The Standard MIDI File and DLS File Formats

by Nicholas Esterer for MUMT 621

Standard MIDI File Format

MIDI established in 1983 as a standard protocol for the transmission of musically descriptive data using serial transmission means.

Standardized by MIDI Manufacturers Association (MMA).

The protocol is widely used, highly documented and succinct.

MIDI Messages

The protocol uses a basic vocabulary of 8 bit long words.

Words that have the most significant bit (MSB) set are known as "status bytes".

These tell whatever device receiving a message how to interpret the following bytes, as well as what channel the message is intended for.

This status bytes are followed by data bytes, which have the MSB cleared.

Example MIDI Messages

Status	Data	Data
10010001	00111100	01011010

This indicates a "note on" message intended for channel 2. The note to be played is middle C (60) and the velocity is 90.

Note that because the MSB must be clear for the byte to be interpreted as a data byte, their range is limited to 0-127.

Other MIDI Status Bytes

o Note Off
o Control Change (such as panning or sustain pedal)
o Program Change (to change the current timbre or "voice").

o Status bytes are available for utility commands like downloading samples or tuning synthesizers.

The Standard Midi File (SMF)

A file format for storing sequences of midi data with timing information and meta data (like lyrics and copyright information).

Uses the Resource Interchange File Format style which is a file formatting technique that stores data in chunks.

The chunks have a header indicating what is in the chunk (an ID) and how long the chunk is (in bytes).

What's in an SMF

MIDI Files contain two kinds of chunks: Header and Track chunks.

Header Chunks

<chunk type><length><format><ntrks><division>

<chunk type> is the 4 characters 'MThd'

<length> is the length of the chunk in bytes, stored as a 32-bit word (4 bytes)

<format> is 0 1 or 2

<number of tracks> is the number of tracks

<division>:

If the MSB is 0 it indicates the resolution of the midi file in number of ticks per quarter note indicating the resolution of the midi file.

If the MSB is 1 it indicates the resolution in ticks per frame, with a specified frame-rate (for synchronizing with film).

Formats

Format 0 has a header chunk followed by only one track chunk containing 16 channels of MIDI data. This is the most interchangeable, but also the most limited.

In format 0, the tempo information is stored within the track and all other MIDI messages

Format 1 and 2 contain one header chunk followed by one or more track chunks.

In format 0, the tempo information is stored within the track among all the other MIDI messages.

In format 1 and 2, track 1 usually contains the tempo information only and the other tracks the other MIDI messages.

Track Chunks

Format:

<chunk type><length><event 1>...<event n>

<chunk type> is the string 'MTrk'

<length> is a 4-byte word indicating the length (n x event length)

the events have the format:

<delta-time><event>

Where delta time is the time in ticks since the last event.

This can be 0 if it is the first event or if the events occur simultaneously.

Delta times are stored as variable length arguments where all but the last byte of a continuous stream of words have the high order bit set.

MIDI Events

Events can consist of:

Status and data bytes

"System exclusive" or sysex data

Meta data which contain auxiliary information such as:

o Sequence number

o Lyrics

o Key Signature

o Copyright notice

Downloadable Sounds (DLS)

Developed by MMA in collaboration with the Interactive Audio Special Interest Group around 1995. Originally started by Creative Labs.

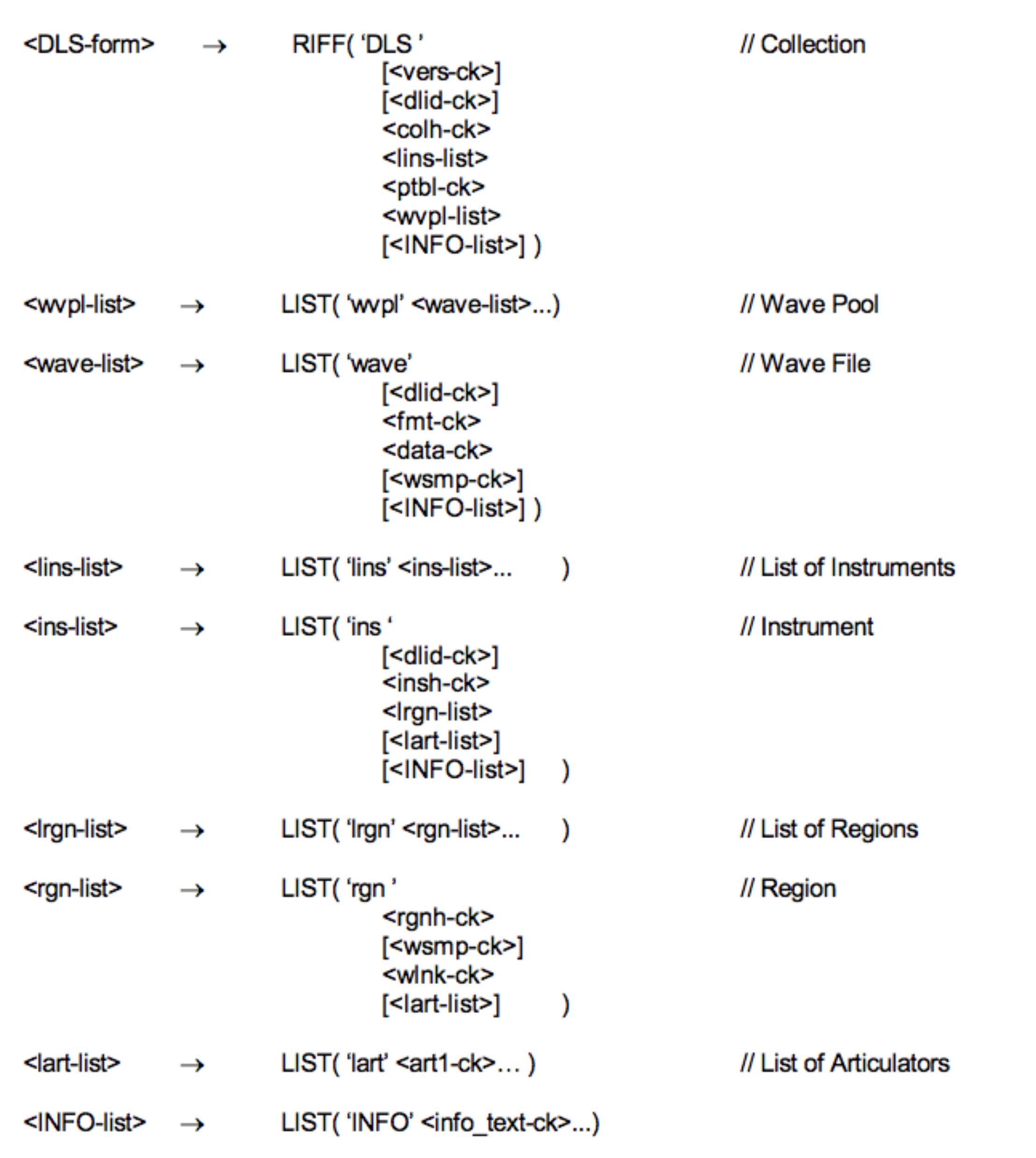
A format for storing:

o Sound samples

o Sample collection mappings to MIDI Program Change and Bank Select messages

o Expression parameters mapped to MIDI controls

DLS File Structure



```
<DLS-form>    →    RIFF( 'DLS '                          // Collection
                        [<vers-ck>]
                        [<dlid-ck>]
                        <colh-ck>
                        <lins-list>
                        <ptbl-ck>
                        <wvpl-list>
                        [<INFO-list>] )

<wvpl-list>   →    LIST( 'wvpl' <wave-list>...)          // Wave Pool

<wave-list>   →    LIST( 'wave'                          // Wave File
                        [<dlid-ck>]
                        <fmt-ck>
                        <data-ck>
                        [<wsmp-ck>]
                        [<INFO-list>] )

<lins-list>   →    LIST( 'lins' <ins-list>...      )     // List of Instruments

<ins-list>    →    LIST( 'ins '                          // Instrument
                        [<dlid-ck>]
                        <insh-ck>
                        <lrgn-list>
                        [<lart-list>]
                        [<INFO-list>]    )

<lrgn-list>   →    LIST( 'lrgn' <rgn-list>...      )     // List of Regions

<rgn-list>    →    LIST( 'rgn '                          // Region
                        <rgnh-ck>
                        [<wsmp-ck>]
                        <wlnk-ck>
                        [<lart-list>]      )

<lart-list>   →    LIST( 'lart' <art1-ck>… )             // List of Articulators

<INFO-list>   →    LIST( 'INFO' <info_text-ck>...)
```

DLS Instrument

Contains:

o List of samples to play.

o How these samples are mapped to MIDI control

o What "articulators" modify the signals according to how the samples are performed.

Instrument Chunk

What?	What controls it	Why?
Locale	Bank and Program change	To choose from various timbres stored in DLS file
List of regions	Internal to DLS file	Stores expression information to be controlled by MIDI messages
Articulation	Internal to DLS file	Stores expression information for synthesizer to be controlled by MIDI messages

Region Chunk

What?	What controls it	Why?
Keyrange	MIDI note number	To map different timbres to different ranges
Keyvelocity	MIDI note velocity	Map different timbres to different velocities
Keygroup	MIDI note number and DLS file	Allows the turning off of other sounds when certain new sounds happen, such as when you play various hi-hat sounds (the closed hi-hat will interrupt the open crash hi-hat sound)

Wave Link (sub chunk of Region)

What?	What controls it	Why?
Channel	Internal to DLS file	Spatialization of sound
TableIndex	Internal to DLS file	Which wave file to play from the pool of samples

Articulation (sub chunk of Instrument)

What?	What controls it	Why?
Source	Some MIDI control value or a constant	To allow more expression. For example, playing a note harder can detune the oscillator more at the attack time, much like it would be when playing a guitar
Control	Some midi control or constant	Allows the user to adjust the level of such an effect
Destination	Internal to DLS file	To specify what the connection controls
Transform	Internal to DLS file	Are the values mapped to an exponential curve, or a linear curve

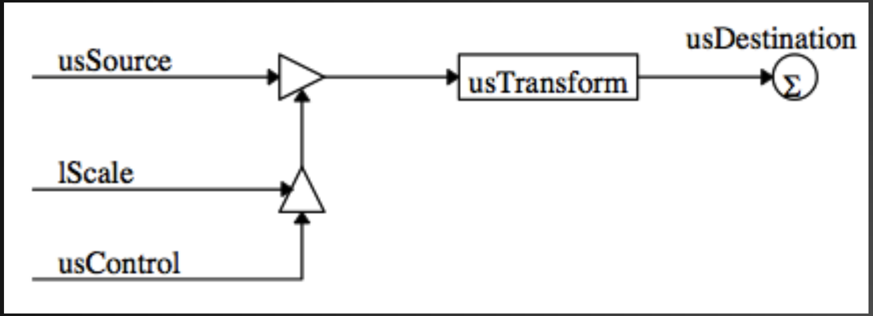
Wave Sample (sub chunk of Region)

Parameters:

UnityNote: What is the original pitch of the sample?
FineTune
Attenuation
Whether or not to loop sound
Where to start looping
How long to loop for

All the parameters are stored with the DLS file.

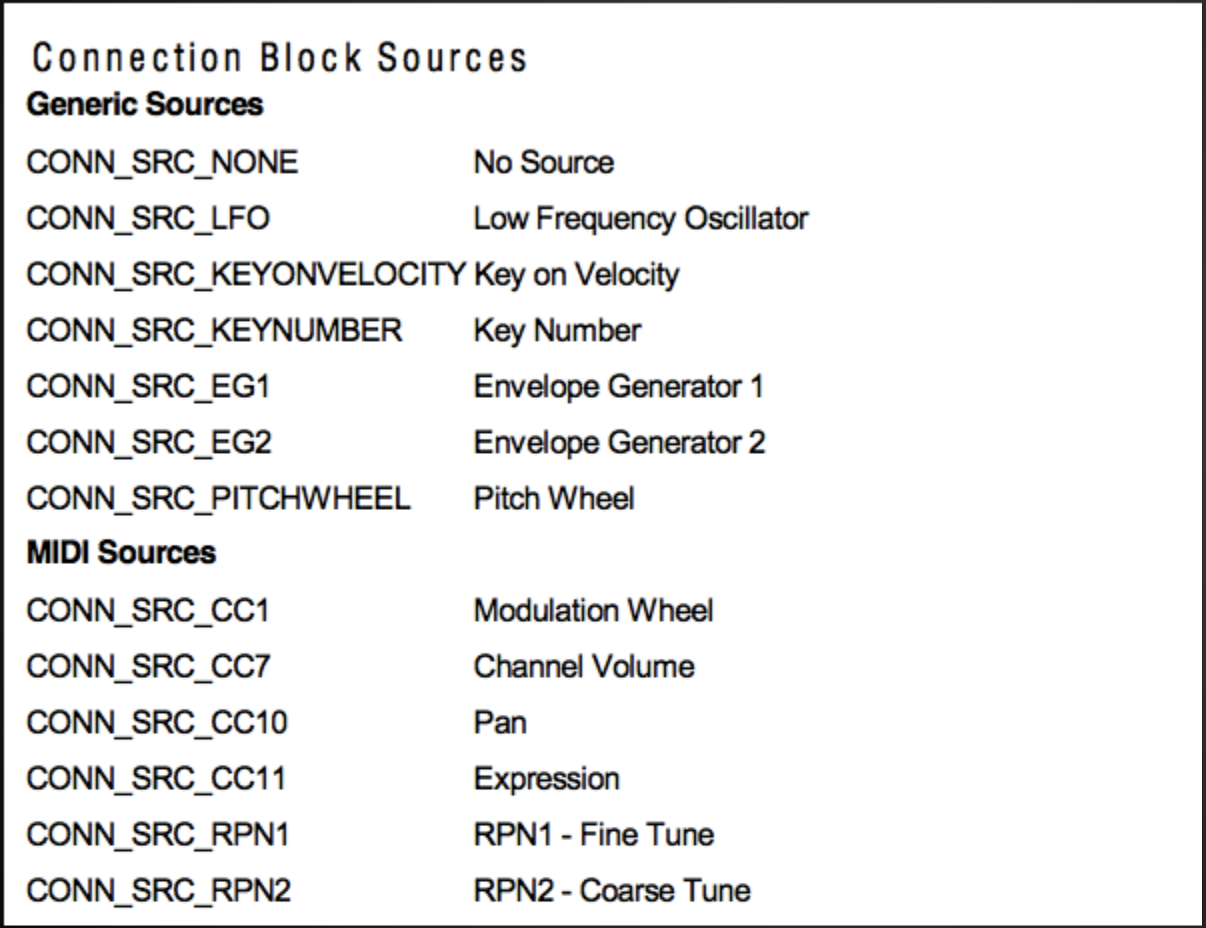
Articulations (sub chunk of Instrument or Region)



(c) 2004 MIDI Manufacturers Association

How the signals are modified by performance parameters
Various sources of signal generation can be connected to a variety of controllers for expressive parametrization.

Connection Blocks

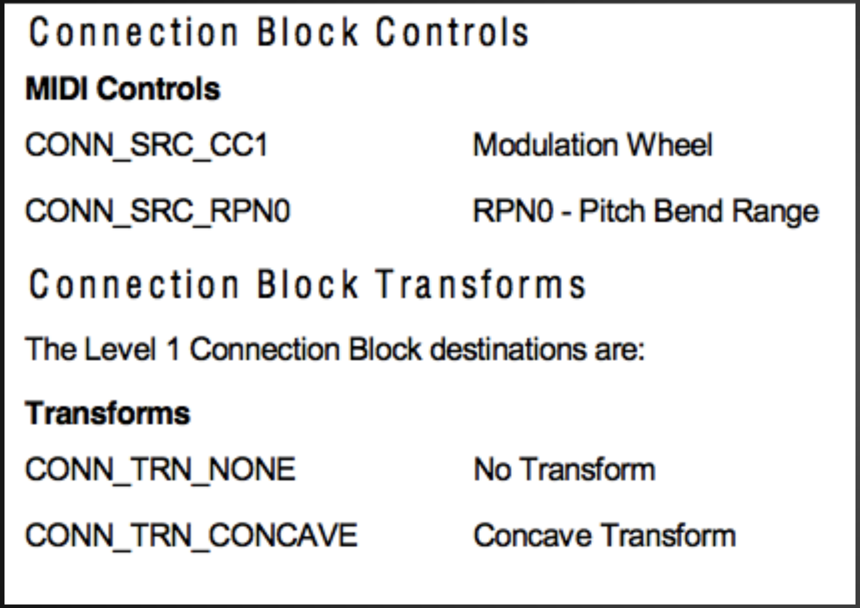


Connection Block Sources

Generic Sources	
CONN_SRC_NONE	No Source
CONN_SRC_LFO	Low Frequency Oscillator
CONN_SRC_KEYONVELOCITY	Key on Velocity
CONN_SRC_KEYNUMBER	Key Number
CONN_SRC_EG1	Envelope Generator 1
CONN_SRC_EG2	Envelope Generator 2
CONN_SRC_PITCHWHEEL	Pitch Wheel
MIDI Sources	
CONN_SRC_CC1	Modulation Wheel
CONN_SRC_CC7	Channel Volume
CONN_SRC_CC10	Pan
CONN_SRC_CC11	Expression
CONN_SRC_RPN1	RPN1 - Fine Tune
CONN_SRC_RPN2	RPN2 - Coarse Tune

(c) 2004 MIDI Manufacturers Association

Connection Blocks



Connection Block Controls

MIDI Controls

CONN_SRC_CC1	Modulation Wheel
CONN_SRC_RPN0	RPN0 - Pitch Bend Range

Connection Block Transforms

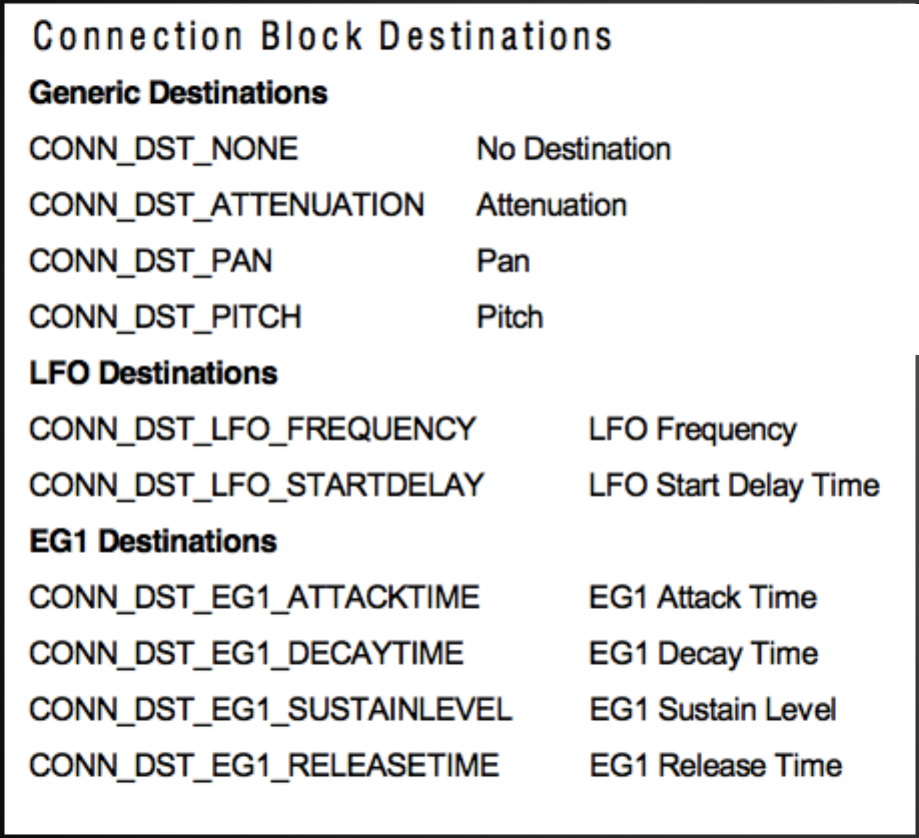
The Level 1 Connection Block destinations are:

Transforms

CONN_TRN_NONE	No Transform
CONN_TRN_CONCAVE	Concave Transform

(c) 2004 MIDI Manufacturers Association

Connection Blocks



Connection Block Destinations

Generic Destinations

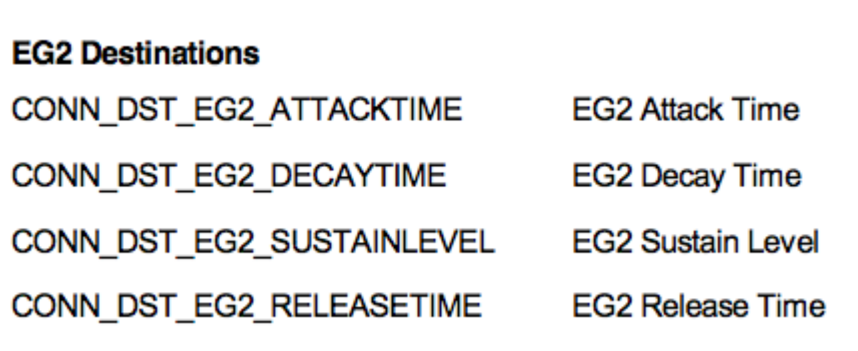
CONN_DST_NONE	No Destination
CONN_DST_ATTENUATION	Attenuation
CONN_DST_PAN	Pan
CONN_DST_PITCH	Pitch

LFO Destinations

CONN_DST_LFO_FREQUENCY	LFO Frequency
CONN_DST_LFO_STARTDELAY	LFO Start Delay Time

EG1 Destinations

CONN_DST_EG1_ATTACKTIME	EG1 Attack Time
CONN_DST_EG1_DECAYTIME	EG1 Decay Time
CONN_DST_EG1_SUSTAINLEVEL	EG1 Sustain Level
CONN_DST_EG1_RELEASETIME	EG1 Release Time



EG2 Destinations

CONN_DST_EG2_ATTACKTIME	EG2 Attack Time
CONN_DST_EG2_DECAYTIME	EG2 Decay Time
CONN_DST_EG2_SUSTAINLEVEL	EG2 Sustain Level
CONN_DST_EG2_RELEASETIME	EG2 Release Time

(c) 2004 MIDI Manufacturers Association

Bibliography

David Burnand. “MIDI.” *Grove Music Online*. *Oxford Music Online*. Oxford University Press, accessed February 5, 2013, http://www.oxfordmusiconline.com/subscriber/article/grove/music/42823.

“Standard MIDI File Structure,” accessed February 4, 2013, http://www.ccarh.org/courses/253/handout/smf/.

“DLS 1 Spec,” accessed February 4, 2013, http://www.midi.org/techspecs/dls/dls.php.

MIDI Manufacturers Association, *Downloadable Sounds Level 1*. ver. 1.1b Los Angeles: MIDI Manufacturers Association, 2004.

“MIDI Message Table 1,” accessed February 4, 2013, http://www.midi.org/techspecs/midimessages.php.