Chris
Sansom

NotaFile

... a Proposed Standard File Format for

Music Notation

(Version 0.5)

NotaFile v 0.5

A Proposed Standard File Format for Music Notation

by Chris Sansom

INTRODUCTION

Like sequencers, word-processors and many other kinds of application software, music notation or scoring packages vary widely and are available on several different types of computer. For this reason there is a good case for a file standard (equivalent to ASCII for text or MIDI File for MIDI data) for transferring information between them—***NotaFile*** is a suggestion for a working title. Compatibility with such a standard could also be incorporated into sequencers as a far more appropriate means than MIDI Files for transferring data into scorewriters.

Work may need to go through several different stages, possibly at different locations, using different hard- and software. Music may be developed, by means of a sequencer (on, say, an Atari ST) in a studio, from which a score and parts would be required. A ***NotaFile*** could be prepared, which would then be taken to a bureau or other suitable firm who would import it into a scoring package (on, say, an Apple Macintosh). The same file (possibly modified by the first scoring package) might be imported into a second scoring package (on, say, an IBM PC) with superior facilities for part extraction. There are many other possible combinations for which ***NotaFiles*** could be useful. *NB:* no reference to specific hard- or software here is intended to imply any preference on the part of the author.

This document consists of four main sections: the rationale and overview which follows below, a detailed specification, appendices containing tables of byte values and, finally, three pieces of "real" music manually encoded into ***NotaFile*** format in order to demonstrate that this standard could work—references to these examples are made at a few points in the main text.

Why not just use MIDI Files?

Although the MIDI File standard has, in a few respects, become somewhat muddled, and rather less standard than it should be, it is in principle an excellent way of transferring MIDI data between programs that process it. It is, however, inappropriate for scoring software …

1. MIDI Files include (or may include) a great deal of information which is irrelevant to such applications, such as Program Change, Pitch Bend, System data, some of the "meta-events" and much of the controller data (see below for a suggestion as to how some of this could be translated by sequencers), as well as channel information for every event. Some of this data also makes very intensive use of memory and disk space.
2. MIDI Files omit certain essential information which could easily be included in ***NotaFiles***, such as clefs, enharmonic information (*eg* D♯/E♭—taking double-flats/sharps and quarter-tones into account, allowance should be made for 70 different note names in an octave), beaming of eighth (and shorter) notes, slurs, etc.
3. Much of the potentially useful information in MIDI Files is in a form which is inappropriate to the purposes of a scoring program: all timing values are expressed in absolute numbers of "ticks", giving rise to quantization problems with all but the simplest

rhythmic usage (very few scoring packages can interpret a quintuplet correctly when importing from a MIDI File, and when they do it has to be by means of mathematical approximation, which is not ideal), and completely ignoring "rogue" elements such as grace notes; information about dynamics is also in an inappropriate form, or more correctly, in several inappropriate forms, such as key-velocity, aftertouch, breath control or volume.

4. MIDI Files, when implemented correctly, provide an extremely compact way of transferring data. However, serious notation software is less likely than MIDI software in general to be used on small, floppy disk-based systems, and it would be worth accepting a somewhat larger type of file in order to transfer the right data.

NB NotaFiles are not intended in any way to replace MIDI Files, but to be complementary to them, and many programs would be expected to implement both standards.

What kinds of software package would implement NotaFiles*, and how?*

There are three main categories:

1. **Sequencers which do not include any notation facilities.** These would most likely use ***NotaFiles*** for export only (if *importing* from a scoring program, it would probably be better to use MIDI Files), and would be able to implement the standard in only a fairly limited way—much of the information would be absent, *eg* beaming information, and it might be hard to define phrase marks, etc. without adding considerably to the program's facilities (*eg* in a graphic or event editor there could be some way of flagging notes or other data with extra information, such as *this very short note should be interpreted as a grace-note, not a 64th note*).

   Such programs could well offer an option as to which data is to be interpreted as dynamic (loudness) information—some of these are listed at the end of point 3 above—and the user could perhaps set ranges of values for each dynamic mark, *eg* 72–88 for ***mf***. Another possibility would be for pitch bend data to be interpreted as notated *glissandi*—in this case the user would have to specify the pitch bend range of the voice patch for which the part is intended.

2. **Sequencers which do include notation facilities.** This is an increasing trend amongst sequencers, and these programs are in an ideal position for implementing ***NotaFiles***. They would still be most likely to use the standard for exporting data, but it is certainly conceivable that they might offer import facilities also. Even packages which offer only fairly basic notation facilities would usually cover all the types of data envisaged for the ***NotaFile*** standard.

3. **Scoring packages.** These would certainly import ***NotaFiles***, and very likely export them as well, as described at the start of this document.

These are merely suggestions, and the details of how ***NotaFiles*** were implemented would be up to the programmers (and users, hopefully) of this software. The possibility is not excluded that other types of music software, such as those that generate music in various ways other than from a MIDI instrument (even some which have nothing to do with MIDI), would implement ***NotaFiles***.

The general concepts behind NotaFiles

The overall framework of a ***NotaFile***, and several of the more "internal" principles, would be somewhere between those of a format 0 and a format 1 MIDI File. This includes the notion of allowing for possible future developments—even to the point of amalgamating the two at some point (though this may, after detailed consideration, prove to be impractical). The suggestions which follow, therefore, are designed with this in mind and conform fairly closely to the existing syntax of MIDI Files.

1. There should be a header or "master" chunk which would identify the file type and include any information which applies to the whole score: the number of staves, initial clef for each staff, time signatures, tempo marks, rehearsal marks, title, etc. There is no reason why many of these data types could not appear in the music chunk (to follow), but it would be up to the software (and the user) to avoid conflicts, because any events found in this chunk should be assumed to be global. Unlike format 1 MIDI Files, however, there should not be the option to include music in this chunk—it would make no sense to do so.

2. The music chunk would make up the bulk of the file, containing the information necessary for a program to produce, automatically, a fairly basic "scroll view" of the music. It is not envisaged that any aspects of layout would be included in this type of file: that, again, would be up to the programs and their users.

It should be borne in mind that there are two fundamental differences between MIDI Files and ***NotaFiles***:

1. The MIDI File is based on an existing standard form of digital data transmission (MIDI) whereas the ***NotaFile*** is not.

2. The MIDI File works in a necessarily sequential way (in terms of time)—in keeping with the aurally perceived nature of music—whereas this need not, and often cannot (as will be seen below), be the case with the ***NotaFile***, which deals purely with the visual aspect of music "on the page". It would be difficult to over-stress this last point.

NB We should learn from the mistakes of the MIDI File, and ensure that there is one file format, and that it is strictly adhered to—especially with regard to some of the potentially vaguer types of event involving various sorts of text.

How NotaFiles *work (with reference to MIDI Files for comparison)*

In the rest of this document, bit 0 means the least significant bit of a byte, and bit 7 the most significant. For convenience, the wording will be as if the **NotaFile** *standard were already established.*

Unlike the MIDI File, in which MIDI events and timing bytes are two quite distinct entities (and alternate strictly), all elements in the ***NotaFile*** are considered as events, including those which concern time.

In a MIDI File, MIDI events are always separated by "delta times" which represent *the time that elapses between the two events*. In a ***NotaFile***, for reasons which will become apparent, all timing information is independent of the preceding event, and is required only when there is a change —not between every pair of adjacent non-timing events. These timing events should more properly be regarded as "position in score" events, since we are dealing with a visual medium.

There are three sorts of positioning events, the values of which are always absolute:

1. **Change of measure.** This occurs whenever the next piece of notation is in a different measure from the previous one, whether earlier or later.

2. **Change of position in measure.** This occurs whenever the next event is in a new place within the measure (again, whether earlier or later): *eg* if a note, an accent, a dynamic mark and a piece of text all occur at the same point, this event is required only once, and not between them.

3. **Change of staff.** This allows, among other things, groups and phrases (especially in keyboard music) to cross from one staff to another. It also opens up the possibility of taking a "vertical" approach to music of a chordal nature (see the Mussorgsky example at the end). It is why the ***NotaFile*** is not organised in "tracks" like a format 1 MIDI File.

Any event can occur in the music chunk, including time signatures and tempi. This allows music of great rhythmic complexity to be included in a ***NotaFile***, with different time signatures and/or tempi on different staves simultaneously. The CHANGE OF STAFF event in such cases has to be handled with great care by the software, since measure 47 on one staff may be at a very different point in time from measure 47 on another, and it is at this new point that the next event will be expected.

This system allows considerable freedom in the way events are ordered in the file. In principle almost any order is possible (even backwards, from bottom to top), except that certain types of grouped events *must* appear contiguously in the file, such as beamed groups of eighth (or shorter) notes and tuplet groups, thus allowing overlapping groups on a polyphonic staff to be included correctly. (This applies only to the order of *note* events in such instances—other information such as expression marks can come in between note events in groups, as indeed change of position events *must*.)

In general, greater flexibility requires greater care on the part of those (mainly programmers) handling the data.

It is, of course, assumed that for the sake of coherence most programmers would prefer to create their files in a reasonably logical order, though that order could well simply be the order in which the user has input the data.

There is, intentionally, nothing in the standard which provides any checks for musical "correctness"—in particular, correct numbers of beats in measures (for instance, there would be nothing to prevent a CHANGE OF POSITION IN MEASURE event from indicating the fifth quarter note in a 4/4 measure). As with so many other aspects, it is a matter for programmers to decide whether their individual products should do this, but the idea here is to allow for unorthodox notational usage if it is desired.

Apart from the events which concern position in the score, there are three other broad groups of events:

1. **Events which have a duration.** These include notes, rests, slurs, octave marks, *crescendi/diminuendi*, trills, pedal marks, etc., though the way the duration is treated varies according to the event type.

2. **Events which have no duration.** These include tempi, time signatures, key changes, clefs, dynamic and expression marks, etc.

3. **Text events.** There are only a few types of these, but they should be used correctly.

Finally, there are three other events:

A **Screen message event** for special purposes,

A **Software-specific event** similar to the "sequencer-specific meta-event" in the MIDI File standard; and the obligatory …

End of Chunk event to provide programs with a means of double checking this information.

As will be seen in the Appendices to this document, there is a far greater need, on the part of programmers, for extensive reference tables than is the case for MIDI Files. These should always be provided as part of the specification. This inevitably leads to many more of the `switch()` or `case` type of statement in the program routines which handle ***NotaFiles*** than is required for MIDI File handling routines, but the results should justify the effort.

DETAILED SPECIFICATION

A suggested file extender for systems which require an MS-DOS-type filename is `.NFL` (at least one notation package already uses `.NOT` for its own files).

Events are distinguished from each other, as in MIDI Files (and indeed MIDI itself) by means of a defining byte (bit 7 set), equivalent to that known as a "status byte" in MIDI. These are followed by varying numbers of data bytes (bit 7 usually clear). However, MIDI status bytes have to include channel information in the low nybble, thus restricting the status types to 8, whereas channel data is irrelevant to ***NotaFiles***, leaving us with the luxury of a possible 128 types. Many of these are, and may well remain, undefined, and those that are defined are grouped in a reasonably logical manner. A few events have no data, and thus consist of only a defining byte.

Much of the following paragraph is lifted *verbatim* from the *Standard MIDI Files 1.0* documentation …

Numerical values which may be greater than 127 (in particular measure numbers and length values for text events), as in MIDI Files, are represented as variable-length quantities. These numbers are represented 7 bits per byte, most significant bit first, to a maximum of 4 bytes. All bytes except the last have bit 7 set, and the last has bit 7 clear. If the number is between 0 and 127, it is thus represented as one byte. The largest number allowed is `0FFFFFFF` (hex), represented as `FF FF FF 7F`, which is in the region of 2.5×10^8, which should be enough for any purpose (in file size terms, around 240 megabytes).

Many other values are in terms of musical note values, represented by one byte having the value of the denominator, with a second byte to account for dotted notes. A quarter note, therefore, is given as `04 00`, a dotted half note as `02 01`, and a double dotted 16th note as `10 02`. The only inconsistency in this system arises from the need for notation of a breve (2 whole-notes in length), which would be given as `00 00`, which therefore does not mean zero duration—quite the contrary: it means the longest available note value. This system allows for (written) note values from a double (even triple or more) dotted breve to a 127th note, and allows for tuplets to a fair degree of complexity, *eg* a quintuplet 8th note is really a 10th note, shown as `0A`, and a triplet 16th note nested in that quintuplet is a 30th note, shown as `1E`. CHANGE OF POSITION IN MEASURE events (and some others) sometimes consist of a string of such values, so a data-length byte (or, very unlikely, bytes) is included.

File Structure

HEADER CHUNK

This is, as mentioned above, similar to a combination of the header chunk and the first track chunk in a format 1 MIDI File. The syntax is:

```
<Header Chunk> = <chunk type> <length> <no of staves>
(<clefs> <staff blocks>) <events>
```

`<chunk type>` is the four ASCII characters NThd.

`<length>` is a 32-bit (not variable-length) representation of the length of the chunk that follows.

`<no of staves>` is a 16-bit word containing the maximum number of staves anticipated in the score (to a theoretical maximum of 65,535). This may be more than are actually used, and it is up to the programs which use the file whether or not empty staves are shown. It may also be zero, if all that is required is a tempo, meter and structural "map" for a score, therefore the staves are numbered from 1, not 0.

`<clefs>` If the number of staves is not zero, there should be one byte for each staff here, giving the initial clef of that staff. This is obligatory. The numbers in these bytes refer to a table of available clefs (see detailed description of CLEF CHANGE event below).

`<staff blocks>` If the number of staves is not zero, there should be one byte containing the number of blocks or groups of staves, up to 127. If no staves are blocked, this byte may contain zero, but it must be present. If it is not zero there will be a further 5 bytes of information for each block. The first pair of these bytes is a 16-bit representation of the number of the first staff in the group, the second being a similar representation for the last staff in the group. The last byte contains three flags: bit 0 is set if the barlines are to be blocked (continuous) across the staff block; bit 1 is set if a square bracket is desired for the block in the left margin; bit 2 is set for a curly brace ({). Bits 1 and 2 may not both be set, but both may be clear, in which case no bracket will be applied to the block.

Two staff blocks may be superimposed, but one must be entirely included in the other. If the outer block has blocked barlines, bit 0 of the flags byte for the inner block will be ignored; if the outer block does not have blocked barlines, the inner group's barlines may be blocked separately.

In complex music with different meters and/or tempi within a block, the barlines should automatically be unblocked wherever they do not coincide, whatever the setting for that parameter.

`<events>` may include anything that might reasonably be applied to the score in a global way, as described above. Any such events should be assumed by the importing program to be global. The table of defining bytes in *Appendix A* gives information on which events are and are not allowed in the header chunk. If any disallowed events are exported in this chunk by badly-behaved software, they should be ignored by the importing software. There must always be at least one event in this chunk, even if it is only the END OF CHUNK event.

The most minimal possible header chunk is therefore:

`4E 54 68 64`	`00 00 00 03`	`00 00`	`FF`
(NThd,	length 3,	no staves,	end of chunk)

MUSIC CHUNK

The syntax of this is the same as for the header chunk, but without the information about staves:

```
<Music Chunk> = <chunk type> <length> <events>
```

Even if there is no music in it, this chunk must be present and must contain at least an END OF CHUNK event.

`<chunk type>` is the four ASCII characters NMus.

`<length>` is again a 32-bit representation of the length of the chunk following.

`<events>` may be any events, including those generally associated with the header chunk, and it is up to programmers to ensure that conflicts are avoided.

The most minimal possible music chunk is therefore:

```
4E 4D 75 73   00 00 00 01   FF
(NMus,        length 1,     end of chunk)
```

Events

This section lists each event type with its defining byte and the syntax for data bytes. Any non-variable bytes are given directly in hexadecimal; the notation `len` refers to a number, in the form of a variable-length quantity, which specifies how many data bytes follow in the event; `varlen` refers to any other variable-length quantity in the event; `text` and `data` refer to however many bytes of (possibly text) data were specified by `len`; a 2-letter mnemonic such as `vv` or `nu` refers to a 1-byte value, usually 7 bits.

The following information is also shown in tabulated form in the Appendices, of which *Appendix A* is a table of defining bytes.

POSITIONING & MEASURE-RELATED EVENTS

`80 varlen` **Change of Measure**

The value represents the number of the measure in which the next event occurs.

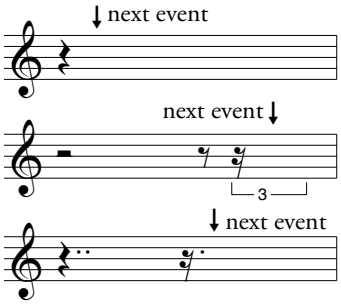
Examples …

Move to measure 50	`80 32`
Move to measure 135	`80 81 07`

`84 len data` **Change of Position in Measure**

The place in the measure, in musically notated terms, at which the next event occurs (it can be looked at as the duration which has elapsed in the measure *before* the event). If this event is absent immediately after a CHANGE OF MEASURE event, the next event will be at the beginning of the new measure.

Examples …




```
84 02 04 00
84 06 02 00 08 00 18 00
84 04 04 02 10 01
```

`88` **First time bar**

This would normally appear in the header chunk. It requires no data, since the mark should be terminated by the program when the end repeat barline occurs.

`89` **Second time bar**

Again, normally in the header chunk.

`8C bl` **Barline type** *(Appendix B)*

Usually in the header chunk, this is a "this time only" change—*ie* the next barline is automatically a normal one, although a normal barline is also allowed for in the data table. The types of barline allowed for include double, triple, repeat and dotted. The barline specified is the one at the *end* of the current measure (it is up to programs and users to ensure that, for example, the final barline is not a start repeat barline).

Example …  `8C 06`

`8D ss ss` **Change of Staff**

A 16-bit number, giving the number of the staff on which the next event occurs. Unless this event is followed by a CHANGE OF MEASURE and/or a CHANGE OF POSITION IN MEASURE event, it should be assumed that the next item of notation will appear at the same position in the same measure as the previous one, but on the new staff.

Example … Move to staff 37 `8D 00 25`

`8E vn` **Change of Voice/Stem Direction**

The high nybble contains a number, usually no greater than 2, indicating in which voice on the staff to place the next event. Bit 0 indicates whether or not to specify stem directions: if it is set, bit 1 is set for up, clear for down. If bit 0 is clear, it should be assumed that voice 1 has stems up and voice 2 down, and bit 1 will be ignored. This event should always immediately follow a NUMBER OF VOICES ON THE STAFF event (`CE`) if that number is greater than 1. It can also be used where there is only one voice, to specify stem directions—in this case, the high nybble is ignored and may be left empty.

EVENTS WITH DURATION

`9n nn ll dd ii ii` **Note** *(Appendix C)*

This is by far the most complex event, and will take some space to describe.

The defining byte `9n` is similar to a MIDI status byte, in that the high nybble signifies a note and the low nybble the octave, giving a possible range of 16 octaves (which should

cover everyone's needs), though most notes are in about the central 8 octaves. Middle C is thought of as C8 for ***NotaFile*** purposes.

The first data byte `nn` is the note name, which is, again, in two nybbles. The high nybble can have a value of 0–6 for the letter (C, D, E, F, G, A, B) and the low nybble has a value of 0–9 for the accidental (if any)—this includes double-sharps and -flats, and quarter-tones. *NB:* This is intended to represent what is written on the page, not what sounds—for example, in the key of C, a second F♯ may appear in a given measure, so the ♯ symbol is not required a second time. What is notated, therefore, is an F (no accidental).

Another potential source of confusion concerns B♯ and C♭ (similarly E♯ and F♭ and conflicts between double-sharps/flats and adjacent naturals). Since we are concerned with notation rather than sound, a C8 which has a ♭ before it would be stored in the file as C♭8, even though it sounds as a B7; similarly a B7 sharpened sounds as C8 but should be stored as B♯7.

If there has been a SPECIAL NOTEHEADS event which specified tablature, the first two bytes function differently. The low nybble of the first byte and the high nybble of the second now serve merely to position the note on the staff, and the low nybble of the second byte contains the number (0–15) which replaces the normal notehead.

The next 2 bytes `ll dd` are the notated value of the note and a byte for dots (if any). This technique has already been described above, but in the case of notes there is no need for a data-length byte as there will always be two. This is because tied notes must always be notated separately (see below).

The last 16-bit word `ii ii` (of which bit 7 *may* be set) contains 15 bits acting as flags giving various supplementary information about the note. These are as follows:

Bit 0 set if this note is the first of a beamed group.

Bit 1 set if this note is the last of a sub-group within a beamed group, *eg* eight 16th notes may be beamed as two groups of four, with a single beam joining those two. This bit would be set in the fourth of the eight notes.

Bit 2 set if this note is the first of a sub-group. See Bit 1, but this bit would be set for the fifth note in the example.

Bit 3 set if this note is the last of a beamed group.

Only one of bits 0–3 may be set.

Bit 4 set if this note is the first of a tuplet group. The value of the note should specify what kind of tuplet is required, and this is provided as a double-check. If a previous note had this bit set, without the routine encountering one with bit 5 set (see below) in between, it should be assumed that a nested tuplet is beginning.

Bit 5 set if this note is the last of a tuplet group (nested or otherwise).

Only one of bits 4 and 5 may be set.

Bit 6 set if brackets are required round the notehead.

Bit 7 set if brackets are required round the accidental only.

Only one of bits 6 and 7 may be set.

Bit 8 set if the note is a grace note with no stroke through the stem.

Bit 9 set if the note is a grace note with a stroke.

Only one of bits 8 and 9 may be set.

Bit 10 set if this note is tied to a following note.

Bit 11 set if this note is tied to a preceding note.

Bits 10 and 11 may both be set, in which case the note is tied before and after.

Bit 12 set if this note requires *tremolando* strokes through the stem.

Bit 13 set if this note has a special notehead. This is a "this note only" instruction and the next note event should return to a normal notehead unless this bit is set again. There is a separate event for making a change to all the noteheads which follow, for percussion, etc.

Bit 14 set if this note is at the beginning of a *glissando*.

If any of bits 12–14 are set, further bytes of data will follow:

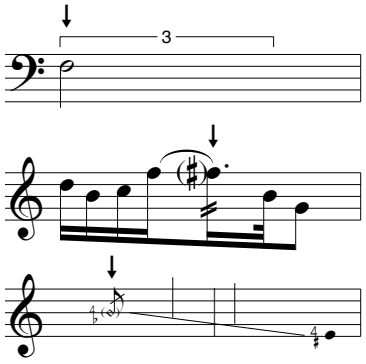
Bit 12 leads to a single byte containing the number of strokes required through the stem (usually up to 4).

Bit 13 leads to a single byte containing a number representing the type of notehead required—this is the same as for the SPECIAL NOTEHEADS event (see below). If bit 4 of this extra byte is set, the note is to be taken as a one-off tablature notation and the low nybble contains the number (0–15) which replaces the normal notehead.

Bit 14 leads to a number of bytes which point to the note at the end of the *glissando* line. They consist of a dummy POSITION IN MEASURE event (complete except for the defining byte) to represent the *length* of the *glissando* and the first two bytes of a dummy NOTE event (the first with the high nybble zero, to give the octave only).

NB: If more than one of bits 12–14 are set, the extra data bytes *must* follow in the order in which they have been described here: *tremolando*, notehead, *glissando*.

NOTE EVENT: examples …




```
97 30 03 00 00 10

99 33 10 01 18 84 02

98 66 08 00 62 40 44 02
02 00 08 27
```

The last two are far more complex than would normally occur, but they should give some idea of the possibilities.

Grace notes, theoretically, have no time value, so should be positioned at the time of the next note following the grace note(s). If there is more than one, they should be given in order with no (time) positioning events in between—see the Debussy example at the end. The same rules apply as to normal notes about contiguous beamed groups.

The importing program should treat information about tied notes carefully and allow for anomalies such as the A♭ in the Debussy example (right hand, first measure, second beat). A "tied after" flag should make the program look for the next note at that pitch on that staff with a "tied before" flag, but not necessarily immediately after the duration of the note with the "tied after" flag. The converse should be applied to a "tied before" flag.

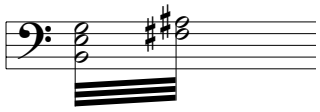
`Dn nn ii` **Note in a Chord with the preceding note event**

Although this is out of numerical order (it is, in effect, a "without duration" event), this is the appropriate place to describe this event. It is identical to the NOTE event, except that the only information contained in it is the name of the note, along with two flags in the third byte. These are bits 2 and 3 (for consistency with the first byte of flags in the NOTE event) to indicate ties before and after. The note should be placed on the same stem as the previous note event read, to form a chord, and matches that note in all other respects. There may be as many of these in succession as desired. This should not be immediately preceded by any of the CHANGE OF POSITION events.

`8F ll dd na data nb data ss` **Two-Element *Tremolando***

This is also out of numerical sequence, but is best described after the NOTE events. The `ll` and `dd` bytes work as for notes; `na` and `nb` contain the number of notes in each of the two "chords" to be alternated (this number may well be 1, in which case it is not a chord) and determine how many bytes appear in each of the two batches of data: these contain two bytes for each note, exactly like the two bytes of the NOTE IN A CHORD event, except that the high nybble of the first byte is zero. The low nybble of `ss` contains the number of tremolando strokes to be notated (up to 15!) and bits 4 and 5 of this byte act as tuplet start and end flags as in the last `ii` byte in the NOTE event.

Example …



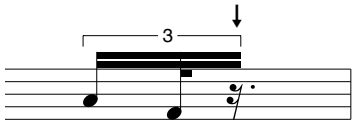
```
8F 02 00 03 06 60 07 20
07 40 02 07 33 07 53 03
```

`A0 ll dd ii` **Rest** *(Appendix D)*

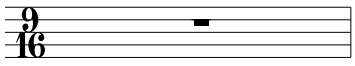
The first two data bytes `ll dd` are exactly the same as those for notes, with one exception: (on the assumption that no-one will want a rest with more than 63 dots after it) if bit 6 in the `dd` byte is set, the remaining bits of that byte are ignored and the `ll` byte is interpreted as a number of measures' rest (if this is 1, it should be shown with the same symbol as a whole-note rest). This means it is not possible to have more than 127 measures shown as a combined rest—it is, in any case, bad practice to leave players counting for too long without cues.

The last byte `ii` is the same as bits 0–6 of the supplementary information word in the note event: this allows rests to be included in beamed groups, specifies their relationship to tuplets and allows for brackets to be added.

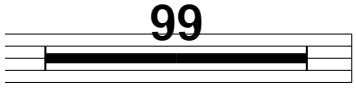
Examples …




```
A0 18 01 28
```



```
A0 01 40 00
```
(whatever the time signature)




```
A0 63 40 00
```

`A1 ss ss len data` **Slur, above or below unspecified**

The first two data bytes give the number of the staff where the slur should end—if (as is usually the case) the slur ends on the staff where it began, this value may be left as zero. This allows slurs to be included which cross staves, which is particularly useful in keyboard music.

The remaining data is of the same kind as that in the CHANGE OF POSITION IN MEASURE event, except that it represents the *length* of the slur.

`A2 ss ss len data` **Slur, above the music**

`A3 ss ss len data` **Slur, below the music**

`A4 len data` ***Diminuendo*** **"hairpin"**

Again, the data represents the *length* of the hairpin. The next few events are similar in this respect.

`A5 len data` ***Crescendo*** **"hairpin"**

`A8 len data` ***8va*** **mark**

`A9 len data` ***8va bassa*** **mark**

`AA len data` ***15ma*** **mark**

`AB len data` ***15ma bassa*** **mark**

`AC ii len data` **Trill, above or below unspecified**

The low nybble of the data byte `ii` may have a value of 0–9 and represents an accidental (if any) to be added above the mark. This nybble works the same way as for note names. In bits 4, 5 and 6 are three flags giving other information:

Bit 4 set if the accidental is required below the trill mark, rather than above. This bit is ignored if there is no accidental.

Bit 5 set if the *tr* part of the mark is to be omitted.

Bit 6 set if the wavy line is to be omitted. It will be omitted anyway if the duration is zero.

The `len` and `data` parts of the event work as described for the hairpin and *8va* events.

`AD ii len data` **Trill, above the music**

`AE ii len data` **Trill, below the music**

`B0 ss ss len data len (text)` **Bracket, above or below unspecified**

This is an all-purpose square bracket which may, like slurs, cross staves, so the `ss ss len data` part of the event works exactly as for slurs. This may be followed by a piece of text to be included in a break at the central point of the bracket. The length of this text is, as usual, a variable-length quantity. If it is zero, the text is omitted and the bracket should be continuous.

`B1 ss ss len data len (text)` **Bracket, above the music**

`B2 ss ss len data len (text)` **Bracket, below the music**

`B4 len data` **"House", above or below unspecified**

This is a horizontal square bracket, similar to a first-time bar mark but without the figure 1. `len` and `data` work in the usual way.

`B5 len data` **"House", above the music**

`B6 len data` **"House", below the music**

`B8 len data` **Type 1 pedal mark, start:** Ped.

`B9 len data` **Type 1 pedal mark, continuation:**

`BA len data` **Type 2 pedal mark:** Ped. ✻

`BD len data` **Line, dots**

An all-purpose horizontal dotted line which may, if necessary, be repositioned by the user after the file has been read. This can be used, above the music, to show the duration of a special instruction.

`BE len data` **Line, dashes**

Similar to the previous event. This is often used for instructions like *accel.* – – –

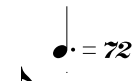
`BF len data` **Line, solid**

Events Without Duration

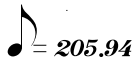
`C0 ll dd nn nn pp` **Tempo Mark**

Although more appropriate to MIDI Files than to ***NotaFiles***, allowance is made here for tempi to 2 decimal places. The `ll` and `dd` bytes give a note value in the same way as for notes and rests. The two `nn` bytes are 7-bit numbers, which allows up to 128 x 128 – 1 = 16,383, which should be more than enough. The `pp` byte is a literal representation of the decimal part of the number, therefore up to 99. *NB:* .2 should thus be represented as 20 (`14` hex) in the final byte, not 2.

Examples …



`C0 04 01 00 48 00`



`C0 08 00 01 4D 5E`

`C2 dn nm` **Time Signature**

This is expressed literally, to allow for unorthodox usage. An exceptional case arises for the "common time" and "*alla breve*" symbols. For both of these, the `dn` byte is zero. When this is the case, the `nm` byte is also zero for common or 1 for *alla breve*. If no time signature is found in the file, a default of 4/4 should be assumed.

Examples …

`C2 06 08`

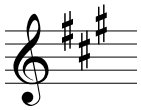
`C2 00 01`

`C2 11 0A`

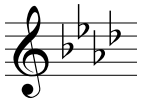
`C4 ks` **Key Signature (conventional)** *(Appendix E)*

The data byte is divided into two nybbles: the low nybble states whether sharps or flats are required (value of 3 or 2 respectively as in other instances where accidentals are involved); the high nybble is a number from 0 to 7, thus allowing the normal range of keys in one byte (and keeping bit 7 clear). If the key is C major or A minor, the low nybble is ignored.

Examples …



`C4 33`

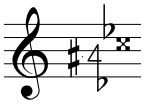


`C4 42`

`C6 len data` **Key Signature (literal)**

This is to allow for unconventional usage and may show any combination of sharps, flats, etc. There should be one data byte for each accidental in the signature, which works exactly as the note-name byte for the NOTE events. The `len` will be the number of these bytes in the event.

Example …



`C6 04 53 46 22 05`

`C8 cl (sl)` **Clef Change** *(Appendix F)*

The `cl` byte is a number in a table of available clefs: treble and bass, both with or without *8* or *15* marks above or below; alto, tenor and soprano C clefs as well as the more archaic form of the soprano clef; 3 different percussion clefs; all in large or small size—or no clef at all.

There is a special case if the low nybble of `cl` equals 7: an extra byte follows containing the number of the staff line (the bottom line being 1 and counting upwards) on which the clef is to be centred. That line will then become G8, F7 or C8 depending upon which clef is used.

If a percussion clef or no clef is applied, a convention is required for how to place notes. It should be assumed that the staff is equivalent to the treble staff, from the top downwards: *ie* that the top line is F9, the next is D9 and so on for however many lines the staff has.

Examples …

`C8 40` (Small treble clef)

`C8 31` (Large percussion clef 2)

`C8 7F` (No clef)

`C8 27 04` (Large C clef centred on 4th line —equivalent to, but distinct from, tenor clef)

`CA tr` **Transposition**

From this point on, the music that is read from the file on that particular staff should be transposed the stated amount. This is useful where orchestral parts double up in a score, but the instruments are in different transpositions. Like MIDI pitch bend, `40` is taken as the mid-point (no transposition). It may well be an option for the user to decide whether a given instrument is to be shown at concert or transposed pitch in the score (most logically, concert pitch in the score and transposed in the part).

Examples … Up a perfect fourth `CA 45`
Down a major thirteenth `CA 2B`

`CC nl` **Number of Staff Lines**

This can occur at any point and is particularly useful for percussion parts. The number of lines in the staff becomes the number specified in `nl` and remains so until the next change. All staves are assumed to have 5 lines until this event is encountered.

Since orthodox clefs no longer apply when the staff has other than 5 lines, the same rule for pitch applies as for percussion or no clef (see above).

`CD nh` **Special Noteheads** *(Appendix G)*

Unlike the bit which can be set in any note event to change the head of that note, this is an "until countermanded" instruction. The number in `nh` refers to a table of available noteheads. If bit 5 is set, the stem is omitted; if bit 6 is set it is a small notehead (grace notes will be small anyway, but this should still be specified if the grace note has a special head unless a particular, grotesque effect is desired). If bit 4 is set, the low nybble is ignored, as the note names will now provide the tablature numbers which replace the normal noteheads.

`CE pp` **Number of Voices on the Staff**

This may change from time to time. It is just about possible to cram four parts on a staff, but two is the recommended maximum. It is assumed to be one per staff for the whole score until this event turns up. If the number is greater than 1, this must be followed by a CHANGE OF VOICE event (`8E`), otherwise the importing software will not know in which voice to place the next event.

`Dn nn ii` **Note in a Chord** (see above)

`E0 dd` **Dynamic Mark** *(Appendix H)*

The numbers `01` to `0F` represent numbers of ***p***s, with `00` representing ***mp***. Similarly `40` is ***mf*** and `41`–`4F` are 1–15 ***f***s. Above `4F` are various other marks such as ***mfp***, ***sfz***, etc.

`E1 ex (ii)` **Expression/Instruction Mark, above or below unspecified** *(Appendix I)*

The first data byte again refers to a table of available marks. If the high nybble is 4 (mordents and turns), there is an extra byte `ii` similar to that for the trill marks: the low nybble may have a value of 0–9 and represents an accidental (if any) to be added above the mark. This nybble works the same way as for note names. Bit 4 is set if the accidental is required below the mark, rather than above. This bit is ignored if there is no accidental.

`E2 ex (ii)` **Expression/Instruction Mark, above the music**

`E3 ex (ii)` **Expression/Instruction Mark, below the music**

`E6 st` **Structure Mark** *(Appendix J)*

This is a 𝄋 (`00`) or 𝄌 (`01`) mark.

`E8 nl` **Beat Repeat**

The data byte is the number of lines of which the mark consists (*eg* 2 for 16th-note figuration).

`EA nb` **Measure Repeat**

The number of measures to be repeated. Only 1 or 2 are conventional.

`EB tx cc` **Text Expression/Instruction, above or below unspecified** *(Appendix K)*

This is not a text event in the true sense, but a macro for compactness. The first data byte is a number in a table of the most commonly used text expressions. The second byte defines the presentation of the text: bits 0 and 1 determine the case with a value of `0`–`3`; bit 2 is set for italicisation and bit 3 is set for abbreviation—many of the text expressions in the table have a dot part-way through, at which point the text should finish if this bit is set. The dot should of course be omitted if this bit is clear. Naturally, any other text may be inserted with an EXPRESSION TEXT event (see below). Also, see the second paragraph of the EXPRESSION TEXT event description (`F8`).

[A further suggestion is that the TEXT EXPRESSION table in particular, and possibly some of the others, could be supplied in a file to developers for inclusion in their packages. Updated versions could be supplied whenever changes are seen to be necessary.]

`EC tx cc` **Text Expression/Instruction, above the music**

`ED tx cc` **Text Expression/Instruction, below the music**

`EF rh` **Auto Rehearsal Mark** *(Appendix L)*

The first time this event occurs (usually in the header chunk) there must be a value other than zero in the data byte. This value represents the type of rehearsal mark to be used. If bit 5 is set, this is taken to mean a measure number at intervals given by the value in bits 0–4 plus one (1–32). Otherwise, if bit 0 is set, the rehearsal marks will be numbers; if bit 1 is set they will be letters (bits 0 and 1 may not both be set); if bit 2 is set (which it may be with bits 0 or 1) there will be a measure number at the beginning of each system. In all the cases so far described, if bit 6 is set the marks will be shown in a box, otherwise they will not. After the first REHEARSAL MARK event, subsequent ones should have zero in the data byte in order to continue the sequence from the preceding mark, unless for some reason a change of style is desired. If the mark is of a type which produces entirely automatic placing (`04`, `20`–`3F`), this event is needed only once, at the beginning of the header chunk.

TEXT EVENTS

These are mainly distinguished by the way in which the importing program is expected to position them in the score, and only some of them are permitted in the header chunk. It is strongly recommended that the different types are implemented, rather than using GENERAL TEXT for everything. In every case, there is a variable-length quantity for the length of the text which follows.

[A further convention is suggested. In order to be able to incorporate musical symbols in the text (*eg Dal* 𝄋 *al* 𝄌), the non-printing ASCII zero character could be used as a "switch" to toggle between the two character sets—a standard character set would have to be agreed for music fonts, possibly something like the *Petrucci* font for the Macintosh. The data part of "*Dal* 𝄋 *al* 𝄌" would then become `45 61 6C 20 00 se 00 20 61 6C 20 00 ca 00` in the file, where `se` is the code for the 𝄋 character and `ca` the code for 𝄌. The final 00 should always be included unless the next text event is to begin in the music character set.]

`F0 len text` **General Text**

This may be anything, and will be placed near the current position, but, as indicated above, is only provided for those cases where none of the events below is appropriate.

`F2 len text` **Title/Identifier**

If found at the beginning of the header chunk, this will be taken as the title for the work to be notated, and should be displayed prominently on the first page, possibly centred, and in a large font. If later in the same chunk, it should be taken as a section heading or subtitle. More than one may be required (see the Mussorgsky example), in which case they should be placed in the file in order of priority, *eg* title, then subtitle, etc.

If found in the music chunk, it will be taken as a staff identifier which may change during the course of the piece (*eg* from Clarinet to Bass Clarinet).

`F4 len text` **Chord Symbol/Fingering Mark**

This should be positioned exactly above the current position. If it consists merely of a number, it should naturally be read as a fingering mark.

`F6 len text` **Lyric**

This should be positioned exactly below the current position. If there is already a lyric event at the same position, it should be assumed that this is a further verse and it should be placed below the previous lyric—it is up to the exporting program to ensure that the lyrics for different verses arrive in the correct order.

`F8 len text` **Expression Text**

This should also be positioned fairly accurately, though accuracy is not so crucial as for chord symbols or lyrics.

In the case of exceptionally long expression texts (see the Mussorgsky example), it may be necessary to break it into several so that the receiving software can handle it. A suggested convention is that an expression string which ends with a space (`20`) should imply that another is to be tacked onto the end of it.

`FA len text` **Rehearsal Mark**

This is for a non-automatic mark (or section heading) and should be positioned above the beginning of a measure.

Other Events

`FC len text` **Screen Message**

This is a special event which many packages may prefer to ignore. It allows a message to be displayed on the screen at any point in the course of reading the file. This may be a progress report, a copyright notice or whatever is deemed appropriate. The syntax works exactly the same way as for text events. How, and whether, it is implemented is entirely a matter for programmers.

`FE len data` **Software-specific Event**

Part quoting, part paraphrasing from the MIDI File 1.0 specification: "Special requirements for particular programs may use this event type: the first byte or bytes of data is a manufacturer ID (one byte or, if the first byte is `00`, three bytes). Manufacturers who define something using this event should publish it so that others may know how to use it. This type of event may also be used by software which elects to use this as its only file format." The last sentence of that is probably the most relevant.

`FF` **End of Chunk**

This must be present, and must be the last event, in both the header and music chunks, even if it is the only event in the chunk.

Under Consideration

In the light of manually encoding the three music examples at the end of this document, three further possibilities might be worth considering for the sake of file compactness:

1. A single event which the importing software would interpret as "repeat everything that has happened in the last <length>". This length might be a single note (8th or whatever) with any accompanying expression marks, etc.; or it might be an entire measure or more.
2. A single event which instructs the program to "place this mark (*staccato* dot or *tenuto* dash would probably be the most common—see the Sansom and Mussorgsky examples) on every note or chord which occurs in the next <length>".
3. It may be desirable to use an equivalent to Running Status as used in MIDI Files, though its use would probably require some rather illogical ordering of events—this will be apparent from examination of the three examples, where consecutive instances of the same event type are relatively uncommon.

The feasibility of these will be investigated if interest in the ***NotaFile*** idea is detected among the music software fraternity.

Chris Sansom, 21 May 1992

Revised 16 May 1993

Very minor revisions 6 August 2014

APPENDICES: REFERENCE TABLES

Where an entry is left blank, that value is so far undefined.

APPENDIX A

DEFINING BYTES (*YES/NO* refers to permissibility of event in header chunk)

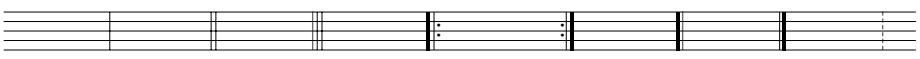
80	Change of Measure	*YES*	C0	Tempo	*YES*
81			C1		
82			C2	Time Signature	*YES*
83			C3		
84	Change of Position in Measure	*YES*	C4	Key Signature (normal)	*YES*
85			C5		
86			C6	Key Signature (literal)	*YES*
87			C7		
88	1st Time Bar	*YES*	C8	Clef	*NO*
89	2nd Time Bar	*YES*	C9		
8A			CA	Transposition	*NO*
8B			CB		
8C	Barline type	*YES*	CC	Number of Staff Lines	*NO*
8D	Change of Staff	*NO*	CD	Special Noteheads	*NO*
8E	Change of Voice/Stem Direction	*NO*	CE	Number of Voices per Staff	*NO*
8F	2-Note/Chord *Tremolando*	*NO*	CF		
90	Note, octave 0	*NO*	D0	Note in Chord, octave 0	*NO*
91	Note, octave 1	*NO*	D1	Note in Chord, octave 1	*NO*
92	Note, octave 2	*NO*	D2	Note in Chord, octave 2	*NO*
93	Note, octave 3	*NO*	D3	Note in Chord, octave 3	*NO*
94	Note, octave 4	*NO*	D4	Note in Chord, octave 4	*NO*
95	Note, octave 5	*NO*	D5	Note in Chord, octave 5	*NO*
96	Note, octave 6	*NO*	D6	Note in Chord, octave 6	*NO*
97	Note, octave 7	*NO*	D7	Note in Chord, octave 7	*NO*
98	Note, octave 8 (starts at middle C)	*NO*	D8	Note in Chord, octave 8	*NO*
99	Note, octave 9	*NO*	D9	Note in Chord, octave 9	*NO*
9A	Note, octave 10	*NO*	DA	Note in Chord, octave 10	*NO*
9B	Note, octave 11	*NO*	DB	Note in Chord, octave 11	*NO*
9C	Note, octave 12	*NO*	DC	Note in Chord, octave 12	*NO*
9D	Note, octave 13	*NO*	DD	Note in Chord, octave 13	*NO*
9E	Note, octave 14	*NO*	DE	Note in Chord, octave 14	*NO*
9F	Note, octave 15	*NO*	DF	Note in Chord, octave 15	*NO*
A0	Rest	*NO*	E0	Dynamic Mark	*NO*
A1	Slur over/under	*NO*	E1	Expression/Instruction over/under	*YES*
A2	Slur over	*NO*	E2	Expression/Instruction over	*YES*

A3	Slur under	*NO*
A4	*Diminuendo* hairpin	*NO*
A5	*Crescendo* hairpin	*NO*
A6		
A7		
A8	*8va*	*NO*
A9	*8va bassa*	*NO*
AA	*15ma*	*NO*
AB	*15ma bassa*	*NO*
AC	Trill over/under	*NO*
AD	Trill over	*NO*
AE	Trill under	*NO*
AF		
B0	Bracket over/under	*NO*
B1	Bracket over	*NO*
B2	Bracket under	*NO*
B3		
B4	House over/under	*YES*
B5	House over	*YES*
B6	House under	*YES*
B7		
B8	Pedal 1 start	*NO*
B9	Pedal 1 continue	*NO*
BA	Pedal 2	*NO*
BB		
BC		
BD	Line	*YES*
BE	Line – – – – – – –	*YES*
BF	Line ———————	*YES*

E3	Expression/Instruction under	*NO*
E4		
E5		
E6	Structure Mark	*YES*
E7		
E8	Beat Repeat	*NO*
E9		
EA	Measure Repeat	*NO*
EB	Text Expression over/under	*YES*
EC	Text Expression over	*YES*
ED	Text Expression under	*NO*
EE		
EF	Auto Rehearsal Mark	*YES*
F0	General Text	*YES*
F1		
F2	Title/Identifier	*YES*
F3		
F4	Chord Symbol/Fingering Mark	*NO*
F5		
F6	Lyric	*NO*
F7		
F8	Expression Text	*YES*
F9		
FA	Rehearsal Mark (non-auto)	*YES*
FB		
FC	Screen Message	*YES*
FD		
FE	Software-specific Event	*YES*
FF	End of Chunk	*YES*

APPENDIX B

BARLINE TYPES (8C)



00	Normal
01	
02	Double bar (thin, thin)
03	Triple bar (thin, thin, thin)
04	Start repeat (thick, thin, 2 dots)
05	
06	End repeat (2 dots, thin, thick)
07	
08	Start double (thick, thin)
09	
0A	End double (thin, thick)
0B	
0C	Dotted
0D–7F	

APPENDIX C

NOTE DATA (9n, Dn, 8F)

00	C
01	C ♮
02	C ♭
03	C ♯
04	C ♭♭
05	C ×
06	C quarter-tone ♭
07	C quarter-tone ♯
08	C three-quarter-tone ♭
09	C three-quarter-tone ♯
0A–0F	
10	D
11	D ♮
12	D ♭
13	D ♯
14	D ♭♭
15	D ×
16	D quarter-tone ♭
17	D quarter-tone ♯
18	D three-quarter-tone ♭
19	D three-quarter-tone ♯
1A–1F	
20	E
21	E ♮
22	E ♭
23	E ♯
24	E ♭♭
25	E ×
26	E quarter-tone ♭
27	E quarter-tone ♯
28	E three-quarter-tone ♭
29	E three-quarter-tone ♯
2A–2F	
30	F
31	F ♮
32	F ♭
33	F ♯
34	F ♭♭
35	F ×
36	F quarter-tone ♭
37	F quarter-tone ♯
38	F three-quarter-tone ♭
39	F three-quarter-tone ♯
3A–3F	
40	G
41	G ♮
42	G ♭
43	G ♯
44	G ♭♭
45	G ×
46	G quarter-tone ♭
47	G quarter-tone ♯
48	G three-quarter-tone ♭
49	G three-quarter-tone ♯
4A–4F	
50	A
51	A ♮
52	A ♭
53	A ♯
54	A ♭♭
55	A ×
56	A quarter-tone ♭
57	A quarter-tone ♯
58	A three-quarter-tone ♭
59	A three-quarter-tone ♯
5A–5F	
60	B
61	B ♮
62	B ♭
63	B ♯
64	B ♭♭
65	B ×
66	B quarter-tone ♭
67	B quarter-tone ♯
68	B three-quarter-tone ♭
69	B three-quarter-tone ♯
6A–7F	

High nybble:

0	C
1	D
2	E
3	F
4	G
5	A
6	B

Low nybble:

0	no accidental
1	♮
2	♭
3	♯
4	𝄫
5	×
6	quarter tone ♭
7	quarter tone ♯
8	three quarter tone ♭
9	three quarter tone ♯

Supplementary data byte 1 (flags)

Bit 7	always 0
Bit 6	start of *glissando*—further data follows
Bit 5	special notehead—1 further byte follows
Bit 4	*tremolando*—1 further byte follows
Bit 3	tied before (allowed with bit 2)
Bit 2	tied after (allowed with bit 3)
Bit 1	grace with stroke (not with bit 0)
Bit 0	grace without stroke (not with bit 1)

Supplementary data byte 2 (flags)

Bit 7	brackets round accidental only (not with bit 6)
Bit 6	brackets (not with bit 7)
Bit 5	last of tuplet group (not with bit 4)
Bit 4	first of tuplet group (not with bit 5)
Bit 3	last of beamed group (not with bit 2, 1 or 0)
Bit 2	first of sub-group (not with bit 3, 1 or 0)
Bit 1	last of sub-group (not with bit 3, 2 or 0)
Bit 0	first of beamed group (not with bit 3, 2 or 1)

APPENDIX D

RESTS: SUPPLEMENTARY DATA BYTE (FLAGS) (A0)

Bit 7	always 0
Bit 6	brackets
Bit 5	last of tuplet group (not with bit 4)
Bit 4	first of tuplet group (not with bit 5)
Bit 3	last of beamed group (not with bit 2, 1 or 0)
Bit 2	first of sub-group (not with bit 3, 1 or 0)
Bit 1	last of sub-group (not with bit 3, 2 or 0)
Bit 0	first of beamed group (not with bit 3, 2 or 1)

APPENDIX E

KEY SIGNATURES (CONVENTIONAL) (C4)

00	C major/A minor		
12	1 flat	13	1 sharp
22	2 flats	23	2 sharps
32	3 flats	33	3 sharps
42	4 flats	43	4 sharps
52	5 flats	53	5 sharps
62	6 flats	63	6 sharps
72	7 flats	73	7 sharps

APPENDIX F

CLEFS (C8)

Large		*Small*
00	treble	40
01	treble, 8 above	41
02	treble, 8 below	42
03	treble, 15 above	43
04	treble, 15 below	44
07	G clef, any position	47
10	bass	50
11	bass, 8 above	51
12	bass, 8 below	52
13	bass, 15 above	53
14	bass, 15 below	54
17	F clef, any position	57
20	alto	60
21	tenor	61
22	soprano, modern	62
27	C clef, any position	67
2C	soprano, archaic	6C
30	percussion 1	70
31	percussion 2	71
32	percussion 3	72
7F	no clef	

Percussion 1: 2 thick vertical lines covering middle 2 spaces of staff

Percussion 2: small square bracket across middle line

Percussion 3: narrow vertical open rectangle

APPENDIX G

SPECIAL NOTEHEADS (CD)

Large		*Small*
00	normal	40
01	all open	41
02	all filled	42
03	diamond	43
04	open diamond	44
05	cross	45
06	square	46
07	open square	47
08	triangle, apex up	48
09	triangle, apex down	49
0A	open triangle, apex up	4A
0B	open triangle, apex down	4B
0F	no head	4F
10–1F	tablature, numbers 0–15	50–5F
20–3F	as above, without stem	60–7F
(2F	undefined	6F)

APPENDIX H

DYNAMIC MARKS (E0)

00	***mp***	40	***mf***
01	***p***	41	***f***
02	***pp***	42	***ff***
03	***ppp***	43	***fff***
:		:	
0F	***ppppppppppppppp***	4F	***fffffffffffffff***
50	***mfp***	5C	
51	***fp***	5D	***sfz***
52	***ffp***	5E	***sffz***
53	***fffp***	5F	***sfffz***
54		60	
55	***fz***	61	***sfp***
56	***ffz***	62	***sffp***
57	***fffz***	63	***sfffp***
58		64	***sfpp***
59	***sf***	65	***sffpp***
5A	***sff***	66	***sfppp***
5B	***sfff***	67	***sffppp***

APPENDIX I

EXPRESSION/INSTRUCTION MARKS (`E1`, `E2`, `E3`)

00	𝄐	*fermata*, round *	40	𝆔	turn
01		*fermata*, square *	41		
02	𝆛	pause mark, angled *	42	𝆕	inverted turn
03	G.P.		43		
04–07			44–47		
08	’	breath	48	𝆖	mordent
09			49	𝆗	lower mordent
0A			4A	𝆘	double mordent
0B			4B	𝆙	lower double mordent
0C–0F			4C–4F		
10	>	accent	50		
11	-	*tenuto*	51		
12	𝆛	accent *	52		
13–17			53–57		
18	·	*staccato*	58		
19	𝆟	*staccato* triangle *	59		
1A–1F			5A–5F		
20	𝆭	down bow	60		
21	𝆮	up bow	61		
22–27			62–67		
28	○	open or harmonic	68		
29	+	closed or muted	69		
2A–2B			6A–6B		
2C	𝆹	Bartók *pizzicato*	6C		
2D–2F			6D–6F		
30–3F			70–7F	miscellaneous marks	

* These marks should be inverted by the importing program if the event is BELOW THE MUSIC (`E3`).

APPENDIX J

STRUCTURE MARKS (`E6`)

`00`	𝄋
`01`	𝄌

APPENDIX K

TEXT EXPRESSION MARKS (EB, EC, ED)

00	*a piacere*
01	*ad lib*
02	*appassionato*
03	*arp.eggiato*
04	*cant.abile*
05	*colla parte*
06	*come sopra*
07	*cresc.endo*
08	*dim.inuendo*
09	*dolce*
0A	*energico*
0B	*espr.essivo*
0C	*flaut.ando*
0D	*gliss.ando*
0E	*graz.ioso*
0F	*leg.ato*
10	*legg.iero*
11	*marc.ato*
12	*lontano*
13	*morendo*
14	*pesante*
15	*rall.entando*
16	*rapido*
17	*rit.ardando*
18	*rubato*
19	*secco*
1A	*semplice*
1B	*sim.ile*
1C	*soli*
1D	*solo*
1E	*sost.enuto*
1F	*sotto voce*
20	*stacc.ato*
21	*stretto*
22	*sub.ito*
23	*ten.uto*
24	*tutta forza*
25	*tutte*
26	*tutti*
27	*unis.ono*
28	*vivo*
29	
2A	
2B	*adagio*
40	*presto*
41	*tranquillo*
42	*trio*
43	*vivace*
44	
45	
46	*a tempo*
47	*accel.erando*
48	*allarg.ando*
49	*attacca*
4A	*l'istesso tempo*
4B	*string.endo*
4C	*tempo I*
4D	
4E	
4F	*al*
50	*al fine*
51	*al segno*
52	*d.c.*
53	*da capo*
54	
55	
56	*arco*
57	*col legno*
58	*div.isi*
59	*pizz.icato*
5A	*sul pont.icello*
5B	*sul tasto*
5C	
5D	
5E	*con sord.ino*
5F	*cuivrez*
60	*fltttzg.*
61	*muta in*
62	*ouvert*
63	*pavillon en l'air*
64	*pavillons en l'air*
65	*senza sord.ino*
66	
67	
68	*una corda*
69	
6A	
6B	*assai*

2C	*alla marcia*	6C	*col*
2D	*alla tedesca*	6D	*con*
2E	*allegretto*	6E	*dal*
2F	*allegro*	6F	*e*
30	*andante*	70	*ma*
31	*brio*	71	*meno*
32	*calmo*	72	*molto*
33	*fuga*	73	*non*
34	*grave*	74	*più*
35	*larghetto*	75	*poco*
36	*largo*	76	*poco a poco*
37	*lento*	77	*sempre*
38	*ma non tanto*	78	*senza*
39	*ma non troppo*	79	*sul*
3A	*maestoso*	7A	*tanto*
3B	*menuetto*	7B	*troppo*
3C	*moderato*	7C	
3D	*mosso*	7D	
3E	*moto*	7E	
3F	*prestissimo*	7F	

Supplementary data byte

Bit 3 set for abbreviation

Bit 2 set for italics

Bit 0, 1
- 0 all lower case
- 1 ALL UPPER CASE
- 2 Sentence case
- 3 Title Case

APPENDIX L

AUTO REHEARSAL MARKS (EF)

Not boxed		*Boxed*
00	same as previous	40
01	numbers only	41
02	letters only	42
04	measure numbers start of each system	44
05	numbers plus measure numbers start of each system	45
06	letters plus measure numbers start of each system	46
20	measure number every measure	60
21	measure number every 2nd measure	61
22	measure number every 3rd measure	62
.		.
3F	measure number every 32nd measure	7F

MUSIC EXAMPLES

This final section contains three extracts of "real" music, each with a possible (and fully commented) ***NotaFile*** listing. The use of the word "possible" is important here—it should be remembered that there may be many different orders in which any given piece of music might be encoded.

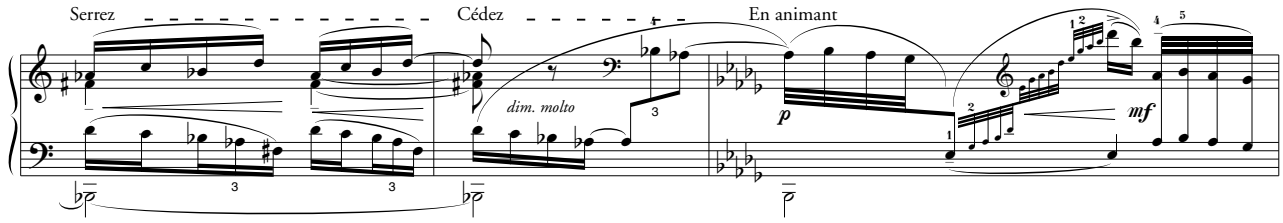
The purpose of this section is principally to demonstrate that the whole idea is feasible. The extracts are in contrasting styles and for different instrumental forces (piano, saxophone quartet and orchestra).

DEBUSSY: *VOILES* (NO. II FROM *PRÉLUDES, 1ER LIVRE*) MEASURES 40–42

This was deliberately chosen for the complexity and detail contained in a small extract of fairly simple-sounding piano music. None of the "setting up" events are listed here—this is intended to be as found in the middle of the file.

Inevitably, decisions will have to be made about where to draw the line between the musical material which would be included in a ***NotaFile*** and layout material which would not. There is just such an instance in the last bar of this extract, where the beaming of the run of grace notes is "tacked on" to the eighth note stem at the end of the first beat. This is, in any case, highly unorthodox notation, so it has been decided to ignore it and pretend that the grace notes are beamed together independently.

The order of events in the music chunk is governed principally by phrase and beam groupings: this seems the most logical approach here. Although it may not be strictly necessary in every case, all the slurs in this example are specified as "above" in view of the meticulous detail which seems to be required.



51 bytes in the header chunk …

`80 28`	measure 40
`F8 06 53 65 72 72 65 7A`	expression text "Serrez"
`84 02 10 00`	next 16th
`BE 02 04 02`	dashed line, double dotted quarter note long
`80 29`	measure 41
`F8 05 43 8E 64 65 7A`	expression text "Cédez" (é is 8E in the Macintosh character set)
`84 02 10 00`	next 16th
`BE 02 04 02`	dashed line, as above
`80 2A`	measure 42
`C4 52`	key signature 5 flats
`F8 0A 45 6E 20 61 6E 69 6D 61 6E 74`	expression text "En animant"

775 bytes in the music chunk …

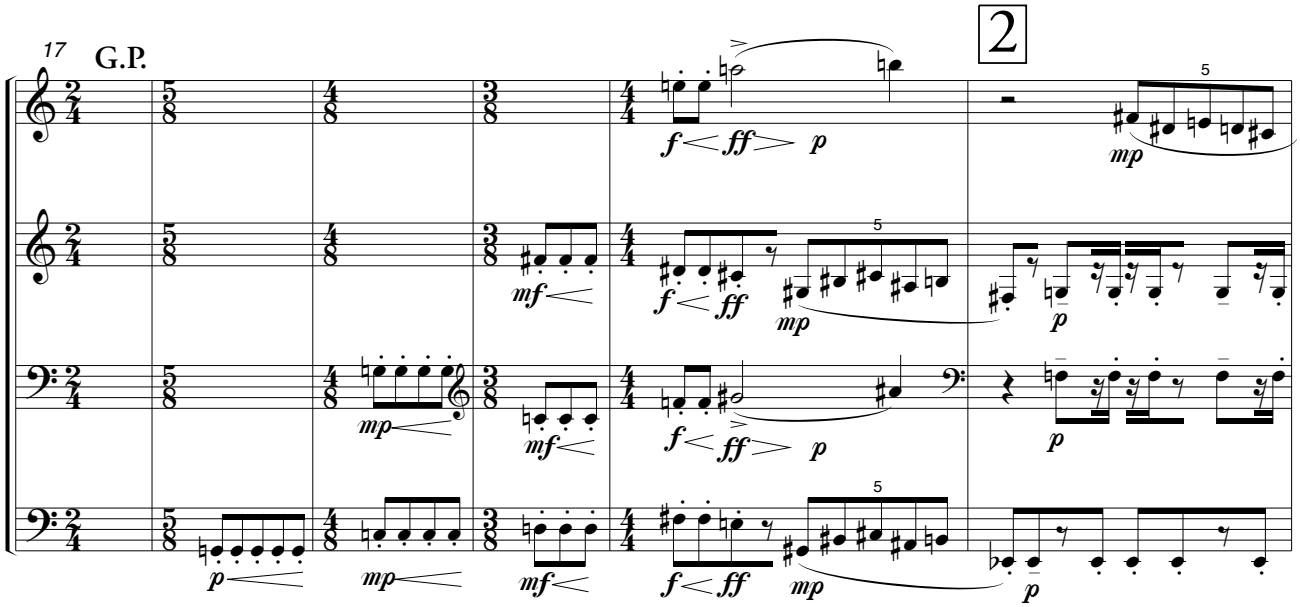
`80 28`	measure 40
`8D 00 02`	staff 2
`8E 20`	voice 2 (already in 2 voices)
`95 62 02 00 0C 00`	B♭5 half note, tied both ways
`80 29`	measure 41
`95 62 02 00 08 00`	B♭5 half note, tied before
`80 28`	back to measure 40
`8E 11`	voice 1, stems down
`98 10 10 00 00 01`	D8 16th, start of beamed group
`84 02 10 00`	next 16th
`98 00 10 00 00 02`	C8 16th, end of sub-group
`84 02 08 00`	next 16th
`97 62 18 00 00 14`	B♭7 triplet 16th, start of tuplet and sub-group
`84 04 08 00 18 00`	next triplet 16th
`97 52 18 00 00 00`	A♭7 triplet 16th
`84 04 08 00 0C 00`	next triplet 16th
`97 33 18 00 00 28`	F♯7 triplet 16th, end of tuplet and group
`84 02 00 00`	back to first beat
`A2 00 00 02 04 00`	slur above, quarter note long
`84 02 04 00`	second beat
`98 10 10 00 00 01`	D8 16th, start of group
`84 04 04 00 10 00`	next 16th
`98 00 10 00 00 02`	C8 16th, end of sub-group
`84 02 04 01`	next 16th
`97 60 18 00 00 14`	B7 triplet 16th, start of tuplet and sub-group (b not needed)
`84 04 04 01 18 00`	next triplet 16th
`97 50 18 00 00 00`	A7 triplet 16th
`84 04 04 01 0C 00`	next triplet 16th
`97 30 18 00 00 28`	F7 triplet 16th, end of tuplet and group
`84 02 04 00`	back to second beat
`A2 00 00 02 04 00`	slur above, quarter note long
`8D 00 01`	staff 1
`8E 20`	voice 2 (already in 2 voices)
`98 33 04 00 00 00`	F♯8 quarter
`E3 11`	*tenuto* dash below
`A5 02 04 00`	*crescendo* hairpin, quarter note long
`8E 10`	voice 1
`98 52 10 00 00 01`	A♭8 16th, start of group
`84 02 10 00`	next 16th
`99 00 10 00 00 00`	C9 16th
`84 02 08 00`	next 16th
`98 62 10 00 00 00`	B♭8 16th
`84 02 08 01`	next 16th
`99 10 10 00 00 08`	D9 16th, end of group
`84 02 00 00`	back to first beat
`A2 00 00 02 04 00`	slur above, quarter note long
`84 02 04 00`	second beat
`8E 20`	voice 2
`98 30 04 00 04 00`	F8 quarter, tied after (# not needed)
`E3 11`	tenuto dash below
`A5 02 04 00`	crescendo hairpin, quarter note long
`8E 10`	voice 1
`98 50 10 00 04 01`	A8 16th, start of group, tied after
`84 04 04 00 10 00`	next 16th
`99 00 10 00 00 00`	C9 16th
`84 02 04 01`	next 16th
`98 60 10 00 00 00`	B8 16th
`84 02 04 02`	next 16th
`99 10 10 00 04 08`	D9 16th, end of group, tied after
`84 02 04 00`	back to second beat
`A2 00 00 02 04 00`	slur above, quarter note long
`80 29`	measure 41
`99 10 08 00 08 00`	D9 8th, tied before
`8E 20`	voice 2
`98 52 08 00 08 00`	A♭8 8th, tied before
`D8 33 08`	add F♯8 to chord
`84 02 08 00`	next 8th
`CE 01`	1 voice on staff
`A0 08 00 00`	8th rest
`ED 08 0C`	text expression below "*dim.*"
`ED 72 08`	text expression below "*molto*"
`84 02 04 00`	next 8th
`C8 50`	bass clef
`84 02 00 00`	back to first beat
`8D 00 02`	staff 2
`8E 11`	voice 1, stems *down*
`98 10 10 00 00 01`	D8 16th, start of group
`84 02 10 00`	next 16th
`98 00 10 00 00 00`	C8 16th
`84 02 08 00`	next 16th
`97 62 10 00 00 00`	B♭7 16th
`84 02 08 01`	next 16th
`97 52 10 00 04 08`	A♭7 16th, end of group, tied after
`84 02 04 00`	second beat
`8E 13`	voice 1, stems *up*
`97 50 0C 00 08 11`	A7 triplet 8th, start of group and tuplet, tied before
`84 04 04 00 0C 00`	next triplet 8th
`8D 00 01`	staff 1
`97 62 0C 00 00 00`	B♭7 triplet 8th
`F4 01 34`	fingering "4"
`84 04 04 00 06 00`	next triplet 8th

```
97 52 0C 00 04 28      A♭7 triplet 8th, end of group and tuplet, tied after
84 02 00 00            back to first beat
8D 00 02               staff 2
8E 10                  voice 1
A2 00 01 04 02 00 20 00  slur above to staff 1, half + 32nd long
80 2A                  measure 42
8D 00 01               staff 1
E0 01                  dynamic mark p
97 50 20 00 08 01      A7 32nd, start of group, tied before
84 02 20 00            next 32nd
97 60 20 00 00 00      B7 32nd
84 02 10 00            next 32nd
97 50 20 00 00 00      A7 32nd
84 02 10 01            next 32nd
97 40 20 00 00 02      G7 32nd, end of sub-group
84 02 08 00            second 8th
8D 00 02               staff 2
CE 02                  2 voices
8E 10                  voice 1
97 20 08 00 04 08      E7 8th, end of group, tied after
F4 01 31               fingering "1"
84 02 00 00            back to first beat
8D 00 01               staff 1
A2 00 02 02 04 00      slur above to staff 2, quarter note long
8D 00 02               staff 2
8E 10                  voice 1
84 02 04 00            second beat
97 40 20 00 01 01      G7 32nd grace note, start of group
F4 01 32               fingering "2"
97 50 20 00 01 00      A7 32nd grace
97 60 20 00 01 00      B7 32nd grace
98 10 20 00 01 02      D8 32nd grace, end of sub-group
8D 00 01               staff 1
C8 40                  treble clef
A5 02 08 00            crescendo hairpin, 8th long
98 20 20 00 01 04      E8 32nd grace, start of sub-group
98 40 20 00 01 00      G8 32nd grace
98 50 20 00 01 00      A8 32nd grace
98 60 20 00 01 00      B8 32nd grace
99 10 20 00 01 08      D9 32nd grace, end of group
99 20 20 00 01 01      E9 32nd grace, start of group
F4 01 31               fingering "1"
99 40 20 00 01 00      G9 32nd grace
F4 01 32               fingering "2"
99 50 20 00 01 00      A9 32nd grace
99 60 20 00 01 08      B9 32nd grace, end of group
9A 10 10 00 00 01      D10 16th, start of group
84 04 04 00 10 00      next 16th
99 60 10 00 00 08      B9 16th, end of group
84 02 04 00            back to second beat
E2 10                  accent mark above
A2 00 00 02 08 00      slur above, 8th long
84 02 04 01            last 8th
E2 11                  tenuto dash above
F4 01 34               fingering "4"
E0 40                  dynamic mark mf
98 50 20 00 00 01      A8 32nd, start of group
8D 00 02               staff 2
D7 50 00               add A7 to chord
84 04 04 01 20 00      next 32nd
8D 00 01               staff 1
98 60 20 00 00 00      B8 32nd
F4 01 35               fingering "5"
8D 00 02               staff 2
D7 60 00               add B7 to chord
84 02 04 02            next 32nd
8D 00 01               staff 1
98 50 20 00 00 00      A8 32nd
8D 00 02               staff 2
D7 50 00               add A7 to chord
84 02 04 03            next 32nd (after triple dotted quarter)
8D 00 01               staff 1
98 40 20 00 00 08      G8 32nd, end of group
8D 00 02               staff 2
D7 40 00               add G7 to chord
84 02 04 01            back to last 8th
8D 00 01               staff 1
A2 00 00 02 04 00      slur above, quarter long
84 02 08 00            back to second 8th
8D 00 02               staff 2
8E 10                  voice 1
A2 00 01 02 04 00      slur above to staff 1, quarter long
84 02 04 00            second beat
97 20 04 00 80 00      E7 quarter, tied before
84 02 00 00            back to first beat
8E 20                  voice 2
95 60 02 00 00 00      B5 half note
```

CHRIS SANSOM: *ANTICIPATION* (NO. 1 FROM *FIVE PIECES FOR SAXOPHONE QUARTET*) MEASURES 17–22

This extract, by the author of this document, was chosen to avoid copyright problems with any other contemporary music. There are three quintuplet groups and frequent changes of time signature, and the notation adheres to the convention established by the "Second Viennese School" (Schoenberg, Berg and Webern) for highly chromatic music: that of placing an accidental before *every* note except for immediate repetitions. Two other modern usages appear here—rests included in beamed groups, and whole measure rests omitted from empty measures. All this is catered for in the code.

As this music is in four discrete parts, it is probably most logical to take a "horizontal" approach and order the music chunk sequentially staff by staff, as if in "tracks". The example therefore consists of an extract from the header chunk and four separate extracts from the music chunk.



31 bytes in the header chunk …

`80 11`	measure 17
`C2 02 04`	2/4 time signature
`E2 03`	expression mark "G.P."
`80 12`	measure 18
`C2 05 08`	5/8
`80 13`	measure 19
`C2 04 08`	4/8
`80 14`	measure 20
`C2 03 08`	3/8
`80 15`	measure 21
`C2 04 04`	4/4
`80 16`	measure 22
`EF 00`	auto rehearsal mark (previously set up as boxed numbers plus measure numbers at start of each system)

144 bytes in the Soprano saxophone part of the music chunk …

`80 15`	measure 21 (already on staff 1)
`99 21 08 00 00 01`	E♮9 8th, start of beamed group
`E1 18`	*staccato* dot
`E0 41`	dynamic ***f***
`A5 02 04 00`	*crescendo* hairpin, quarter long
`84 02 08 00`	next 8th
`99 20 08 00 00 08`	E9 8th, end of group
`E1 18`	*staccato* dot
`84 02 04 00`	second beat
`99 51 02 00 00 00`	A♮9 half note
`A1 00 00 02 02 01`	slur, dotted half note long
`E1 10`	accent
`E0 42`	dynamic ***ff***
`A4 02 04 00`	*diminuendo* hairpin, quarter long
`84 02 02 00`	third beat
`E0 01`	dynamic ***p*** (NB this places the mark half-way through the half note)
`84 02 02 01`	last beat
`99 61 04 00 00 00`	B♮9 quarter
`80 16`	measure 22
`A0 02 00 00`	half note rest
`84 02 02 00`	third beat
`98 33 0A 00 00 01`	F♯8 quintuplet 8th, start of group
`A1 00 00 04 02 00 08 01`	slur, half plus dotted 8th long (there is a dotted eighth note at the end of the phrase, after the extract shown)
`E0 00`	dynamic ***mp***
`84 04 02 00 0A 00`	next quintuplet 8th
`98 13 0A 00 00 00`	D♯8 quintuplet 8th
`84 04 02 00 05 00`	next quintuplet 8th
`98 21 0A 00 00 00`	E♮8 quintuplet 8th
`84 04 02 01 14 00`	next quintuplet 8th (after dotted half plus 20th)
`98 11 0A 00 00 00`	D♮8 quintuplet 8th
`84 06 02 00 05 00 05 00`	last quintuplet 8th
`98 03 0A 00 00 08`	C♯8 quintuplet 8th, end of group

276 bytes in the Alto saxophone part of the music chunk …

`80 14`	measure 20 (already on staff 2)
`98 33 08 00 00 01`	F♯8 8th, start of group
`E1 18`	*staccato* dot
`E0 40`	dynamic ***mf***
`A5 02 04 01`	*crescendo* hairpin, dotted quarter long
`84 02 08 00`	next 8th
`98 30 08 00 00 00`	F8 8th
`E1 18`	*staccato* dot
`84 02 04 00`	next 8th
`98 30 08 00 00 08`	F8 8th, end of group
`E1 18`	*staccato* dot
`80 15`	measure 21
`98 13 08 00 00 01`	D♯8 8th, start of beamed group
`E1 18`	*staccato* dot
`E0 41`	dynamic ***f***
`A5 02 04 00`	*crescendo* hairpin, quarter long
`84 02 08 00`	next 8th
`98 10 08 00 00 00`	D8 8th
`E1 18`	*staccato* dot
`84 02 04 00`	second beat
`98 03 08 00 00 00`	C♯8 8th
`E1 18`	*staccato* dot
`E0 42`	dynamic ***ff***
`84 02 04 01`	next 8th
`A0 08 00 08`	8th rest, end of group
`84 02 02 00`	third beat
`97 43 0A 00 00 01`	G♯7 quintuplet 8th, start of group
`E0 00`	dynamic ***mp***
`A1 00 00 04 02 00 08 00`	slur, half plus 8th long
`84 04 02 00 0A 00`	next quintuplet 8th
`97 63 0A 00 00 00`	B♯7 quintuplet 8th
`84 04 02 00 05 00`	next quintuplet 8th
`98 03 0A 00 00 00`	C♯8 quintuplet 8th
`84 04 02 01 14 00`	next quintuplet 8th
`97 53 0A 00 00 00`	A♯7 quintuplet 8th
`84 06 02 00 05 00 05 00`	last quintuplet 8th
`97 61 0A 00 00 08`	B♮7 quintuplet 8th, end of group
`80 16`	measure 22
`97 33 08 00 00 01`	F♯7 8th, start of group
`E1 18`	*staccato* dot
`84 02 08 00`	next 8th
`A0 08 00 08`	8th rest, end of group

```
84 02 04 00          second beat
97 41 08 00 00 01    G♮7 8th, start of group
E1 11                tenuto dash
E0 01                dynamic p
84 02 04 01          next 8th
A0 10 00 04          16th rest, start of sub-group
84 02 04 02          next 16th
97 40 10 00 00 08    G7 16th, end of group
E1 18                staccato dot
84 02 02 00          third best
A0 10 00 01          16th rest, start of group
84 04 02 00 10 00    next 16th
97 40 10 00 00 02    G7 16th, end of sub-group
E1 18                staccato dot
84 04 02 00 08 00    next 16th
A0 08 00 08          8th rest, end of group
84 02 02 01          last beat
97 40 08 00 00 01    G7 8th, start of group
E1 11                tenuto dash
84 02 02 02          next 8th
A0 10 00 04          16th rest, start of sub-group
84 02 02 03          next 16th
97 40 10 00 00 08    G7 16th, end of group
E1 18                staccato dot
```

268 bytes in the Tenor saxophone part of the music chunk …

```
80 13                measure 19 (already on staff 3)
97 41 08 00 00 01    G♮7 8th, start of group
E1 18                staccato dot
E0 00                dynamic mp
A5 02 02 00          crescendo hairpin, half note long
84 02 08 00          next 8th
97 40 08 00 00 00    G7 8th
E1 18                staccato dot
84 02 04 00          next 8th
97 40 08 00 00 00    G7 8th
E1 18                staccato dot
84 02 04 01          next 8th
97 40 08 00 00 08    G7 8th, end of group
E1 18                staccato dot
80 14                measure 20
C8 40                treble clef
98 01 08 00 00 01    C♮8 8th, start of group
E1 18                staccato dot
E0 40                dynamic mf
A5 02 04 01          crescendo hairpin, dotted quarter long
84 02 08 00          next 8th
98 00 08 00 00 00    C8 8th
E1 18                staccato dot
84 02 04 00          next 8th
98 00 08 00 00 08    C8 8th, end of group
E1 18                staccato dot
80 15                measure 21
98 31 08 00 00 01    F♮8 8th, start of group
E1 18                staccato dot
E0 41                dynamic f
A5 02 04 00          crescendo hairpin, quarter long
84 02 08 00          next 8th
98 30 08 00 00 08    F8 8th, end of group
E1 18                staccato dot
84 02 04 00          second beat
98 43 02 00 00 00    G♯8 half note
A1 00 00 02 02 01    slur, dotted half note long
E1 10                accent
E0 42                dynamic ff
A4 02 04 00          diminuendo hairpin, quarter long
84 02 02 00          third beat
E0 01                dynamic p
84 02 02 01          last beat
98 53 04 00 00 00    A♯8 quarter
80 16                measure 22
C8 50                bass clef
A0 04 00 00          quarter note rest
84 02 04 00          second beat
97 31 08 00 00 01    F♮7 8th, start of group
E1 11                tenuto dash
E0 01                dynamic p
84 02 04 01          next 8th
A0 10 00 04          16th rest, start of sub-group
84 02 04 02          next 16th
97 30 10 00 00 08    F7 16th, end of group
E1 18                staccato dot
84 02 02 00          third beat
A0 10 00 01          16th rest, start of group
84 04 02 00 10 00    next 16th
97 30 10 00 00 02    F7 16th, end of sub-group
E1 18                staccato dot
84 04 02 00 08 00    next 16th
A0 08 00 08          8th rest, end of group
84 02 02 01          last beat
97 30 08 00 00 01    F7 8th, start of group
E1 11                tenuto dash
84 02 02 02          next 8th
A0 10 00 04          16th rest, start of sub-group
84 02 02 03          next 16th
97 30 10 00 00 08    F7 16th, end of group
E1 18                staccato dot
```

368 bytes in the Baritone saxophone part of the music chunk …

`80 12`	measure 18 (already on staff 4)
`96 41 08 00 00 01`	G♮6 8th, start of group
`E1 18`	*staccato* dot
`E0 01`	dynamic ***p***
`A5 04 02 00 08 00`	*crescendo* hairpin, half note plus 8th long
`84 02 08 00`	next 8th
`96 40 08 00 00 00`	G6 8th
`E1 18`	*staccato* dot
`84 02 04 00`	next 8th
`96 40 08 00 00 00`	G6 8th
`E1 18`	*staccato* dot
`84 02 04 01`	next 8th
`96 40 08 00 00 00`	G6 8th
`E1 18`	*staccato* dot
`84 02 02 00`	next 8th
`96 40 08 00 00 08`	G6 8th, end of group
`E1 18`	*staccato* dot
`80 13`	measure 19
`97 01 08 00 00 01`	C♮7 8th, start of group
`E1 18`	*staccato* dot
`E0 00`	dynamic ***mp***
`A5 02 02 00`	*crescendo* hairpin, half note long
`84 02 08 00`	next 8th
`97 00 08 00 00 00`	C7 8th
`E1 18`	*staccato* dot
`84 02 04 00`	next 8th
`97 00 08 00 00 00`	C7 8th
`E1 18`	*staccato* dot
`84 02 04 01`	next 8th
`97 00 08 00 00 08`	C7 8th, end of group
`E1 18`	*staccato* dot
`80 14`	measure 20
`97 11 08 00 00 01`	D♮7 8th, start of group
`E1 18`	*staccato* dot
`E0 40`	dynamic ***mf***
`A5 02 04 01`	*crescendo* hairpin, dotted quarter long
`84 02 08 00`	next 8th
`97 10 08 00 00 00`	D7 8th
`E1 18`	*staccato* dot
`84 02 04 00`	next 8th
`97 10 08 00 00 08`	D7 8th, end of group
`E1 18`	*staccato* dot
`80 15`	measure 21
`97 33 08 00 00 01`	F♯7 8th, start of beamed group
`E1 18`	*staccato* dot
`E0 41`	dynamic ***f***
`A5 02 04 00`	*crescendo* hairpin, quarter long
`84 02 08 00`	next 8th
`97 30 08 00 00 00`	F7 8th
`E1 18`	*staccato* dot
`84 02 04 00`	second beat
`97 21 08 00 00 00`	E♮7 8th
`E1 18`	*staccato* dot
`E0 42`	dynamic ***ff***
`84 02 04 01`	next 8th
`A0 08 00 08`	8th rest, end of group
`84 02 02 00`	third beat
`96 43 0A 00 00 01`	G♯6 quintuplet 8th, start of group
`E0 00`	dynamic ***mp***
`A1 00 00 04 02 00 08 00`	slur, half plus 8th long
`84 04 02 00 0A 00`	next quintuplet 8th
`96 63 0A 00 00 00`	B♯6 quintuplet 8th
`84 04 02 00 05 00`	next quintuplet 8th
`97 03 0A 00 00 00`	C♯7 quintuplet 8th
`84 04 02 01 14 00`	next quintuplet 8th
`96 53 0A 00 00 00`	A♯6 quintuplet 8th
`84 06 02 00 05 00 05 00`	last quintuplet 8th
`96 61 0A 00 00 08`	B♮6 quintuplet 8th, end of group
`80 16`	measure 22
`96 22 08 00 00 01`	E♭6 8th, start of group
`E1 18`	*staccato* dot
`84 02 08 00`	next 8th
`96 20 08 00 00 00`	E6 8th
`E1 11`	*tenuto* dash
`E0 01`	dynamic ***p***
`84 02 04 00`	next 8th
`A0 08 00 00`	8th rest
`84 02 04 01`	next 8th
`96 20 08 00 00 08`	E6 8th, end of group
`E1 18`	*staccato* dot
`84 02 02 00`	third beat
`96 20 08 00 00 01`	E6 8th, start of group
`E1 18`	*staccato* dot
`84 04 02 00 08 00`	next 8th
`96 20 08 00 00 00`	E6 8th
`E1 18`	*staccato* dot
`84 02 02 01`	next 8th
`A0 08 00 00`	8th rest
`84 02 02 02`	next 8th
`96 20 08 00 00 08`	E6 8th, end of group
`E1 18`	*staccato* dot

TABLEAUX D'UNE EXPOSITION

PROMENADE

M. P. MUSSORGSKY

Orchestration by
Maurice Ravel

Allegro giusto, nel modo russico; senza allegrezza, ma poco sostenuto

2 Flauti
e Flauto Piccolo

3 Oboi

2 Clarinetti in Si ♭

Clarinetto basso

2 Fagotti

Contrafagotto

Corni I, II in Fa

Corni III, IV in Fa

3 Trombe in Do

Tromboni I, II

Trombone III
e Tuba

Allegro giusto, nel modo russico; senza allegrezza, ma poco sostenuto

Violino I

Violino II

Viola

Violoncello

Contrabasso

MUSSORGSKY, ORCH. RAVEL: *TABLEAUX D'UNE EXPOSITION*, FIRST PAGE

Although there is relatively little music in this example, it is included to demonstrate how a score is set up at the beginning of each chunk. This includes the main title and other text at the top, all the instrument names, key signatures and transpositions (the note data for the horns in the music chunk is at concert pitch). Note that a somewhat archaic convention is used for the trumpets: the part is written without key signature, although it is not transposed (the trumpets are in C, while the piece is in B flat).

Note also the four consecutive EXPRESSION TEXT events in the header chunk. This marking in the score is exceptionally long and in such cases it is recommended that the text be broken into several events, as not all notation programs will be able to handle such long strings.

From the third bar, this music is chordal in nature, so the extract is also used to show that a "vertical" approach can be used just as easily as a horizontal one. Measure rests have been omitted—these should really be regarded as an optional event, since any sensible notation program will, at least optionally, put these in empty measures automatically.

This example is shown as if this were the *entire* file, so END OF CHUNK events are included.

256 bytes in the header chunk …

`4E 54 68 64`	chunk type "NThd"
`00 00 00 F8`	length
`00 10`	16 staves
`00 00 00 00 10 10 00 00 00 21 10 00 00 20 10 10`	clefs
`8`	8 staff blocks
`00 01 00 06 03`	woodwind, blocked barlines, square bracket
`00 03 00 04 04`	clarinets, curly brace (last bit ignored)
`00 05 00 06 04`	bassoons, curly brace
`00 07 00 0B 03`	brass, blocked barlines, square bracket
`00 07 00 08 04`	horns, curly brace
`00 0A 00 0B 04`	trombones & tuba, curly brace
`00 0C 00 10 03`	strings, blocked barlines, square bracket
`00 0C 00 0D 04`	violins I & II, curly brace
`F2 19 54 41 42 4C 45 41 55 58 20 44 27 55 4E 45 20 45 58 50 4F 53 49 54 49 4F 4E`	title "TABLEAUX D'UNE EXPOSITION"
`F0 10 4D 2E 20 50 2E 20 4D 55 53 53 4F 52 47 53 4B 59`	general text "M. P. MUSSORGSKY"
`F0 1E 4F 72 63 68 65 73 74 72 61 74 69 6F 6E 20 62 79 0D 4D 61 75 72 69 63 65 20 52 61 76 65 6C`	general text "Orchestration by<CR>Maurice Ravel"
`F2 09 50 52 4F 4D 45 4E 41 44 45`	title "PROMENADE"
`F8 10 41 6C 6C 65 67 72 6F 20 67 69 75 73 74 6F 2C 20`	expression text "Allegro giusto, " (NB space at end)
`F8 12 6E 65 6C 20 6D 6F 64 6F 20 72 75 73 73 69 63 6F 3B 20`	expression text "nel modo russico; "
`F8 12 73 65 6E 7A 61 20 61 6C 6C 65 67 72 65 7A 7A 61 2C 20`	expression text "senza allegrezza, "
`F8 11 6D 61 20 70 6F 63 6F 20 73 6F 73 74 65 6E 75 74 6F`	expression text "ma poco sostenuto"
`C2 05 04`	5/4 time signature
`80 02`	measure 2
`C2 06 04`	6/4
`80 03`	measure 3
`C2 05 04`	5/4
`80 04`	measure 4
`C2 06 04`	6/4
`80 05`	measure 5
`C2 05 04`	5/4
`FF`	end of chunk

1225 bytes in the music chunk …

`4E 4D 75 73`	chunk type “NMus”
`00 00 04 C1`	length
`F2 19 32 20 46 6C 61 75 74 69 0D 65 20 46 6C 61 75 74 6F 20 50 69 63 63 6F 6C 6F`	identifier “2 Flauti<CR>e Flauto Piccolo”
`C4 22`	key signature 2 flats
`8D 00 02`	staff 2
`F2 06 33 20 4F 62 6F 69`	identifier “3 Oboi”
`C4 22`	key signature 2 flats
`8D 00 03`	staff 3
`F2 13 43 6C 61 72 69 6E 65 74 74 69 20 69 6E 20 53 69 00 nn 00`	identifier “Clarinetti in Si ♭” (with toggle to music character set and back again—nn is dummy value for ♭)
`CA 42`	transposed +2 semitones
`8D 00 04`	staff 4
`F2 10 43 6C 61 72 69 6E 65 74 74 69 20 62 61 73 73 69`	identifier “Clarinetto basso”
`CA 4E`	transposed +14 semitones
`8D 00 05`	staff 5
`F2 09 32 20 46 61 67 6F 74 74 69`	identifier “2 Fagotti”
`C4 22`	key signature 2 flats
`8D 00 06`	staff 6
`F2 0D 43 6F 6E 74 72 61 66 61 67 6F 74 74 6F`	identifier “Contrafagotto”
`C4 22`	key signature 2 flats
`8D 00 07`	staff 7
`F2 11 43 6F 72 6E 69 20 49 2C 20 49 49 20 69 6E 20 46 61`	identifier “Corni I, II in Fa”
`CA 47`	transposed +7 semitones
`8D 00 08`	staff 8
`F2 13 43 6F 72 6E 69 20 49 49 49 2C 20 49 56 20 69 6E 20 46 61`	identifier “Corni III, IV in Fa”
`CA 47`	transposed +7 semitones
`8D 00 0A`	staff 10 (staff 9 left to last: the music starts there)
`F2 0E 54 72 6F 6D 62 6F 6E 69 20 49 2C 20 49 49`	identifier “Tromboni I, II”
`C4 22`	key signature 2 flats
`8D 00 0B`	staff 11
`F2 13 54 72 6F 6D 62 6F 6E 65 20 49 49 49 0D 65 20 54 75 62 61`	identifier “Trombone III<CR>e Tuba”
`C4 22`	key signature 2 flats
`8D 00 0C`	staff 12
`F2 09 56 69 6F 6C 69 6E 6F 20 49`	identifier “Violino I”
`C4 22`	key signature 2 flats
`8D 00 0D`	staff 13
`F2 0A 56 69 6F 6C 69 6E 6F 20 49 49`	identifier “Violino II”
`C4 22`	key signature 2 flats
`8D 00 0E`	staff 14
`F2 05 56 69 6F 6C 61`	identifier “Viola”
`C4 22`	key signature 2 flats
`8D 00 0F`	staff 15
`F2 0B 56 69 6F 6C 6F 6E 63 65 6C 6C 6F`	identifier “Violoncello”
`C4 22`	key signature 2 flats
`8D 00 10`	staff 16
`F2 0B 43 6F 6E 74 72 61 62 61 73 73 6F`	identifier “Contrabasso”
`C4 22`	key signature 2 flats
`CA 4C`	transposed +12 semitones
`8D 00 09`	staff 9
`F2 0E 33 20 54 72 6F 6D 62 65 20 69 6E 20 44 6F`	identifier “3 Trombe in Do”

98 40 04 00 00 00	G8 quarter note
F0 02 31 2E	general text "1."
E1 11	*tenuto* dash
E0 41	dynamic ***f***
84 02 04 00	next quarter
98 30 04 00 00 00	F8 quarter
E1 11	*tenuto* dash
84 02 02 00	next quarter
98 62 04 00 00 00	B♭8 quarter
E1 11	*tenuto* dash
84 02 02 01	next quarter
99 00 08 00 00 01	C9 8th, start of group
E1 11	*tenuto* dash
A1 00 00 02 04 00	slur, quarter long
84 02 02 02	next 8th
99 30 08 00 00 08	F9 8th, end of group
84 02 01 00	last quarter
99 10 04 00 00 00	D9 quarter
E1 11	*tenuto* dash
80 02	measure 2
99 00 08 00 00 01	C9 8th, start of group
E1 11	*tenuto* dash
A1 00 00 02 04 00	slur, quarter long
84 02 08 00	next 8th
99 30 08 00 00 08	F9 8th, end of group
84 02 04 00	next quarter
99 10 04 00 00 00	D9 quarter
E1 11	*tenuto* dash
84 02 02 00	next quarter
98 62 04 00 00 00	B♭8 quarter
E1 11	*tenuto* dash
84 02 02 01	next quarter
99 00 04 00 00 00	C9 quarter
E1 11	*tenuto* dash
84 02 01 00	next quarter
98 40 04 00 00 00	G8 quarter
E1 11	*tenuto* dash
84 04 01 00 04 00	last quarter
98 30 04 00 00 00	F8 quarter
E1 11	*tenuto* dash
80 03	measure 3
98 40 04 00 00 00	G8 quarter
D8 10 00	add D8 to chord
D7 62 00	add B♭7 to chord
E1 11	*tenuto* dash
E0 41	dynamic ***f***
8D 00 0B	staff 11
97 40 04 00 00 00	G7 quarter
D6 40 00	add G6 to chord
E1 11	*tenuto* dash
E0 41	dynamic ***f***
8D 00 07	staff 7
98 40 04 00 00 00	G8 quarter
D7 62 00	add B♭7 to chord
E1 11	*tenuto* dash
E0 41	dynamic ***f***
8D 00 08	staff 8
98 10 04 00 00 00	D8 quarter
D7 40 00	add G7 to chord
E1 11	*tenuto* dash
E0 41	dynamic ***f***
84 02 04 00	next quarter
98 00 04 00 00 00	C8 quarter
D7 30 00	add F7 to chord
E1 11	*tenuto* dash
8D 00 07	staff 7
98 30 04 00 00 00	F8 quarter
D7 50 00	add A7 to chord
E1 11	*tenuto* dash
8D 00 0B	staff 11
97 30 04 00 00 00	F7 quarter
D6 50 00	add A6 to chord
E1 11	*tenuto* dash
8D 00 09	staff 9
98 30 04 00 00 00	F8 quarter
D8 00 00	add C8 to chord
D7 50 00	add A7 to chord
E1 11	*tenuto* dash
84 02 02 00	next quarter
98 62 04 00 00 00	B♭8 quarter
D8 10 00	add D8 to chord
D7 60 00	add B7 to chord
E1 11	*tenuto* dash
8D 00 0B	staff 11
97 40 04 00 00 00	G7 quarter
D6 40 00	add G6 to chord
E1 11	*tenuto* dash
8D 00 07	staff 7
98 62 04 00 00 00	B♭8 quarter
D8 10 00	add D8 to chord
E1 11	*tenuto* dash
8D 00 08	staff 8
98 40 04 00 00 00	G8 quarter
D7 62 00	add B♭7 to chord
E1 11	*tenuto* dash
84 02 02 01	next quarter
98 50 04 00 00 00	A8 quarter
D8 00 00	add C8 to chord
E1 11	*tenuto* dash
8D 00 07	staff 7
99 00 04 00 00 00	C9 quarter
D8 30 00	add F8 to chord
E1 11	*tenuto* dash
8D 00 0B	staff 11
97 30 04 00 00 00	F7 quarter
D6 30 00	add F6 to chord
E1 11	*tenuto* dash
8D 00 09	staff 9
CE 02	2 voices
8E 20	voice 2
98 50 04 00 00 00	A8 quarter
D8 00 00	add C8 to chord
E3 11	*tenuto* dash below
8E 10	voice 1
99 00 08 00 00 01	C9 8th, start of group
E2 11	*tenuto* dash above
A2 00 00 02 04 00	slur above, quarter long
84 02 02 02	next 8th
99 30 08 00 00 08	F9 8th, end of group
84 02 01 00	last quarter
CE 01	1 voice
99 10 04 00 00 00	D9 quarter
D8 50 00	add A8 to chord
D8 30 00	add F8 to chord
E1 11	*tenuto* dash

```
8D 00 0B                staff 11
97 10 04 00 00 00       D7 quarter
D6 10 00                add D6 to chord
E1 11                   tenuto dash
8D 00 07                staff 7
99 10 04 00 00 00       D9 quarter
D8 30 00                add F8 to chord
E1 11                   tenuto dash
8D 00 08                staff 8
97 50 04 00 00 00       A7 quarter
D8 10 00                add D8 to chord
E1 11                   tenuto dash
80 04                   measure 4
98 50 04 00 00 00       A8 quarter
D8 00 00                add C8 to chord
E1 11                   tenuto dash
8D 00 07                staff 7
99 00 04 00 00 00       C9 quarter
D8 30 00                add F8 to chord
E1 11                   tenuto dash
8D 00 0B                staff 11
97 30 04 00 00 00       F7 quarter
D6 30 00                add F6 to chord
E1 11                   tenuto dash
8D 00 09                staff 9
CE 02                   2 voices
8E 20                   voice 2
98 50 04 00 00 00       A8 quarter
D8 00 00                add C8 to chord
E3 11                   tenuto dash below
8E 10                   voice 1
99 00 08 00 00 01       C9 8th, start of group
E2 11                   tenuto dash above
A2 00 00 02 04 00       slur above, quarter long
84 02 08 00             next 8th
99 30 08 00 00 08       F9 8th, end of group
84 02 04 00             next quarter
CE 01                   1 voice
99 10 04 00 00 00       D9 quarter

D8 62 00                add B♭8 to chord
D8 30 00                add F8 to chord
E1 11                   tenuto dash
8D 00 0B                staff 11
97 60 04 00 00 00       B7 quarter
D6 60 00                add B6 to chord
E1 11                   tenuto dash
8D 00 07                staff 7
99 10 04 00 00 00       D9 quarter
D8 30 00                add F8 to chord
E1 11                   tenuto dash
8D 00 08                staff 8
98 62 04 00 00 00       Bb8 quarter
D8 10 00                add D8 to chord
E1 11                   tenuto dash
84 02 02 00             next quarter
98 40 04 00 00 00       G8 quarter

D7 62 00                add B♭7 to chord
E1 11                   tenuto dash
8D 00 07                staff 7

98 62 04 00 00 00       B♭8 quarter
D8 10 00                add D8 to chord
E1 11                   tenuto dash
8D 00 0B                staff 11
97 40 04 00 00 00       G7 quarter
D6 40 00                add G6 to chord
E1 11                   tenuto dash
8D 00 09                staff 9

98 62 04 00 00 00       B♭8 quarter
D8 40 00                add G8 to chord
D8 10 00                add D8 to chord
E1 11                   tenuto dash
84 02 02 01             next quarter
99 00 04 00 00 00       C9 quarter
D8 40 00                add G8 to chord
D8 20 00                add E8 to chord
E1 11                   tenuto dash
8D 00 0B                staff 11
97 00 04 00 00 00       C7 quarter
D6 00 00                add C6 to chord
E1 11                   tenuto dash
8D 00 07                staff 7
99 00 04 00 00 00       C9 quarter
D8 20 00                add E8 to chord
E1 11                   tenuto dash
8D 00 08                staff 8
98 40 04 00 00 00       G8 quarter
D8 00 00                add C8 to chord
E1 11                   tenuto dash
84 02 01 00             next quarter
98 00 04 00 00 00       C8 quarter
D7 20 00                add E7 to chord
E1 11                   tenuto dash
8D 00 07                staff 7
98 40 04 00 00 00       G8 quarter
D7 40 00                add G7 to chord
E1 11                   tenuto dash
8D 00 0B                staff 11

97 21 04 00 00 00       E♯7 quarter

D6 21 00                add E♯6 to chord
E1 11                   tenuto dash
8D 00 09                staff 9
98 40 04 00 00 00       G8 quarter
D8 00 00                add C8 to chord
D7 40 00                add G7 to chord
E1 11                   tenuto dash
84 04 01 00 04 00       last quarter
98 30 04 00 00 00       F8 quarter
D8 00 00                add C8 to chord
D7 50 00                add A7 to chord
E1 11                   tenuto dash
8D 00 0B                staff 11
97 30 04 00 00 00       F7 quarter
D6 30 00                add F6 to chord
E1 11                   tenuto dash
8D 00 07                staff 7
98 30 04 00 00 00       F8 quarter
D7 50 00                add A7 to chord
E1 11                   tenuto dash
8D 00 08                staff 8
98 00 04 00 00 00       C8 quarter
D7 30 00                add F7 to chord
E1 11                   tenuto dash
FF                      end of chunk
```