absolute of high byte = number of frames per second. Low byte = resolution within one frame If positive: division of a quarter-note.

The track data format

The **MTrk block type** is where actual song data is stored. It is simply a stream of “events” (MIDI and non-MIDI), preceded by delta-time values.

Some numbers in **MTrk blocks** are represented in a form called a **variable-length quantity**. These numbers are represented 7 bits per byte, most significant bits first. All bytes except the last have bit 7 set, and the last byte has bit 7 clear. If the number is between 0 and 127, it is thus represented exactly as one byte. Since this explanation might not be too clear, some examples :

Number (hex)	Representation (hex)
00000000	00
00000040	40
0000007F	7F
00000080	81 00
00002000	C0 00
00003FFF	FF 7F
001FFFFF	FF FF 7F
08000000	C0 80 80 00
0FFFFFFF	FF FF FF 7F

The largest number which is allowed is 0FFFFFFF so that the variable-length representation must fit in 32 bits in a routine to write variable-length numbers.

Each track block contains one or more MIDI events, each event consists of a delta-time and the number of the event. The delta-time is stored as a variable-length quantity and represents the time to delay before the following event. A delta-time of 0 means, that the event occurs simultaneous with the previous event or occurs right at the start of a track. The delta-time unit is specified in the header block.

The format of the track information block

OFFSET	*Count*	*TYPE*	*Description*
0000h	4	char	ID='MTrk'
0004h	1	dword	Length of header data
0008h	?	rec	<delta-time>, <event>

Three types of events are defined, **MIDI event**, **meta event and system exclusive event**. The first event in a file must specify status; delta-time itself is not an event.

Meta Events

Meta events are non-MIDI information.

The format of the meta event

OFFSET	*Count*	*TYPE*	*Description*
0000h	1	byte	ID=FFh
0001h	1	byte	Type (<=128)
0002h	?	?	Length of the data, 0 if no data stored as variable length quantity
	?	byte	Data

A few meta-events are defined. It is not required for every program to support every meta-event. Meta-events initially defined include:

FF 00 02 ssss Sequence Number
This optional event, which must occur at the beginning of a track before any nonzero delta-times, and before any transmittable MIDI events, specifies the number of a sequence.

FF 01 len text Text Event
Any amount of text describing anything. It is a good idea to put a text event right at the beginning of a track, with the name of the track, a description of its intended orchestration, and any other information which the user wants to put there. Programs on a computer which does not support non-ASCII characters should ignore those characters with the hi-bit set. Meta event types 01 through 0F are reserved for various types of text events, each of which meets the specification of text events(above) but is used for a different purpose:

FF 02 len text Copyright Notice
Contains a copyright notice as printable ASCII text. The notice should contain the characters (C), the year of the copyright, and the owner of the copyright. If several pieces of music are in the same MIDI file, all of the copyright notices should be placed together in this event so that it will be at the beginning of the file. This event should be the first event in the first track block, at time 0.

FF 03 len text Sequence/Track Name
If in a format 0 track, or the first track in a format 1 file, the name of the sequence. Otherwise, the name of the track.

FF 04 len text Instrument Name
A description of the type of instrumentation to be used in that track.

FF 05 len text Lyric
A lyric to be sung. Generally, each syllable will be a separate lyric event which begins at the event's time.

FF 06 len text Marker
Normally in a format 0 track, or the first track in a format 1 file. The name of that point in the sequence, such as a rehearsal letter or section name ("First Verse", etc.).

FF 07 len text Cue Point
A description of something happening on a film or video screen or stage at that point in the musical score ("Car crashes into house," "curtain opens," "she slaps his face," etc.)

FF 2F 00 End of Track
This event is not optional. It is included so that an exact ending point may be specified for the track, so that it has an exact length, which is necessary for tracks which are looped or concatenated.

FF 51 03 tttttt Set Tempo, in microseconds per MIDI quarter-note
This event indicates a tempo change. Another way of putting "microseconds per quarter-note" is "24ths of a microsecond per MIDI clock." Representing tempos as time per beat instead of beat per time allows absolutely exact dword-term synchronization with a time-based sync protocol such as SMPTE time code or MIDI time code. The amount of accuracy provided by this tempo resolution allows a four-minute piece at 120 beats per minute to be accurate within 500 usec at the end of the piece. Ideally, these events should only occur where MIDI clocks would be located. This convention is intended to guarantee, or at least increase the likelihood, of compatibility with other synchronization devices so that a time signature/tempo map stored in this format may easily be transferred to another device.

FF 54 05 hr mn se fr ff SMPTE Offset
This event, if present, designates the SMPTE time at which the track block is supposed to start. It should be present at the beginning of the track, that is, before any nonzero delta-times, and before any transmittable MIDI events. The hour must be encoded with the SMPTE format, just as it is in MIDI Time Code. In a format 1 file, the SMPTE Offset must be stored with the tempo map, and has no meaning in any of the other tracks. The ff field contains fractional frames, in 100ths of a frame, even in SMPTE-based tracks which specify a different frame subdivision for delta-times.

FF 58 04 nn dd cc bb Time Signature
The time signature is expressed as four numbers. nn and dd represent the numerator and denominator of the time signature as it would be notated. The denominator is a negative power of two: 2 represents a quarter-note, 3 represents an eighth-note, etc. The cc parameter expresses the number of MIDI clocks in a metronome click. The bb parameter expresses the number of notated 32nd-notes in a MIDI quarter-note (24 MIDI Clocks).

FF 59 02 sf mi Key Signature

sf = -7:	7 flats
sf = -1:	1 flat
sf = 0:	key of C
sf = 1:	1 sharp
sf = 7:	7 sharps
mi = 0:	major key
mi = 1:	minor key

FF 7F len data Sequencer-Specific Meta-Event
Special requirements for particular sequencers may use this event type. The first byte or bytes of data are a manufacturer ID. (Growth of the MIDI spec proper is preferred over use of this event type. This type of event may be used by a sequencer which elects to use MIDI as its only file format; sequencers with their established feature-specific formats should probably stick to the standard features when using the MIDI format.)

System Exclusive Events

The system exclusive event is used as an escape to specify arbitrary bytes to be transmitted. It has two forms. (To compensate for some manufacturer-specific modes, the F7h event is used if a F0h is to be transmitted.) Each system exclusive event must end with an F7h event.

The format of a system exclusive event:

OFFSET	*Count*	*TYPE*	*Description*
0000h	1	byte	ID=F0h or ID=F7h
0001h	?	?	Length as variable length qty.
	?	byte	bytes to be transmitted