Bruce McLean

DARMS

A REFERENCE MANUAL

by

Raymond Erickson

The DARMS Project
c/o Prof. Raymond Erickson
Department of Music
Queens College
Flushing, New York 11367

June 1976

Copy 7

TABLE OF CONTENTS

4. Horizontal stratification: Linear Decomposition Mode (Backspace Code) C4

II. Attributes of Notes and Rests (other than Space Codes)

A. Order of Encoding D1

1. Accidentals D2

2. Special Noteheads D3

3. Durations (including Rests) D4

4. Ties D5

5. Stems D6

6. Tremolos D7

7. Beams D8

8. Articulation Symbols D9

9. Fingerings D10

10. Ornaments and Editorial Accidentals D11

11. Slurs D12

12. Figured Bass Symbols D13

B. Recycling for double-stemmed Notes D14

III. Other Music Symbols (actually in score graphically)

A. Clefs E1

B. Key Signatures E2

C. Meter Signatures E3

D. Barlines E4

E. Dynamics E5

F. Literals (natural language information) E6

G. Repetition Signs E7

H. Dictionary of Additional Symbols E8

I. Chord Analysis Symbols TBA E9

IV. Special Codes

A. Codes Necessary for Conveying Musical Information

1.	Groupette Definer Codes	F1
2.	Ossia Stave Codes	F2
*3.	Size Codes	F3

B. Convenience Codes – Abbreviations on the Token Level

1.	Stave Transposition Codes	F4
2.	Doubling Mode Codes	F5
*3.	Equate Codes (String Definition)	F6
*4.	Matrix Code TBA	F7
5.	Comment Code	F8
6.	Distribution Code TBA	F9

Scopes A6 backside

ACKNOWLEDGEMENTS

This version of the DARMS Reference Manual, which supplants my much briefer document of 1971, has been "completed" several times in the past few years. Indeed, on two occasions, large portions of the text were typed with the assumption that DARMS itself had achieved relative stability and completeness.

That a description of DARMS is finally appearing in what is hoped will be a form adequate for most applications is due in no small measure to my several colleagues in the DARMS Project who, in the course of the past several years, have discovered lacunae in the facilities of DARMS and caused solutions to their problems to be found. This manual has been considerably enriched--if also necessarily delayed!--by their contributions.

David A. Gomberg (American University, Washington, D.C.) began, in 1971, to encode a large portion of Elliott Carter's Double Concerto as part of the work that resulted in his doctoral dissertation[1] as well as major expansions of DARMS syntax. In addition, he was the first to work out a formal definition of canonical DARMS, which has greatly assisted

[1] David A. Gomberg, "A Computer-Oriented System for Music Printing" (Sever Institute of Washington University, St. Louis, 1975).

the work of others developing DARMS support software.

Anthony B. Wolff collaborated closely with me during 1973-74 as one of the intermediate forms of the DARMS manual took shape; a few chapters in the manual were originally drafted by him. Mr. Wolff also initiated the design and implementation of the Input DARMS-to-Canonical DARMS Translator (popularly known as the "Canonizer" program) to process DARMS datasets "approved" as well-formed by my Input DARMS Syntax Checker program simultaneously under development. Finally, because of his professional qualifications in the three fields of music, computer science, and psychology, Mr. Wolff (a doctoral candidate in Clinical Psychology at McGill University, Montreal), has contributed significantly to the understanding of the psycholinguistic problems encountered in designing an encoding language for music.[2]

More recently, Mr. Bruce McLean, a graduate student at the School of Advanced Technology, SUNY-at-Binghamton, raised important questions whose consideration ultimately resulted in a generalization of certain DARMS facilities and a major reorganization of the manual. Mr. Wolff's successor as principal implementor of the "canonizer" program, Mr. McLean has also concerned himself with formalizing both Input DARMS and musical Braille.

Thanks are also due Ms. Judy Vogel who, as a

---

[2]These problems will be discussed in Raymond Erickson and Anthony B. Wolff, "The DARMS Project: Implementation of an Artificial Language for the Representation of Music," to be published in a volume, edited by Sally and Walter Sedelow, in the series Advances in Computers (Mouton Press, 1977?).

doctoral student in music at the City University of New York, was my perceptive research assistant during 1973-74. Her comments, questions and suggestions resulted in clearer and more complete explanations of DARMS facilities and were influential in the creation of others.

Finally, but certainly not least, is Stefan Bauer-Mengelberg, the originator of DARMS (in the form known as the Ford-Columbia language) who has coordinated the work of all during the entire history of the DARMS Project. The basic principles of music encoding-language design he laid down in the 1960's have proved sound and adequate for the task; indeed, one of the most remarkable things about DARMS is the consistency and continuity that has been maintained within its structure as its facilities have expanded enormously. In addition to being the final arbiter of disputes over matters of DARMS syntax among those who have contributed to it, Mr. Bauer-Mengelberg has generously given of his time as this manual has taken final form. He has, in effect, served as my editor, in this capacity improving the document in countless details. To him also must go special credit for conceiving the form of the Instrument Code paradigm (section A).

I am also indebted to several outside sources that have directly or indirectly helped bring this manual into being. Among them are the IBM Systems Research Institute in New York

City (Research Fellowship, 1970-71); the National Endowment for the Humanities (Major grant, 1973-74, with co-principal investigator Wolff; and co-sponsorship, with the School of Advanced Technology and the Department of Music, SUNY-at-Binghamton, of MUSICOMP '76, at which this manual is to be introduced); and the American Council of Learned Societies (grant-in-aid, 1976).

Lastly, I have been blessed with a typist of uncommon intelligence and skill in the person of Susan Hellauer-Strapac, a graduate student at Queens College. Her expertise in medieval paleography has been gloriously manifest as my manuscript, wrinkled, smudged and laden with several generations of cryptic glosses and corrections, has been transcribed into the elegant format--further enhanced by the musical examples drawn by Peter Dreyfuss, also of Queens College--that the manual now possesses.

The responsibility for omissions, ambiguities and inconsistencies--not to say errors--rests, of course, with me, and I welcome corrections and suggestions for improvement to be incorporated into supplements to be issued from time to time.

R.E.
Department of Music
Queens College
City University of New York
June, 1976

AN INTRODUCTION TO DARMS

DARMS (initially called the Ford-Columbia Music Representation after the foundation and university that sponsored the first stage of its development) is an artificial language for the representation of musical scores to computers. It was originally conceived by Stefan Bauer-Mengelberg in 1963 as a component in a proposed new technology of automated music-printing. A few years later, however, the name DARMS (in honor of a helpful advocate of the project) gradually began to replace the longer and more unwieldy appellation; it is interpreted as an acronym for Digital Alternate Representation of Musical Scores.

Because of its intended application to music-printing, DARMS has been designed to yield comprehensive and accurate encodings of complete musical scores; furthermore, since it will be some time before the encoding process can be fully automated with the help of optical scanners, considerable effort has been expended to assure that the code is easily learned and used.

A properly encoded DARMS dataset will contain all the information contained in the musical score that has been supplied by the composer through notation and comments; it will thus specify the relative position of all symbols in the score but will not resolve ambiguities of interpretation that are the

concern of the performer or music analyst; likewise, it will not represent purely editorial features such as page boundaries, over which the composer has no control. On the other hand, provision has been made for encoder comments that will not be interpreted as being part of the score.

There are many encoding order and abbreviation options available to the DARMS encoder. A consequence of this is that a given musical passage may be rendered in many different but equivalent encodings. The options enable the encoding process to be simplified or shortened without loss of information; furthermore, the likelihood of encoding and keyboarding errors is reduced. Therefore, the user is encouraged to master the abbreviations, options and "convenience codes" at the outset, as there is no merit in creating raw datasets that are unnecessarily large.

The encoder is, at the same time, warned against incomplete encoding (except for highly specialized applications), for DARMS (like certain programming languages) has an extensive series of defaults; therefore, values for certain information parameters (such as stem direction) will be supplied by the DARMS system software if countermanding values are not explicitly encoded. Suppression of codes for default categories could therefore result in the generation of incorrect information by the User-DARMS-to-Canonical DARMS Translator discussed below. In any event, incomplete encodings are not desirable if a library of DARMS-encoded scores is to be established and shared by a community of users.

DARMS System Software

A body of software (programs) to facilitate the use of DARMS in music research has been under development for some time and, as this manual is being written, is largely finished but insufficiently tested. (DARMS users are encouraged to send datasets to the author to provide test data.) Because the syntax of music notation (and therefore its DARMS representation) is so rich and variable, a comprehensive Syntax Checker is a real necessity for locating grammatical errors in DARMS datasets. However, since the completion of such a program has had to wait for the stabilization of DARMS syntax itself, it has been decided that the version of DARMS described in this manual, to be known as DARMS 76, will be that supported by the software. (This does not apply to errors in the manual that may in the course of time be discovered, of course.) Future additions and modifications to DARMS are not precluded, but these will be incorporated into later versions of DARMS with their own designations and corresponding software.

A "user-DARMS" or "input-DARMS" dataset (the terms are synonymous and refer to the raw data resulting from the encoding process) that is judged by the syntax-checker to be well-formed, that is, free from syntax errors, can be processed by the User-DARMS to Canonical-DARMS Translator, a software component designed by Anthony B. Wolff and being implemented by him and Bruce McLean. Popularly known as the "Canonizer", this program takes any of the many possible correct encodings of a score and converts it into the unique canonical-DARMS data-string for that

score. The purpose of "canonization" is to transform the input data into a form more suitable for machine processing: thus the rigorous imposition of precedence rules upon the ordering of the information.

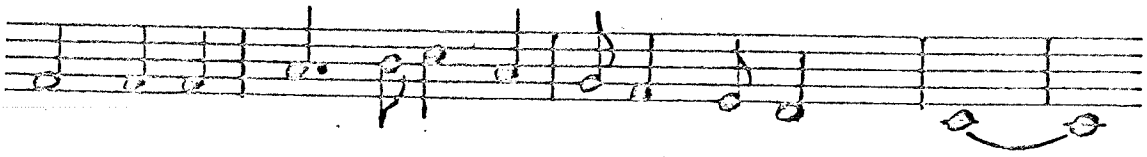
Several steps are involved in the canonization process. First, comments are removed (but their texts saved and their locations recorded for future user reference), all abbreviations are expanded and all default values supplied. Then, the information for each Instrument in turn is redistributed to form a "canonical" string in which the information is given as a series of vertical slices, internally ordered from bottom to top, moving from left to right. When this intra-Instrument canonization is completed for every Instrument, the user may opt for inter-Instrument or "score canonization", whereby the several Instrument strings are combined to form vertical slices, internally ordered from the bottom to top of the score, proceeding from left to right. (Obviously, the last stage applies only to scores involving more than one Instrument.)

The comprehensiveness and syntactic complexity of DARMS may overwhelm the novice at first, but, if its grammar is studied in stages, the spirit underlying the system and the naturalness of the mnemonics are easily grasped. To facilitate this, a few encoding examples are given below. The reader should fully understand the DARMS code for each example before proceeding to the next and be aware that there are other valid encodings of the given passages. However, in all cases the DARMS encodings

can be translated back into the original music notation with no change or loss of information. It is recommended that the reader actually make such reverse transcriptions to test his or her understanding of the principles involved in each example.

Example 1

Read the sections on Space Codes (B1), Delimiters (C1), Order of Encoding (D1), Duration Codes (D4), Tie Codes (D5), Stem Codes (D6), and Barline Codes (E4). With the basic information in these sections as background, the reader can encode the following passage.



[1] 22HU 22QU 22QU / 24Q.U 25ED 26QD 24QU / 23EU 22QU 21EU 20HU / 19WJ / 19W

[2] 2H 2Q 2Q / 4Q. 5E 6Q 4Q / 3E 2Q 1E 0H / 19WJ / 19W

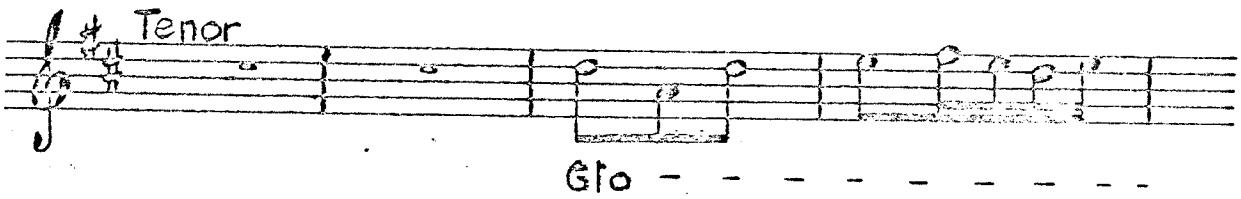
[3] 2H { Q / 2 { Q / 2 / 4. 5E 6Q 4 / 3E 2Q 1E 0H / 19WJ / { W / 19

There are given for this passage three encodings, equivalent in meaning, requiring 73, 51 and 45 (or 46) characters, respectively. (As explained in section C1, a Delimiter Blank consists of one or more consecutive blanks; therefore, each Delimiter Blank is counted here as one character.) Version 1 is in unabbreviated DARMS; the others utilize various DARMS abbreviations. Version 2, for example, demonstrates the suppression of Stem Codes where a given Stem Code is equivalent to the default value; also, "2-Suppression" (B1.2) is applied to the

Space Codes 20-29. Version 3 employs not only these abbreviations but also "Sigma Suppression" (B1.2) and "Delta Suppression" (D3.4f), applicable when either two successive notes have the same Space Code (sigma or σ) or Duration Code (delta or δ).

Example 2

To the categories of information required above we now add Instrument Codes (A; only the first page of section A need be read), Pseudo-Space Codes (B6), Short-Form Beam Codes (D8), Clef Codes (E1), Key Signature Codes (E2), Meter Signature Codes (E3), and Literal Codes (E6). Because of the length of the DARMS representations, the data-strings have been segmented at points where a Delimiter Blank occurs.

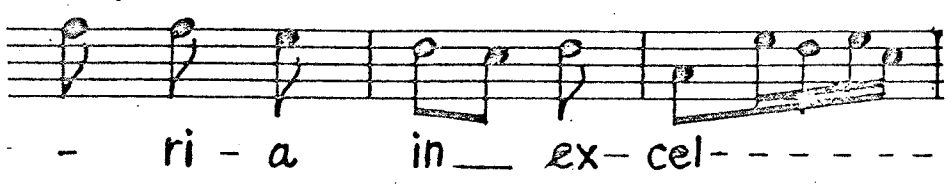


(Version 1)

I4 !G !K2# 00@¢TENOR$ R2W / (7,@¢GLO-$ 4 7) / (8 (9 8 7 8)) /

(Version 2)

I4 !G !K2# 00@¢TENOR$ R2W / !&,(7 4 7) / (8 (9 8 7 8)) /
@¢GLO-



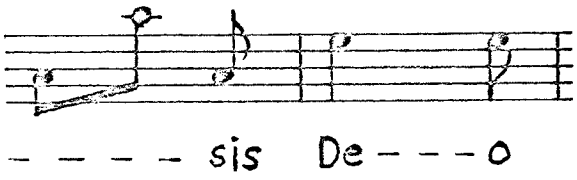
(Version 1, continued)

9E 9,@RI-$ 8,@A$ / (7,@IN_$ 6) 7,@EX-$ / (4D,@CEL-$ (8 7 8 6)) /

(Version 2, continued)

9E 9 8 / (7 6) 7 / (4D (8 7 8 6)) /

¬√2E,RI- ¬E,A / IN_ ¬Q,EX- / CEL-



(Version 1, completed)

(4D 31) 4,@SIS$ / 8Q,@¢DE-$ E,@O$ /

(Version 2, completed)

(4D 31) 4 / 8Q E,&,

¬/Q, SIS / ¢DE- ¬Q,O$ &$ /

The reader's attention is drawn to certain features of the encodings of this example:

1. The encoding of Version 1 does not employ Linear Decomposition Mode and therefore results from a single pass through the score moving from left to right, all the information in a given vertical slice in the score being encoded before any in the next slice to the right.

2. The encoding of Version 2 employs Linear Decomposition Mode. Here the tenor part has been encoded in two layers, viz., music, then text. The musical notation and corresponding DARMS code for each layer have been segmented (at points where Delimiter Blanks occur) because of the page width of the manual; however, all of Layer One (music information) is assumed to precede Layer Two (text information) in the encoding stream.

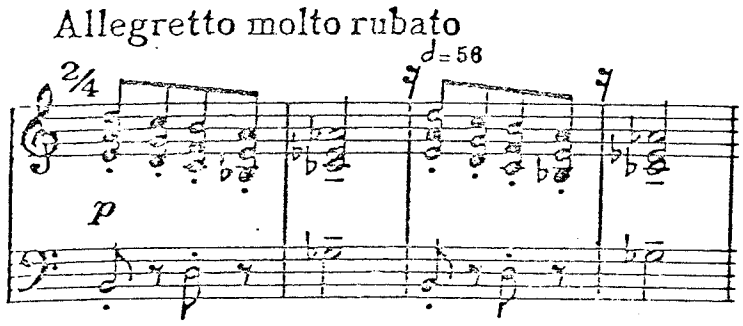
3. The choice of the integer 4 as Instrument Code Identifier for the tenor voice is completely arbitrary. Any integer not already otherwise assigned might have been chosen to stand for "Tenor".

4. The Pseudo-Space Code of 00 indicates the placement of information "above the stave". The Pseudo-Space Code 50 ("below the stave") is not given in the code for the text since 50 is the default value for the position of a Literal.

5. Note the use of a special abbreviation for two whole measures of rest: R2W . See page D4.7.

Example 3.

The opening of the eleventh Bartok Bagatelle, Op. 6*, provides opportunities to use abbreviation techniques even more powerful than those above: Vertical Structure Definers and References (B3--B5) and Stave Transposition Codes (F4). The sections on Dynamics (E5) and Equate Codes (F6) should also be read.




```
!G,77!F 00!M2/4,00+2@¢ALLEGRETTO MOLTO RUBATO$
!&,=1=(!|1|1+3+3' |,0|' |,19|' -18|,18|') / 19-+2-+3-H_ / =$,=1$,&,
!&,!+50,=2=3E' RE 5' R / 8-H_ / =$,=2$,!+0,&,
VP ¬/2,00RS 00+1@|!'HU|=56$ ¬/,00RS,&$
```

*Used by permission of Boosey and Hawkes.

The following observations may be made:

1. Linear Decomposition Mode is used, after the clefs and meter signature are encoded, to permit the encoding to be done in several passes or layers: upper stave (right hand), lower stave (left hand), and literal material. To illustrate more clearly the syntactic arrangement of code substrings produced by the Linear Decomposition of the score, what would normally be a continuous string has been segmented at points where Delimiters occur.

2. Encoding the parallel chords of the upper stave as a single layer is made possible by the use of Chord Structure Definers and References. This also means that Short-form Beam Codes may be employed, since all the notes of the chord are on the same beam system and Chord Structures and References of the Base-Increment type can be treated as a single Space Code for this purpose.

3. Since the music for measures 1 and 2 is repeated literally in measures 3 and 4, respectively, the former segment is defined as Equate-string 1 (right hand) and 2 (left hand), which are then "called" by appropriate Equate Code References upon their recurrence. It would have been even more efficient (although perhaps less useful for this illustration) to embrace the right- and left-hand code (including Linear Decomposition Mode Codes) for measures 1 and 2 within an Equate String Definer. Then a single Equate String Reference could stand for the music in <u>both</u> hands for measures 3 and 4.

4. Duration Codes are required for the first note and first rest of the lower stave in measure 1, since Duration Codes for notes and rests propagate forward independently, although they be the same (e.g., E); however, by virtue of this and by the rules for suppression of Duration Codes (D4.3), the Duration Codes can be and are suppressed for the balance of the measure.

5. The rest above the second and fourth barlines is treated as a Literal; it could also be encoded as a Dictionary Code of the form

34RS , 00?|RS|

or

AN?|RS|

(using a Space Code, Pseudo-Space Code or Attach Prefix, respectively, to indicate vertical position).

6. Three different types of Vertical Structures are employed: that using the comma to separate the codes for symbols vertically aligned (the clefs in this example), a Space-Pattern Chord Structure (defined and referenced as Chord "1"), and a Root-Increment Chord Structure (near the end of the code for the right-hand music).

7. By means of the Stave Transposition operators, the Space Codes for the left-hand music (second stave) are brought into the 01-49 range, thus making them eligible for "2-suppression", which results in a considerably shorter DARMS representation.

Instrument Codes

Throughout this manual, the proper noun "Instrument" refers to a DARMS Instrument, that is, an instrumental or vocal part of any type that is to be distinguished from any other for purposes of encoding and part-extraction. Thus, Flute and Trumpet will be encoded as distinct Instruments as will Violin I and Violin II, assuming, in both cases, that the parts have their own staves in the score. Sometimes, however, a question may arise over whether two or more parts ought to be encoded as separate Instruments; for example, in an orchestral score, the music for Oboe I and Oboe II may appear on one stave throughout, or two staves throughout, or alternately on one and on two staves. The disposition of the music must, in all cases, be clearly represented; to this end, Instrument Codes are employed.

Instrument Codes are of two types:

1. Unqualified Instrument Codes are of the form

   I*i*

   where *i* is an integer identifier greater than zero that designates an instrumental or vocal part, for example, Piano, Soprano I, etc.

The Unqualified Instrument Code will signify that all parts of the given Instrument are being encoded.

2. Qualified Instrument Codes are of the form

Ii:1.2.3. ... k

where i is an identifier as defined above, and

1,2,3... reference respectively the first, second third, etc., of k parts of Ii.

Thus, if the Oboes are assigned the Instrument Code I5 then I5:1 signifies Oboe I, I5:3 signifies Oboe III, and I5:1.3 signifies Oboes I and III.

By use of an Unqualified or Qualified Instrument Code the encoder is able to specify precisely which parts of which Instrument are "active", that is, are being encoded, at any particular time.

Because scoring practices vary widely from composer to composer and from score to score, and even within a score, the following rules are offered to assist the encoder in determining the proper Instrument Code for the passage to be represented. (The length of the passage is determined by the encoder as discussed below in "Encoding a Multi-Instrument Score.") As will be seen, changes of Instrument Code are required within the encoding of an Instrument if parts are added, deleted, doubled, or changed in any other way. It is also suggested that the Rules for Stem Code suppression (D6) be reviewed.

I. Music for one solo instrument: No Instrument Code is required.

II. All other scores. (Multi-Instr. Scores)

A. IF there is only one instrument of a given type in the score (e.g., one Viola, as in a string quartet), THEN use an Unqualified Instrument Code for that Instrument.

B. IF there is more than one instrument of a given type in the score, AND

1. IF all instruments of the given type have their own separate staves throughout (as do Violins I and II in most string quartets), THEN each instrument of the given type receives its own Unqualified Instrument Code.

2. IF the instruments of the given type do not have their own separate staves throughout AND

a. IF the instruments always share the same stave, AND

✓i. IF they play in unison (i.e., play the same note on the same stem) AND (no distinguishable parts)

✓α. IF all the instruments of the given type are playing, THEN use an Unqualified Instrument Code to cover all parts.

✓β. IF not all of the total number of parts of the given type (say, only Oboes I and III) are playing, THEN use an appropriate Qualified Instrument Code (here, I5:1.3, if I5 stands for Oboe) to specify the active parts.

✓ii. IF they do not play in unison but share the same stem AND

✓α. IF all the instruments of the given type are playing, THEN use an Unqualified Instrument Code, indicating all parts are being encoded, and encode the chords using normal conventions described in sections B3-B5. The topmost note

of the chord will be interpreted as belonging to the first part of the given type (e.g., Oboe I), the next highest to the second part, etc.

β. IF not all parts of the given type (say, only Oboes I and III) are playing, THEN use a Qualified Instrument Code (here, I5:1.3) to define all the active parts and encode the chords using normal conventions described in sections B3-B5. The highest note of the chord will be interpreted as belonging to the part specified by the first qualifier (i.e., I5:1), the next highest to that specified by the second qualifier (i.e., I5:3), etc.

iii. IF they neither play in unison nor share the same stem AND

α. IF there are always exactly two parts within the given Instrument (e.g., Oboes I and II are the only Oboe parts)

(1) AND IF the Stem Code will unambiguously differentiate between the parts (U for Oboe I and D for Oboe II) AND IF it is desired that both parts be encoded simultaneously THEN use an Unqualified Instrument Code. (normal stem code defaults)

*(2) OTHERWISE encode each part separately using appropriate Qualified Instrument Codes. Stem Codes U will be assumed for part I and D for part II except where specifically encoded to the contrary. (See section D6.)

β. IF there are two or more parts within the given Instrument (say, Oboes I, II and III), THEN use

Qualified Instrument Codes (e.g., I5:1, I5:2, I5:1.3, etc.) as required by the context. Stem Codes here (unlike in ∝.(2) above) are not always inferrable from the Instrument Code. Therefore, the normal rules for Stem Code suppression (section D6) must be applied.

b. IF they are sometimes or always on separate staves, THEN follow the rules under II.B.2.a. above when the parts are written on the same stave AND use a Qualified Instrument Code (specifying the parts being encoded) and the appropriate Space Codes to indicate the stave on which they appear. Thus, if Oboes I and III are temporarily on adjacent staves of the same brace, then I5:1 will be associated with the stave whose lowest line has the Space Code 21, whereas I5:3 will be associated with the second stave, whose lowest line has the Space Code 71. The combination of Qualified Instrument Code and Space Code permits the precise representation of arrangement of parts of a given Instrument over any number of staves.

Encoding a Multi-Instrument Score

The DARMS encoder has many options available when determining how a multi-Instrument score is to be represented in DARMS. First, there is the option of encoding each Instrument in full (i.e., from beginning to end) before proceeding to the next. (The order in which the Instruments are encoded is immaterial, since certain score-layout information, including vertical ordering of Instruments, will be provided to the DARMS system software.) Encoding a string quartet in this fashion--as would be natural if encoding from the individual parts--

will result in four substrings, each headed by a different encoder-assigned Instrument Code; thus, there will be only four Instrument Codes in the entire DARMS representation of the score.

Secondly, at the opposite extreme, there is the option of encoding the score as a series of vertical time-slices, each of which cuts across all the active Instruments. A canonical DARMS dataset, designed to facilitate machine processing for music-analytic and music-printing purposes, is ordered in this way. However, this method is not generally recommended for the human encoder, since an Instrument Code will be required for every change of Instrument within each vertical slice, thus significantly lengthening the code required to represent the score and increasing the opportunity for error in the data preparation process. The encoder is therefore advised that, in most cases, this procedure will prove wasteful.

Thirdly, there is "segmented" encoding, a kind of compromise between the two methods described above, in which the part for a given Instrument is divided into several linear sections. These are encoded in the order in which they are performed except that information pertaining to other Instruments (~~itself~~ itself possibly segmented) is interpolated between the encoding of the several sections. Thus, in segmented encoding, there will be as many statements of the Instrument Code for a given Instrument as there are segments. (One could regard the full-Instrument encoding method, the first of the three options described, as a special case of segmented encoding, the number

Local and Global Scopes

In multi-Instrument scores, the information for a particular instrument must, of course, be preceded by the appropriate Instrument Code. However, certain types of information are permitted before the first Instrument Code and, for that matter, anywhere in a DARMS dataset:

1) Groupette Definers (F1); these are global and refer to all Instruments,
2) Comments (F8),
3) Canonizer-parameters (undefined at present),
4) Score layout information for graphic programs (undefined at present,
5) An Equate-String Definer (F6) such that if its prefix and suffix were to be removed, there would remain only the token types corresponding to 1), 2) and 3) above.

In addition, the Instrument Code I0 (zero) may be used to indicate the global scope of other parameters (e.g.,'I0 (zero),!',!G' indicates that all Instruments are size-reduced and have a G clef.

Summary of Normal Scope of Switches and Definers:

1.	Chord-Structure Definers (B4) and References (B5)	A
2.	Key Signature Codes (E2)	B
3.	Meter Signature Codes (E3)	B
4.	Groupette Definers (F1)	A
5.	Ossia Stave Codes (F2)	B
6.	Size Codes (F3)	B
7.	Stave Transposition Codes (F4)	B
8.	Doubling Mode Code (F5)	B
9.	Equate Definers and References (F6)	A

A: Global or orchestral scope. Values hold across Instrument boundaries.

B: Local or instrumental scope. Holds throughout the given Instrument (even if the code for the Instrument is interrupted by information for other Instruments) until changed. May be made Global by the use of the I0 (zero) Instrument Code."

LDM - "B"

Beam Ties } "B"
Attach Slurs

of segments for each Instrument being one.) A typical example of segmentation is the system-by-system encoding explained below in "Instrument Suppression in a Score."

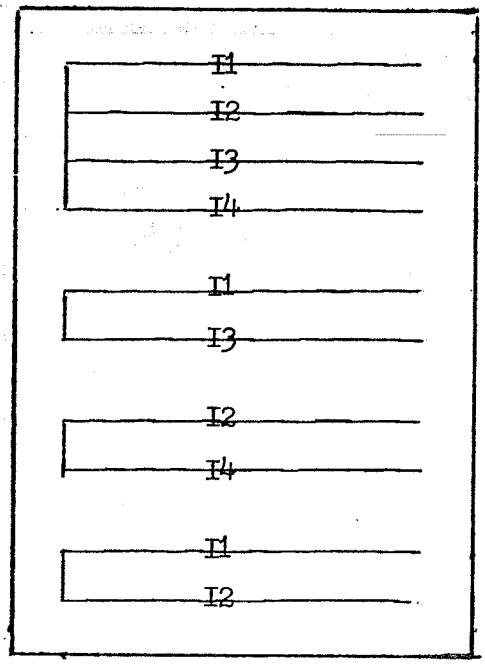
In summary, the information for a given Instrument may be completely contained in a single segment or it may be given in several discrete segments within a DARMS data-string. When appropriate, vertical time-slices through a score (or part of it) may be encoded as an alternate method. This range of choices, coupled with the power of Linear Decomposition Mode (C4) within a segment, provides a very high degree of flexibility at the encoding stage. Any possible correct encoding of a score will, however, be transformable into the same precedence-ordered canonical representation of that score.

← See opposite page.

Instrument Suppression in a Score

It often occurs that the stave(s) of an Instrument that is inactive for the length of a system (width of a page) will be suppressed. When this occurs the amount suppressed must be represented by a Skip Code (C3) to permit proper alignment of Instruments in the canonical form of the dataset being encoded. This must be done regardless of whether the Instrument is being encoded in a single segment or in several segments. If the stave suppression continues over two or more systems within a segment, the total amount of continuous suppression may be represented by a single Skip Code ¬d , the value of d being equal to the total amount suppressed.

Consider the following illustration.



a_i = 5 measures

b_i = 8 measures

c_i = 7 measures

d_i = 6 measures

Let each line stand for a stave containing all the information, in that system, for the designated Instrument and let each letter--subscripted with the value of the Instrument Identifiers 1, 2, 3, and 4--stand for the DARMS representation of the information on the corresponding staves in each system, the staves of a given system being joined by the vertical lines at the left ends of the "staves."

Given these assumptions, the information for I1, <u>coded continuously from the beginning to the end of the page</u>, is represented by

$$\cdots \quad a_1 \quad b_1 \quad \neg/7 \quad d_1 \quad \cdots \qquad ;$$

likewise, that for I2 is

$$\cdots \quad a_2 \quad \neg/8 \quad c_2 \quad d_2 \quad \cdots \qquad ;$$

that for I3 is

... $\underline{a}_3$ $\underline{b}_3$ ¬/7 ¬/6 ...
OR: ¬/13 ... ;

and that for I4 is

... $\underline{a}_4$ ¬/8 $\underline{c}_4$ ¬/6 ... .

(For conventions concerning the Delimiters on either side of the Skip Code, see section C3, "Positioning Symbols with the Skip Code.")

However, if the encoding proceeds system-by-system, that is, the information for all the Instruments contained within a given system is completely encoded Instrument by Instrument before encoding of the next system is commenced, it is not necessary to account explicitly for suppressed Instruments. Instead, the Instrument code of the first Instrument of the new system to be encoded will have the form

!In is a new system announcer

where n is an Unqualified or Qualified Instrument Identifier as defined above. Note that the Instruments within a system may be encoded in any order and that !In does not impute to In any specific vertical position within the system. When a code of the form !In is stated, it is assumed that all the information for all Instruments in earlier systems has been encoded.

In accordance with these principles, and given the same abbreviations and definitions for the page of score represented above, a system-by-system DARMS coding of that page could have

the form:

... !I1,$\underline{a}_1$ I2,$\underline{a}_2$ I3,$\underline{a}_3$ I4,$\underline{a}_4$!I1,$\underline{b}_1$ I3,$\underline{b}_3$!I2,$\underline{c}_2$ I4,$\underline{c}_4$!I1,$\underline{d}_1$ I2,$\underline{d}_2$ ...

|←——brace 1——→|←brace 2→|←brace 3→|←brace 4——→|

Segmented Encoding of Instruments (further remarks)

An Unqualified Instrument Code, regardless of how many individual parts may be subsumed under it and regardless of whether the same Instrument is elsewhere encoded with Qualified Instrument Codes, will of course cause the horizontal-position pointer to advance for all parts of the Instrument. On the other hand, a Qualified Instrument Code will cause the horizontal-position pointer to advance only for those parts of the Instrument specified by that Qualified Instrument Code.

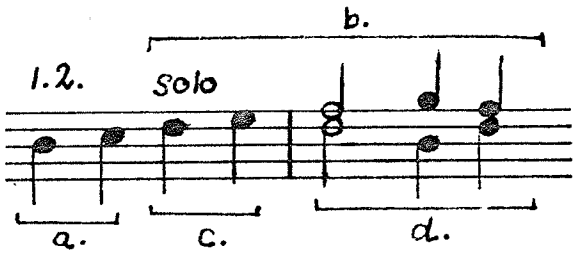
The reactivation of any Qualified or Unqualified Instrument will cause the pointer to be reset at the position at which the encoding for the previous segment of the part(s) had ceased (if the segment of code had been terminated with a Delimiter Comma) or the position corresponding to the next vertical slice of that part (if the segment of code had been terminated with a Delimiter Blank). In other words, it should be possible to extract, from a well-formed input-DARMS dataset, segments of code corresponding to each part of each Instrument which, after the Instrument Codes and Delimiters following them are deleted, should give the complete text of the given part from beginning to end. Therefore, the encoder should take particular care that the last character of one segment is precisely that which should precede the first

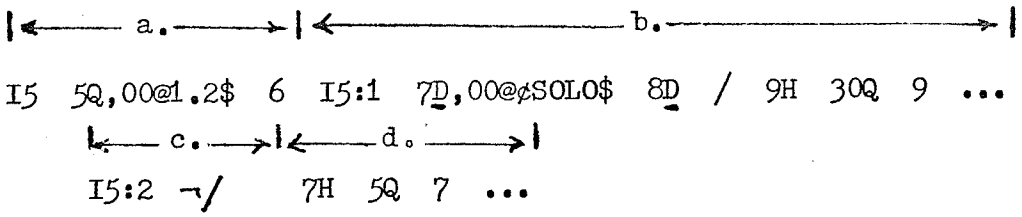
character in the next segment of the same part.

Sometimes it will happen that a particular part of the current Instrument is temporarily suppressed. In this case, the use of the Skip Code (C3) will cause the horizontal-position pointer to "skip" an amount corresponding to the length of the suppression. (See above, "Instrument Suppression in a Score.")

The principles concerning the segmentation of Instruments, Delimiters and the Push Code are further illustrated in the following example.

Example:





In this example, within

a. all parts play, therefore Unqualified Instrument Code is used. Code terminates with Delimiter blank, signifying that the next segment of all parts of I5 begin in the next vertical slice.

b. only the part for I5:1 is encoded. The position pointer for I5:1 is correspondingly advanced, but the "current

position" of I5:2 remains static at the next vertical slice after the second note of the example.

c. when the encoding of I5:2 is resumed, the Instrument Code I5:2 causes the horizontal-position pointer to be reset to the "current position" of I5:2 as defined above. Since I5:2 is suppressed for the last two quarter-notes of that measure, an appropriate Skip Code ~~(¬/ would also be valid in lieu of 5H)~~ is used to skip over the region containing no encodable information for I5:2.

d. the encoding of musical information in I5:2 is resumed.

Note that literals (E6) giving scoring particulars do not affect the Instrument Code currently in effect. It is the encoder's responsibility to state a new Instrument Code when necessary to conform with such instructions given in the score. Nonetheless, these literals must be encoded as such for a complete representation of the score.

If the segments of code I5:1 or I5:2 are to be joined, the concatenated segments of code for each would be:

(I5:1) 5Q,00@1.2$ 6 7D,00@¢SOLO$ 8D / 9H 30Q 9 ...

(I5:2) 5Q,00@1.2$ 6 ¬/ 7H 5 7 ...

(Note that stem information is incomplete when the Instrument segments are removed from their original contexts in this manner. However this deficiency could be easily remedied by the segment-catenation program itself.)

Change of Instrument by a Player

If a player is responsible for more than one instrument (e.g., Flute and Piccolo, several percussion instruments, etc.), the point in the score at which the player is instructed to change instruments is designated by a special Instrument Code of the form

Ip¬Iq

where p is the Instrument Identifier for the old instrument and q that for the new. See example in section B7, "Attach Codes for other DARMS Music Symbols."

Note that this special code is required in addition to the encoding, as a Literal (E7), of any verbal instructions in the score.

Encoding a Score in Multi-Instrument Vertical Slices

The following rules must be followed if a score is to be encoded as a series of complete vertical slices that cut across Instrument boundaries:

1. The information for each Instrument within a vertical slice must be followed by a Delimiter Comma. This also holds for the last-encoded Instrument within a slice;

2. When the vertical slice is completely encoded, the position pointer is advanced, and a Delimiter Blank implicitly inserted in the code for each Instrument string, by means of the code I0 (zero) followed by a Delimiter Blank.

Thus if a string quartet score were encoded in complete vertical slices, its DARMS representation would be of the general form

I1,...,I2,...,I3,...,I4,...,I0 I1,...,I2,...,I3, etc.

where the dots stand for musical information and the Delimiter Blank after I0 causes the position-pointer for all Instruments to be advanced to the next vertical slice.

3. If there is information within a vertical slice displaced so as to require a Double Comma Delimiter preceding its code (see conditions in section C1), then the vertical slice is encoded in more than one vertical "pass", the intra-pass Delimiter being the Comma (indicating exact

vertical alignment) and the inter-pass Delimiter being the Double Comma (to indicate that, although the information to follow occurs simultaneously with that of the previous pass, it is displaced slightly to the right). It is conceivable that there may be more than two passes required to fully encode a vertical slice; in any event, the number of Double Comma Delimiters will be less than the number of vertical passes within a single vertical slice.

Pg. A.10, line 2: Comma, not blank, should follow every Instrument Code (total of 10 changes).

Pg. A.11, second line of code under example, and
Pg. A.12, fifth line from bottom (code for I5:2):

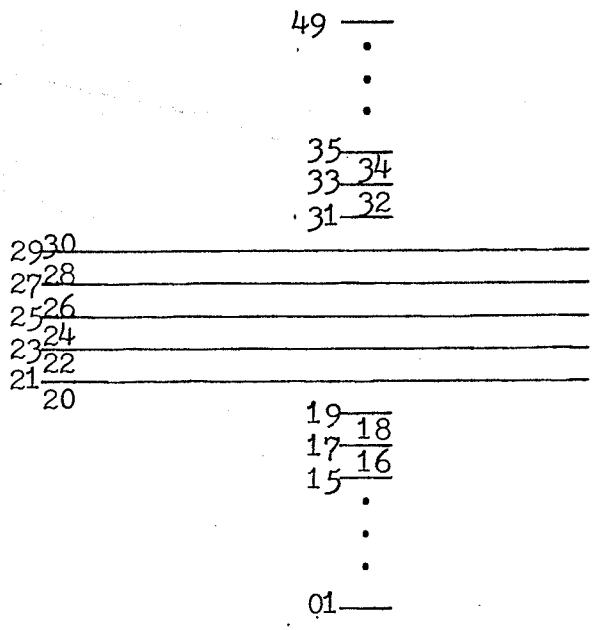
replace the incorrect ' H 7H'
with '¬/ 7H'
or 'W,7H'
Both are correct.

Pg. A.12, line 8: delete the parenthesized phrase.

Pg. F2.2, third line from bottom: change both occurrences of '!&' to '!''

Space Codes[1]

Space Codes specify the precise vertical placement of a Musical Symbol with respect to a particular stave. The Space Codes for the topmost stave of an Instrument[2] are:



Positions on subsequent (lower) staves within the same system of the same Instrument are referenced by adding, for each additional stave, 50 to the Space Codes of the corresponding positions of the first stave. Thus, in a typical organ score, the Space Codes for the lines of the second ("left hand") stave are (from bottom to top) 71, 73, 75, 77, and 79, and those for the pedal stave are (from bottom

---

[1] The metalinguistic symbol for Space Code is σ (lower case sigma).

[2] The term "Instrument" is used throughout to designate music for a given Instrument Code (A).

to top) 121, 123, 125, 127, and 129. Note that staves are added below, not above; the special case of "ossia" staves is treated in section F2.

Abbreviations

1. Default Space Codes. Many categories of symbols have been assigned default values for σ, some of which are given below. See the pertinent sections of the manual for details.

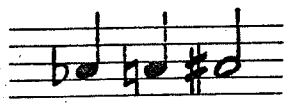
Symbol Category	Default Value of σ
Rests (D4)	depends on denomination; see section D4
Clefs (E1)	G clef 23 F clef 27 C clef 25
Meter Signatures (E3)	25, (and, in the case of multi-stave Instruments, 75, 125, ... as required)

2. "2-Suppression." Space Codes 20-29 may always be abbreviated by dropping the tens digit. Space Codes 00 to 09 must therefore always be supplied with leading zero to avoid ambiguity. A Space Code of 9 will always be interpreted as 29, not 09.

3. "Sigma Suppression." If two successive single notes or two successive chords, regardless of what else may intervene between them, have the same Space Code ~~(or Pseudo-Space Code)~~, the second Space Code ~~(or Pseudo-Space Code)~~ may be suppressed as long as there does not intervene 1) a change of Instrument (A) or 2) an Equate Code Definer prefix (F6). For any given note, Sigma Suppression and Delta

Suppression (D4) are mutually exclusive conditions. Change of accidental on two successive notes or chords with identical Space Codes does <u>not</u> preclude the use of Sigma Suppression.

Example:



4-Q *Q #H

4. If, in encoding a multi-staved Instrument, the encoder wishes to encode a sizeable segment of a stave other than the topmost one, the Stave Transposition Control (F4) may be employed. This permits the encoding of any stave as if it were the topmost stave, thus permitting the use of "2-suppression." The saving in characters may therefore be considerable.

Double-Space Code

For symbols that have a variable vertical extent and are perpendicular to the stave, and thus require that both lower and upper terminal points be encoded in order to achieve an accurate DARMS representation, the Double-Space Code is to be used instead of a Space Code (B1).

The Double-Space Code has the form

$$\sigma_1 | \sigma_2$$

where σ_1 is the lower boundary and

σ_2 is the upper boundary of the symbol in question.

This code may be used both to define the vertical extent of certain symbols, for example arpeggio indications (facilities for which are given in sections D11 and E8); ~~It is also used~~ and to indicate the height (size) of certain symbol types, for example Literals (B6), Meter Signatures (E3) and the like. It may also be used in encoding obliquely positioned symbols that are "stretched" (C3). (e.g. (g ! ! s s . !))

77|5!M3:4

The following symbol types may employ the Double-Space Code:

- Meter Signatures (E3)
- Barlines (E4)
- Dynamics (E5)
- Literals (E6)
- Some Dictionary Symbols (E8)
- Rests (D4)

Vertical Structures

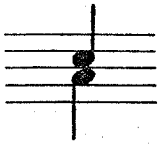
There are three ways to encode information which is vertically aligned in the score. Generally, such alignment indicates that the events referenced take place simultaneously.

1. Full encoding, using the comma as Delimiter.

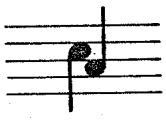
This format, which is valid for all simultaneities, employs a Delimiter Comma (C1) between items of information that, in the original score, are vertically aligned. Although it is not necessary to encode the Notes (which proper noun, in DARMS, encompasses both single notes and chord structures) or other symbols in any particular order, the "canonical" ordering is from bottom to top. For the order of encoding within Notes see section D1.

N.B. Unless otherwise indicated by the use of Stem Identifier digits (D6) all stems in the same time-slice are assumed to be distinct.

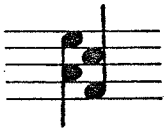
Examples:



4QD,6U



6Q,,4



4QD1,8D1,,2U1,6U1

Note: the Double-Comma Delimiter is explained in section C1.

2. Chord-Structure I: Space-Pattern Format.

This method, more efficient than full encoding (above), facilitates the encoding of chords.

Conditions for Use:

a. All notes in the chord must be attached to the same stem.

b. The notes of the chord are encoded as a series of cells, one for each note, which are bounded by a vertical bar on both left and right.

c. The minimum amount of information in each cell is the Space Code and, if relevant, Accidental Code. In addition, all information not pertaining to the entire chord is represented within the cells to which the information applies in the order established for single notes. Thus, a single cell could theoretically contain the following ordered codes:

Space Code (always required)

Accidental Code (required if accidental present)

Notehead Code

Duration Code (if different from chord as a whole)

Tie Code {Stem, Trem, Beam}: N/A

Articulation Code
Appropriate Dictionary Codes
Fingering Code
Ornament Code
Slur Code
} (internal ordering determined by proximity to notehead)

d. The ordering of information common to the chord as a whole is based on the treatment of the entire

(and Accid. code)

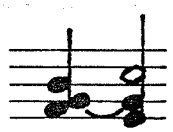
cell complex as if it were the Space Code in the encoding of a single note. Thus, short-form beam openings precede the cell complex while codes for all other attributes follow the cell complex in the normal order of encoding as specified in section D1.

e. A Tie Code outside the cell complex implies that <u>every</u> note of the chord has a tie; it does not imply that the other ends of these ties are necessarily in the same time-slice.

f. If there is a conflict between information within a cell and that outside the cell complex (e.g., a Duration Code appears in one cell that conflicts with a Duration Code following the last note cell as in the example below), then the information within the note cell takes precedence <u>for that note only</u>.

g. Space Codes for notes encoded in Space-Pattern Format may <u>under no circumstances</u> be suppressed. However, a Chord-Structure Reference (B5) may replace the entire cell complex since the Space Codes and Accidentals (<u>but no other attributes</u>) can be computed from it.

Examples of Space-Pattern Format:



|2|3J|5|Q |1|3|6H| (N.B. Duration 'Q' propagates from first chord to first two notes of second chord.)

3. Chord-Structure II: Base-Increment Format.

Base-Increment Format may be employed ONLY if all information except Space Codes and Accidentals is common to all notes of the chord. When this is the case, Base-Increment Format is normally the most efficient method of chord representation.

This method differs from the others described above in that, except for the bottom note, the intervallic rather than the Space Code structure of the chord is encoded. Such an encoding takes the form

$$\sigma_0 \alpha_0 \pm i_1 \alpha_1 \pm i_2 \alpha_2 \cdots \pm i_n \alpha_n$$

where σ_0 is the Space Code of the lowest note ("base") of the chord;

$i_1 \cdots i_n$ represent the amount by which the Space Code of any note exceeds that of the next lower note, the notes being encoded from bottom to top. Note that the value of any i will be 1 less than the numerical category of harmonic interval, e.g., if the musical interval being encoded is a fifth, then the corresponding $i = 4$; and

$\alpha_0 \cdots \alpha_n$ are Accidental Codes as required.

The resulting string is assigned the position of a Space Code in the encoding stream, viz., immediately in front of the Notehead Code.

Conditions for Use:

a. Among the notes of the chord, only Space

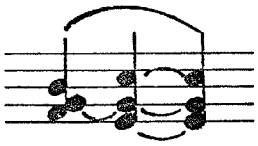
Codes and Accidental Codes may vary.

b. If ties are associated with the chord, there must be one on <u>every</u> note of the chord, since that is what is implied by the use of a Tie Code in Base-Increment Format. If some notes of the chord have ties and others do not, then Base-Increment Format may not be employed.

c. From a. above it follows that all notes of the Chord Structure are attached to the same stem.

Examples of Base-Increment Format:

1+2#+2Q



|2|3J|5|QL1 1+2+3J 1+2+3L2

<u>OR</u>

|2|3J|5|QL1 1+2+3J QL2

The second of these examples illustrates the appropriate use of both Space-Pattern and Base-Increment Formats.

d. If the chord form is repeated in the encoding stream without any single notes, other chords, Equate Code ~~Definer prefix~~ (F6) or Instrument Code (A) intervening, the Base-Increment representation itself may be suppressed in the code for the repetition. Thus, there is "sigma suppression" ~~for chords~~ as well as δ-suppression for chords. This is exemplified in the second DARMS representation of the second example given in c. above. Also, Space-Pattern chords will have sigma vector propagated forward.

Chord Structure Definers

Frequently found chord-forms may be defined and referenced in abbreviated form. This applies to both types of Chord Structures (Space-Pattern and Base-Increment) discussed in the section on Vertical Structures (B3).

A Chord Structure Definer is of the form

!|i|

where i, an integer, is the Chord Structure Identifier. (If i has been used as a Chord Structure Identifier previously, its former meaning is superseded by the new.) The Definer is placed immediately in front of the encoding of the Space Code information for the chord, with no blank or other Delimiter intervening.

N.B. Chord Structure Definers stand only for the Base-Increment intervallic relationship of either chord type, whether encoded explicitly (Base-Increment representation) or implicitly (Space-Pattern representation). A Definer does not imply any accidentals; these must be supplied

1. in the Chord Structure representation following the Definer (for that occurrence only) according to the principles set forth in the discussion of Vertical Structures (B3, types 2 and 3); and
2. in subsequent Chord Structure References in

the manner discussed in section B5.

Examples of Chord Structure Definers are given in section B5.

Chord Structure Reference

Every appearance of defined Chord Structures may be encoded by a Chord Structure Reference of the form

$$\alpha_{\underline{a}}\sigma_{\underline{a}}\alpha_{\underline{b}}\sigma_{\underline{b}}\ldots|\underline{i},\sigma_0,\rho|$$

where $\underline{i}$ is the Chord Structure Identifier,

σ_0 is the Space Code of the lowest note of the chord, and

ρ is the number of times that the chord is stated in immediate succession.

The α's and the σ's preceding the first vertical bar stand for accidentals and Space Codes, respectively, and are discussed further below. When σ_0 or $\underline{i}$ are suppressed, the value(s) last employed in any Chord Structure Definer or Reference are assumed. If ρ is 1 (i.e., if the Reference stands for a single chord) the Reference may be of the form $|\underline{i},\sigma_0|$. However, the commas must be encoded unless the omissions start at the right end. Hence, for a single immediate repetition of a Chord Structure at the same Space Code level, the code

||

may be employed.

Chord Structure References, like the code for the Chord Structures themselves, are assigned the position in the encoding stream normally used for the Space Code and Accidental Code. Attributes of the chord other than the

Base-Increment relationship (e.g., beams, duration, articulation, etc.) are not inferrable from Chord Structure References and must be explicitly encoded each time.

Whenever the chord being represented involves accidentals (α), the Accidental Codes and their Space Codes (σ) prefix the main part of the Reference, as shown above. Note that the Accidental Code precedes the corresponding Space Code (as in Codes for non-orthodox key signatures, E2), the reverse of the procedure for the normal encoding of accidentals. Note also that, when $\rho > 1$, certain attribute classes encoded with the Chord Structure Reference are assumed to be physically associated with only the first or (in the case of short beam closings) last chord represented by the Chord Structure Reference and therefore absent from the notation for the other chords represented by the same Chord Structure Reference. These are:

Beam openings and (closings assoc'd. only with last chord)

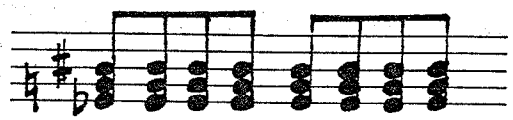
Accidentals

Slurs

Fingerings

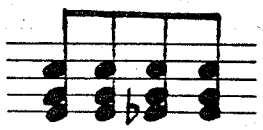
All other attributes (incl. artic.) are assumed to be common to all the chords represented by the Chord Structure Reference when $\rho > 1$.

Examples:



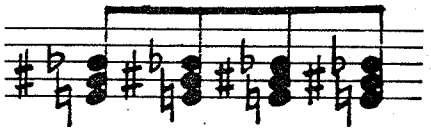
(!|3|1-+2*+2# |,,3|) (|,,4|)

BUT



(!|4|1+2+3 || -1|,,2|)

In the case of



~~Equate Codes (F6) are advised.~~

~~incorrect: (=5=1*+2#+2-^=$^=5,3$) → (1*+2#+2-^^^^^)~~

The most efficient encoding is: '(1*+2#+2-^E^E^E)' ~~not desired~~ =
although EquateCodes (F8) may also be used:
(=5=1*+2#+2-=$^=5$^=5$^=5$)

OR

(=5=1*+2#+2-^=$^=5,2$^1*+2#+2-)

↑
this blank
not necessary

Note that when accidentals must be indicated on the upper notes of a short doubled passage, it is just as easy and perhaps slightly more efficient to use Base-Increment format (B3.4).

Pseudo-Space Codes

Whereas certain categories of musical symbols (notes, clefs, etc.) require that, for their meaning to be clear, their vertical position with respect to a stave be precisely represented, other categories (tempo indications, dynamics, rehearsal numbers, texts, etc.) do not. However, it is nonetheless often necessary to indicate the approximate location of symbols of the latter type, and to satisfy this need a set of DARMS symbols called Pseudo-Space Codes is available.

A Pseudo-Space Code is of the form

$$\sigma' \pm n$$

where σ' is a multiple of 50 as determined by the following table:

σ'	Interpretation
00	"Above the topmost stave"
	"Below the stave" (for a one-stave Instrument)
50	OR
	"Approximately midway between the first two staves" for an Instrument having two or more staves)
	"Below the bottom stave" (for a two-stave Instrument)
100	OR

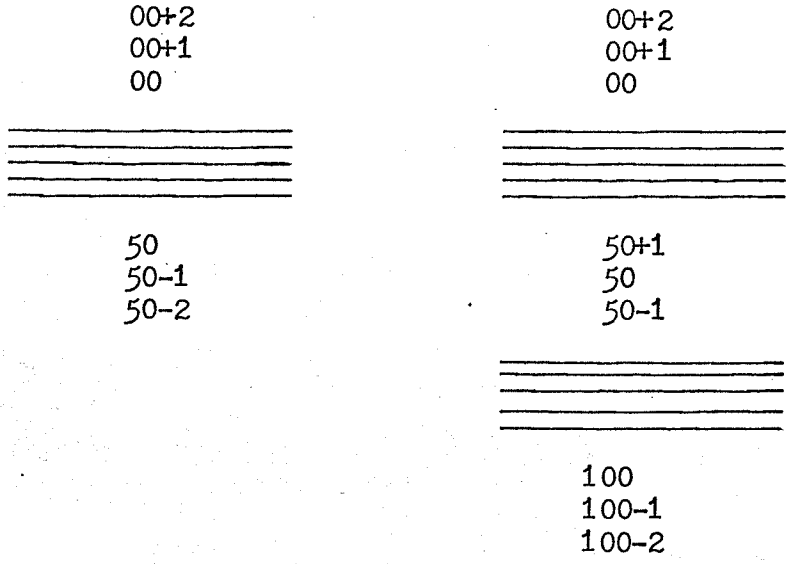
"Approximately midway between the second and third staves" (for an Instrument having three or more staves)

etc.

---

and where n is an integer 0, 1, 2, . . . specifying distinct horizontal layers of information within the general loci "above," below," or "midway between" staves. If n = 0, then the signed integer portion of the Pseudo-Space Code may be suppressed; a code of this form will refer to the layer closest to the stave (whether above or below) or, in the case of a layer between staves, the approximate midpoint between the staves in question. Additional layers, as required, may be specified by the signed integer, +n indicating the n^{th} layer above the 0^{th} layer and -n indicating the n^{th} layer below. This facility is particularly useful for the encoding of strophic texts. See section E6.

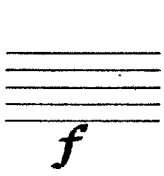
The above description is summarized below:



Note that because the multiples of 50 are employed in Pseudo-Space Codes they are thereby automatically removed from the set of theoretically available Space Codes for notes.

Pseudo-Space Code values are assumed by default for the following symbol types unless another Space Code or Pseudo-Space Code is explicitly given:

Symbol Type	Default Value of σ'
Dynamics (E5)	50
Literals (E6)	50
Some Dictionary Symbols (E8)	see "s Default" column of Dictionary Table.



Allegro

00!@¢ALLEGRO$

May 'A' ever be suppressed? No.

Attach Prefix

The Attach Prefix is used to indicate the graphic proximity--but not contiguousness--of one symbol (here called the "attached symbol") to another (the "reference symbol"). It has the general form

Ac

where c is a compass direction of up to three letters (e.g., N or SSW); c may be suppressed if it is equal to the default value for the attached symbol (as found in the Dictionary Table in section E8). Although the value of c may indicate that a symbol is placed to the left (W) or right (E) of the reference symbol, it does not affect the horizontal-position pointer in any way.

The Attach Prefix does not provide the specificity of the Space Code and is therefore used only in those cases where such precision is neither necessary nor desired. Thus, while it conveys a clear indication of graphic (and, likely, musical) association, it makes possible the representation of the approximate placement of symbols whose meaning is not dependent on precise vertical or horizontal position.

Attach Codes

The concatenation of an Attach Prefix and the code for one or more attached symbols results in an Attach Code, of

which there are three types: Simple, Multiple and Nested. A separate Attach Code is required for each value of c that may pertain to symbols attached to a given reference symbol.

A. Simple Attach Codes

A Simple Attach Code has the general form

Acs

where s is one DARMS Music Symbol or attachable Note attribute which is understood to lie in direction c from the point of reference. (Attachable Note attributes and points of reference are discussed below but, in general, the point of reference is the DARMS Music Symbol or Note attribute immediately preceding the Attach Code.)

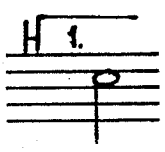
B. Multiple-Attach Codes

A Multiple-Attach Code has the general form

$$Ac|s_1 \cdots s_n|$$

where $s_1 \cdots s_n$ is the concatenation of n codes (optionally separated by commas) representing the n attached symbols, ordered in proximity to the point of reference, that lie in direction c from the point of reference.

Example:



6HANW|@1.$?H|

C. Nested Attach Codes

A series of Simple or Multiple-Attach Codes, differing perhaps in c, is always understood to be associated with the reference symbol encoded immediately to the left of the first Attach Code in the series. The Nested Attach Code, however, permits the representation of a symbol as being attached to another attached symbol.

The Nested Attach Code has the general form

$$!Acs\$$$

where A and c are as above,

s is the symbol being attached, and

! and $ establish and cancel, respectively, one degree of Attach Code nesting.

It is possible to carry the nesting to any degree by deferring the $. Thus, to take a highly unlikely but nonetheless useful example, a DARMS substring of the form

$$s_0 Ac_1 s_1 !Ac_2 s_2 !Ac_3 s_3 \$Ac_4 s_4 \$Ac_5 s_5 s_6$$

where s_0 through s_6 are DARMS Note attributes or Music Symbols and

s_1 through s_5 are attached, signifies that

s_1 is attached to s_0	(zero-degree nesting) ,
s_2 is attached to s_1	(first degree nesting) ,
s_3 is attached to s_2	(second degree nesting) ,
s_4 is attached to s_1	(by virtue of the first $, which re-establishes first degree nesting) ,
s_5 is attached to s_0	(by virtue of the second $, which cancels the one remaining degree of Attach Code nesting).

It is to be pointed out that any s in the formulation above may be taken as in a Multiple-Attach Code, that is, in lieu of s_i there may be a string of the form

$$|s_m \cdots s_n|$$

Likewise, Nested Attach Codes are permitted within the vertical bars of Multiple-Attach Codes. Generally, this feature is to be employed when the compass direction of the Nested Attach Code(s) is different from that immediately in front of the left vertical bar.

Attach Codes for Notes

Under normal circumstances Note attributes need not be attached; rather, they are simply stated according to the Order of Encoding (D1). In cases where the attributes are positioned in two or more directions or where there exists some other condition that cannot be adequately represented by the normal encoding order, the following categories of Note attributes can be attached to other attributes of the same Note to yield an accurate representation:

~~Accidentals~~
Articulations (D9)
Dictionary symbols associated with Notes (E8)
Fingerings (D10)
Ornaments (D11)
Slurs (D12)

It is also permitted to encode a Literal (E6) as being attached to a Note attribute.

The attribute to which a symbol is attached must be explicitly encoded to avoid ambiguity, even though it could

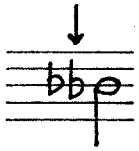
otherwise be suppressed. A list of attributes and the point from which the value of c is to be determined follows.

attach receptor attribute

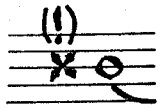
Space Code: nothing may be attached to the Space Code.

Accidental: the accidental or, if there are multiple accidentals for a single note, the accidental with which the attached symbol is most likely associated.

Examples:



5--AN?ASH



5##AN@(!)$WL

(Note that the two characters standing for a double-sharp are treated as a syntactic unit.)

Duration or, in case of a special notehead, Notehead: the notehead. (This is the most likely reference point for many attachable symbols.)

Tie: the point at which the tie begins or ends, whichever is the case.

Stem: the end of the stem farthest from the Notehead.

Examples:

4QUAN|'L|

6QDAS|'L|

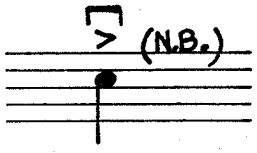
One-Note Tremolo: the midpoint of the slash farthest from the notehead.

Beam (Long form only): the end of the stem where it terminates in the beam (of its own system) farthest from the notehead. See also the special use within

the Skip Code illustrated in "Attach Codes Using the Skip Code" below.

Articulation: the articulation mark.

Example:



6Q>AE@(ØN.ØB.)$?D

Slur: the point at which the slur begins or ends, whichever is the case.

Ornament: the ornamental sign.

Fingering: the fingering digit, fingering parenthesis, or fingering slur (beginning or end, whichever is the case) last encoded.

Dictionary Symbol: the Dictionary symbol. If the symbol is of variable horizontal or vertical extent, the reference point is that symbol in its minimal extent in either direction.

Figured-Bass: the symbol farthest from the notehead.

If one or more attachable symbols lie in one direction from the reference point and others lie in another, then, as stated earlier, a separate Attach Code is required for each value of c (assuming that the symbols are not in default position and must be attached). Thus, a given attribute, encoded in normal position within the code for a Note, may serve as the reference symbol for a string of Attach Codes.

(To encode several symbols attached in the same direction, see paragraph B on Multiple-Attach Codes above; to encode symbols attached to other attached symbols, see paragraph C on Nested Attach Codes above.)

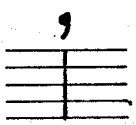
Attach Codes for other DARMS Music Symbols

With regard to DARMS Music Symbols other than Notes, the Attach Prefix may be employed as a substitute for the Space Code in the following categories of symbols:

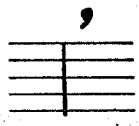
attachable:
- Clefs (E1)
- Dictionary Symbols (E8)
- Dynamics (E5)
- Literals (E6)
- Meter Signatures (E3) .

One or more Attach Codes (Simple, Multiple- or Nested) may be suffixed to a given reference symbol. Note, however, that while many of the attachable categories are syntactically complete DARMS forms when they employ Space Codes or Pseudo-Space Codes (which may, in turn, be defaulted), they lose this independence by virtue of the Attach Prefix, which mandates that the Attach Code be directly appended (with no Delimiter intervening) to the code for the reference symbol.

Examples:

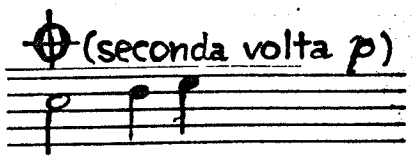


/A?,



/ANE?,

{Barlines, } : attach receptors

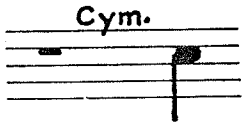


6H,?GAE@SECONDA VOLTA |VP|)$ 7Q 8

The Attach Code as an Independent Token

The sole exception to this rule occurs when the reference symbol is the printed stave itself. Since there is no DARMS code for the stave, an Attach Code standing by itself (i.e., preceded by a Delimiter) will represent a symbol attached to a stave. Thus, the Attach Prefix AE following a Delimiter will imply a position to the right of the end of the printed stave and AN and AS will be equivalent to the Pseudo-Space Codes (B6) corresponding to "above the topmost stave of the Instrument" (viz., 00) and "below the bottom stave of the Instrument" (viz., a multiple of 50). (If an independent Attach Code is encoded within Ossia Mode (F2), it will be assumed to belong to the ossia stave referenced by the explicit or implicit Space Code most recently encountered in the encoding stream.)

Example:



RW AN@¢CYM.$∧ I12-I16∧ 6Q

Note that the change of Instrument from I12 to I16 (the latter here signifying "cymbal") requires a change of Instrument Code as explained in section A, "Change of Instrument by a Player."

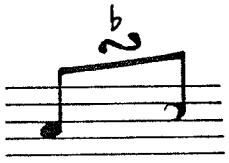
Attach Codes Using the Skip Code

If a symbol must be encoded as being associated with a symbol of horizontal extent (including the Note attributes Tie, Beam, Slur, Fingering and Ornament) at a point subsequent to its beginning, the Skip Code

¬d

as defined in section C3 is suffixed to the Attach Prefix. This represents that the attached symbol is positioned at a specific horizontal position within the slur, beam or other symbol of horizontal extent. The position pointer, however, is unaffected, being, in effect, "anchored" at the attack point of the note..

Example:



4(B1AN¬SO?!AN-$ 6B2)

↓ accids may be attached

* another specifier of "anchored" position precedes the type preceded by a comma

Note "skip" within Attached Note Attributes

Delimiters

Delimiters are used to separate tokens in a DARMS string. In addition to this syntactic function, they also are used to control the forward motion of the imaginary horizontal-position pointer that marks the "current position" in a score being encoded.

There are three Delimiters:

1. Comma. The Delimiter Comma does not cause any change in the horizontal-position pointer. It is therefore used between codes for symbols that are vertically aligned or after other codes--such as Instrument Codes, Stave Transposition Codes or Groupette Definers--that do not represent musical symbols, per se. In short, a DARMS code of any type should be terminated with a Delimiter Comma except when the pointer is to be advanced.

2. Blank. The Delimiter Blank causes the horizontal-position pointer to advance to the position of the next-encoded musical symbol to the right. Note that the "next encoded musical symbol to the right" will not necessarily occur in the next vertical slice to the right, since it is possible to skip over information by means of Linear Decomposition Mode and the Skip Code. Normally, however, the distance advanced will amount to that re-

Delimiters

Delimiters are used to separate DARMS tokens from each other in the encoding stream and to indicate the vertical alignment or non-alignment of information contained in any two successive DARMS Music Symbols in the same Instrument (Qualified or Unqualified).

There are three Delimiters:

1. Blank. A Delimiter Blank[1] following a DARMS representation of a symbol signifies that the information represented by the next DARMS-coded symbol in that Instrument is, in the score, to the right of the former; the horizontal-position pointer is therefore advanced to the vertical plane of the latter symbol. If a Delimiter Blank follows a DARMS code that does not represent a symbol in the score--for example, a Groupette Definer or Size Code--then the pointer is unaffected by the Blank. If there is a series of such codes, each separated by Blanks, none of the Blanks will have any effect on the position pointer. Thus, in

!+50 5Q3:W !& 6Q

only the Blank after 6Q causes the horizontal-position pointer to advance to the right.

2. The Delimiter Comma serves as a syntactic separator in DARMS strings (as does the Delimiter Blank), but the Comma

[1] A Delimiter Blank consists of one or more blanks in succession.

presented by the smallest duration given within the vertical slice encoded immediately before the Delimiter Blank.

A Delimiter Blank may be represented by a string of one or more blanks following a complete DARMS code. Two or more blanks in succession are considered equivalent to a single blank. The Delimiter Blank must not, of course, be confused with blank characters within a Literal.

3. Double Comma. Information between a Double Comma and the next Delimiter is defined to occur at the same temporal position as that represented by the DARMS-coded music symbol immediately preceding the Double Comma; however, the music symbol represented by the code following the Double Comma is understood to be offset slightly to the right for purely graphic reasons.

Exceptions: The Double Comma is not required (since displacement is always assumed) when two notes 1) have the same stem (explicit or, in the case of whole notes, implicit) AND 2) have Space Codes differing by 1 (i.e., the notes are a second apart).

All these conditions are illustrated in the examples which follow.

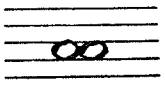
NOTE: There are special uses of Delimiters, based on the principles given in this chapter, involving Instrument Codes (A: "Encoding a Score in Multi-Instrument Vertical Slices"), Linear Decomposition Mode (C4), and the Skip Code (C3.3ff). See the places cited for details.

does not cause the position pointer to advance. The codes for a group of symbols, vertically aligned in the score and encoded in immediate succession within a DARMS string, will therefore be separated by Delimiter Commas. Delimiter Commas may also follow DARMS codes that do not represent actual symbols in a score; in these contexts (as is also the case with Delimiter Blanks) the Commas serve as syntactic separators with no positional significance.

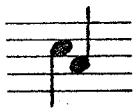
*NOTE: The reader is advised to refer to sections A, C3 and C4 for special information regarding Delimiters following Instrument Codes, Skip Codes and Linear Decomposition Mode Codes, respectively.

3. Double Comma. Information between a Double Comma and the next Delimiter is defined to occur at the same temporal position as that represented by the DARMS-coded music symbol immediately preceding the Double Comma; however, the music symbol represented by the code following the Double Comma is understood to be offset slightly to the right for purely graphic reasons. However, when two notes are 1) on the same stem (explicit or implicit) and 2) have Space Codes differing by 1 (i.e., span the musical interval of a second), the graphics of music dictate that one of the notes be displaced laterally. Such displacement is always understood to exist under the stated conditions and is therefore not to be represented by the Double Comma.

Examples:

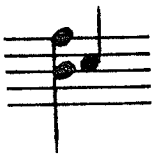


4W,,4



6Q,,4

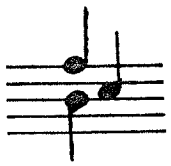
1+2+6Q,,RE



5QD1,9D1,,6U

NOT

5QD1,,6U,9D1



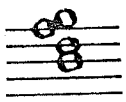
5QD1,9U1,,6U2

NOT

5QD1,,6U1,9U2

4+1QU

(no Double Comma)



5+2+2+1W

Symbols of Horizontal Extent: Identifiers

A music or graphic symbol that begins in one time-slice in a score and ends in another is said to have horizontal extent. Both the beginning and end points are normally encoded and, in some cases (e.g., slurs that change direction), intermediate points as well. Furthermore, some symbols, such as horizontal lines, may be represented by a single code employing the Stretch Operator (C3).

In order to allow for variability in horizontal extent and to distinguish the beginning of a symbol from its end, integer identifier pairs are used in conjunction with codes that designate the symbol in question: an odd-numbered identifier (equal to or greater than 1), incorporated into the code for a particular symbol, designates that code as marking the beginning of the symbol; the next higher even integer is then used to mark the termination of the symbol.

When the termination of a symbol of horizontal extent is encoded the identifier pair immediately becomes available for reuse within that Instrument. Thus a string of the form

6QL1 ... 4HL2L1 ... 2QL2

is valid. However, were the two slur codes L2 and L1 reversed in the code for the middle note, the encoding would not have been valid. For convenience in error-checking and -correction,

it is nonetheless recommended that identifier pairs not be immediately reused for a given symbol type. However, the same identifier pair may be "active" simultaneously in the same Instrument for several different DARMS symbol categories or Note attributes, without causing ambiguity. Note, however, that the Dictionary is treated as a single category, despite the multiplicity of symbols therein.

Although most identifier pairs are "local" to a DARMS syntactic category within a given Instrument, a given pair may cross Instrument boundaries when the associated symbol does. (See the example in section C3, "Codes Incorporating Both the Push Operator and Instrument Codes.") This means, of course, that the same identifier pair cannot be already active in the other Instrument for that symbol type.

Symbols having more than two end points

If a slur, glissando or other similar symbol has two or more openings but a single closing, it is possible to render this information accurately in DARMS.

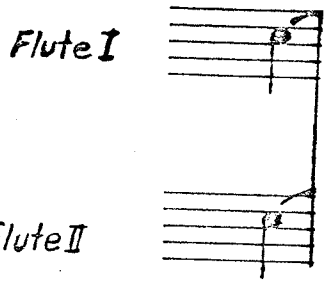
1. If the symbol begins in different parts of the same Instrument (i.e., different Qualified Instrument Codes are involved) but end at the same point, graphically speaking, then within the same Qualified or Unqualified Instrument

a. the codes for the two openings points of the symbol are given the same odd-numbered integer identifier and that for the closing point the next-higher even integer

or b. the codes for the opening points may be given different odd-numbered identifiers and those for the closing point will involve the corresponding even-numbered identifiers. Just as the terminations are graphically congruent, so will all termination codes be concatenated together as one compound code.

2. In all other cases, the method outlined in b. immediately above must be employed.

Example:



(a) I20:1, ... 6QL1 / I20:2, ... 6QL3 ... I20:1,2,5QL2L4

(b) same except where noted 6QL1 ... I20:1.2,5QL2

(Note: If there are only two flutes in the score, then I20 may be substituted for I20:1.2 .)

If a slur, glissando, or other similar symbol has one opening but two or more termination points, the opening point of the

symbol is encoded with an odd-numbered identifier as normal and all the end points are encoded with the same next-higher even integer identifier. The conditions under which this may occur, viz., multiple encodings or redistribution of parts of an Instrument over a larger number of staves, will be detected by the DARMS system software.

Example:

6QL1 /,?)@1.$5 5QL2 ... :/!/,?)6,,?(@2.$7 4QL2 ... ?(8

If the penultimate example above were revised so that the slur began in Flutes I and II on a single stave and terminated on two staves (Flute I and Flute II, respectively), the DARMS representation would be as follows:

I20:1.2,5QL1 ... I20:1, ... 6QL2 ... I20:2,6QL2 ...

Codes Employing the Push Operator

The character ¬ in DARMS is called the Push Operator. It is employed in various contexts to indicate that a music symbol of some sort is positioned to the right of the current alignment of the position pointer, is "stretched out" over a horizontal distance corresponding to a specific duration, or begins in one DARMS Instrument and ends in another. It ~~can also be~~ is also used to signify that a single player replaces one musical instrument with another.

The Skip Code

The Skip Code is used to advance the horizontal-position pointer an absolute distance or durational equivalent even when this involves "skipping over" certain other symbols. (These will have already been, or will be, legitimately encoded elsewhere in the data string.) The Skip Code is especially useful for the placement of Dynamics (B5) and Literals (B6) and is often essential for encoding in Linear Decomposition Mode. It is also employed to represent temporary suppression of an Instrument when encoding Instrument by Instrument (A).

The form of the Skip Code is

¬d

where d is

Skip found in <Fingering>, <ALT_ED_ACC ID>, etc.

A. Any valid Duration Code (see section D4) or concatenate of Duration Codes. Except where occurrences of Duration Codes W and Z are concatenated, in which case the interpretation defined in section D4 is understood to govern, concatenation will mean that the value of d is the sum of the durations denoted.

Examples:

H E.. WW (= breve) HQE3S

N.B.: The Skip Code, not being a Note, is not subject to the rules of Delta Suppression (D4).

OR

B. A code of the form

/n

where n is an integer, specifying the n^{th} barline to the right. If n = 1 , then it may be suppressed.

Examples:

/2 /20 /

OR

C. A concatenation of B. and A. above, in that order.

Examples:

/5QE /5Q3 /Q.

OR

D. A code of the form

n$

where n may be any integer or decimal representing the number of units, each unit being equal to the width of the

when '¬ begs' is used, a ¬/ or ¬δ may also have to be used to keep HPP advancement clean.

quarter note. This facility is necessary for encoding certain 20th century scores where nothing occurs although space is intentionally left blank on the stave.

Examples:

5$ .5$ 32.4$

¬begs has no effect on HPP duration counters and no duration-filler adjustment (¬δ) is necessary (bec. I believe ¬begs will always be anchored)

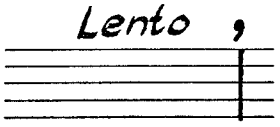
Positioning Symbols with the Skip Code

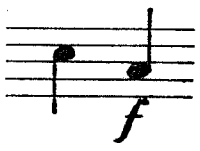
The principal use of the Skip Code is the placement of music symbols to the right of the "current position." Since the movement of the horizontal-position pointer is also affected by the Delimiters that are found on either side of the Skip Code it is important to understand their function.

If a Skip Code is followed by a Comma (the normal case), the music symbol whose code immediately follows the Comma will be assumed to be vertically aligned with the position (or corresponding symbol) located at distance d to the right of the point from which the skip was made.

as of 9/82 this is no longer true

Examples:



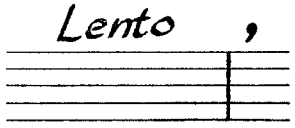


00@¢LENTO$ ¬/,?, !&,6Q 4 &,¬Q,VF

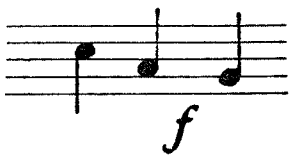
If a Skip Code is followed by a Blank, the music symbol whose code immediately follows the Blank will be assumed to be the very next symbol to the right of the position (or correspond-

ing symbol) located at distance d to the right of the point from which the skip was made. It is understood that no other symbol encoded previously (i.e., in another layer of Linear Decomposition Mode for that Instrument) lies between the symbol being encoded by the use of the Skip Code and the vertical slice that would have been represented had a Comma followed d .

Examples:



00@¢LENTO$ ¬/ ?,

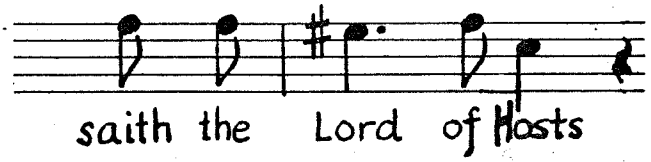


!&,6Q 4 3 &,¬Q VF

For the discussion concerning Delimiters that precede the Skip Code, it will be useful to think of the Skip Code, its right-hand Delimiter (just described) and the code for the music symbol that follows that Delimiter to be a single unit, here called a "Skip-Symbol pair."

If a Skip Code is preceded by a Blank (the normative situation), the Skip Code is said to be "unanchored," and the horizontal-position pointer will be understood to be permanently advanced (at least within that layer of Linear Decomposition Mode) by the Skip Code.

Example:



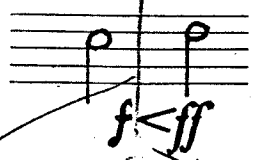
@...SAITH ¬E,THE ¬E,¢LORD ¬Q.,OF ¬E,¢HOSTS ...$

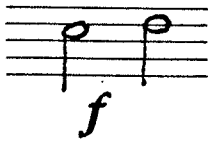
(This example represents the encoding of the text only and assumes that Note information is encoded in another pass through the ~~data~~ passage in Linear Decomposition Mode.)

If a Skip Code is preceded by a Comma, the Skip Code and the position pointer are said to be "anchored," meaning that the position pointer is advanced only temporarily by the Skip Code (by an amount controlled by d and the Delimiter following the Skip Code, as explained above), immediately is terminated by a Delimiter. There may be any number of such Skip-Symbol pairs in series, each preceded by a Comma and each causing a temporary advance and immediate return of the position pointer. Thus, the distance specified in each Skip Code of such a series will be reckoned from the same anchored position.

When the Delimiter terminating a Skip-Symbol pair is a Blank, then the pointer, after its return, is released from its anchored mode and then is advanced an amount appropriate for the DARMS music symbol immediately to the left of the (first) Skip-Symbol pair. In other words, the position pointer is advanced as if the Skip-Symbol pair(s) had not been there at all.

Examples





6H,¬Q,VF<1,¬QE..,V<2 7H,VFF 6H,¬Q,VF 7H

8/4/84: 6H,¬Q,VF<1,¬Q,V<2 * 7H,VFF *

*The Blank, when encoded, causes the pointer to advance from the symbol represented by 6H to that represented by 7H. It is as if ,¬Q,VF<1,¬QE..,V<2 in the first example or ,¬Q,VF in the second example had not been there at all.

Stretch Codes

LIT, DICT, ORN

Stretch Codes represent symbols (normally Literals) that are "stretched out" evenly over a distance specifiable in terms of duration, Space Codes or both.

There are two classes of Stretch Codes, the class being chosen on the basis of the direction in which the symbol unfolds.

For symbols that are "stretched" horizontally or obliquely, there is employed a code of the general form

~~or~~ ~~$\sigma_1|\sigma_2$ dzs~~ $\left[\begin{smallmatrix}\sigma\\ \sigma_1|\sigma_2\end{smallmatrix}\right]\neg> \underline{d}\,\underline{z}\,\underline{s}$ (size)

where σ signifies that the symbol is horizontally positioned along the axis specified by σ (which may be defaulted if it is the default value for the Space Code of the symbol); or Pseudo-Space Code or Attach Prefix

upt. 9/14/83

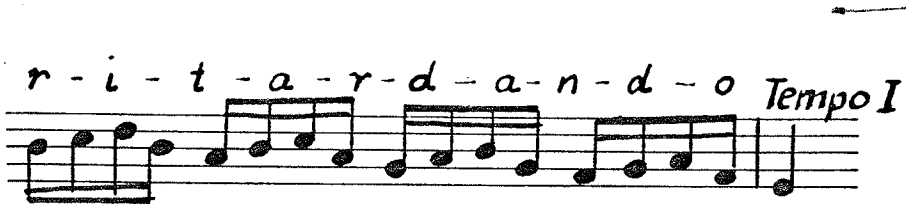
$\sigma_1|\sigma_2$ signifies that the symbol is obliquely positioned, with the Space Code of its left terminus being σ_1 and the Space Code of its right terminus being σ_2;

¬> is the Stretch Operator;

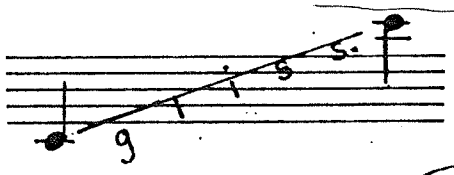
d is the code for the distance stretched (as defined above under "The Skip Code");

z is an optional Size Code, ~~Pseudo-Space Code or Attach Prefix. (See the definition of σ above.)~~

Examples:



((5,00¬>W!@R-I-T-A-R-D-A-N-D-O$ 6 7 5))../1Q,00@¢TEMPO I$...



!&,19QANE?_1 33ASW?_2 & 19|33¬>Q@GLISS.$

For symbols that are "stretched" vertically, there is employed a code of the general form

$$\sigma_1|\sigma_2\neg>\underline{z}\underline{\underline{s}}$$

where $\sigma_1|\sigma_2$ represents the vertical extent of the symbol, σ_1 being its starting point and σ_2 being its terminal point. (Note that unlike most uses of the Double-Space Code, it is possible here for

Can't get by without an example for this!

σ_1 to represent a higher position than σ_2.)

→ is the Stretch Operator (which may be defaulted without introducing ambiguity);

z is an optional Size Code applying to s only;

and s is the code (minus its Space Code) for the symbol being stretched.

Stretched symbols should not be confused with symbols of horizontal extent, the definition and encoding of which are given in section C2. Likewise, they should not be confused with Size-reduced or -enlarged symbols, which are treated in section F3.

Codes Incorporating Both the Push Operator and Instrument Codes

The Push Operator is associated with Instrument Codes in two contexts:

1. The encoding of a change of musical instrument by a player (both instruments--each having its own DARMS Instrument Code--sharing the same stave or system). This is discussed in section A, "Change of Instrument by a Player."

2. The encoding of a symbol that begins in one Instrument and ends in another. This is accomplished by two or more codes for every such symbol, each code having the general form

si→pIn.m$

where s is the code for a symbol of horizontal or vertical extent (e.g., a line, slur, etc.);

i is an identifier, being odd for the left-most (and,

if necessary or desirable, intermediate points of the symbol) and being the next higher even integer for the termination of the symbol;

p may have three different formulations and corresponding interpretations:

> (s) "proceeds to" (In.m)

< (s) "proceeds from" (In.m)

| (s) "proceeds in vertical alignment with" (the next or last encoded point of s as found in In.m) ;

In.m is any Unqualified or Qualified Instrument Code (the form of the former being reduced to In, of course); and

$ terminates the sub-string begun with the ¬.

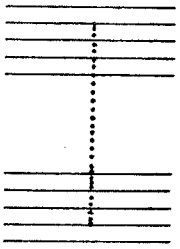
Since si may be a Note attribute, such as a slur, codes of the above form may appear within the code for a Note. } NB

If s has horizontal extent (i.e., is not perpendicular to the stave), the left-most point of the symbol must be considered the beginning and therefore receives the odd-numbered identifier and the > character. It therefore follows that the right-most point must be considered the end and therefore receives the even numbered identifier and the < character.

For symbols of vertical extent that cross Instrument boundaries the choice of vertical bar is mandated, and odd and next-higher even integers again serve to designate beginning and end, respectively; however, it is immaterial how these are assigned. Good encoding style suggests, however, that the first point encoded should receive the odd-numbered

identifier.

Example:



I4 ... 7?:7¬|I5$

I5 ... 3?:8¬|I4$

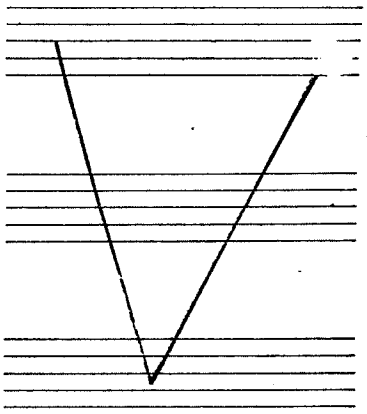
Just as it is not necessary, given a single Instrument, to encode Intermediate Slur Codes (D12) for every note within a slur unless special circumstances require it, it is not necessary, when a symbol crosses several Instruments in the course of its extent, to encode its position within each Instrument between its outer limits, unless circumstances require it.

Examples:

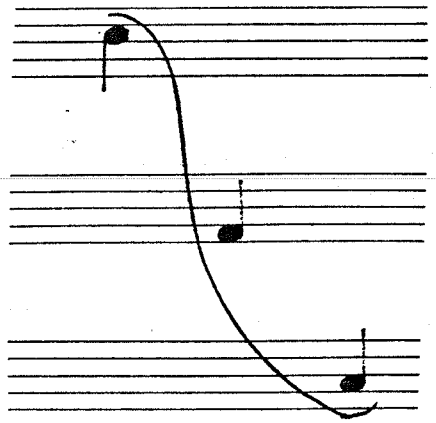
I1.1

I1.2

I2



I1.1	... 5?_3¬>I2$ ...
I2	... 4?_3¬<I1.1$¬>I1.1$ ...
I1.1	... 1?_4¬<I2$ ...
I1.2	No Code Required



9/14/83
1) why parens?
2) why '-' at end, not after <id>?
'-' only at end of slur code or also possible before transition code?

I2 ... 6Q!L5+¬>I3$ ...

I3 ... 2Q(L5¬<I2$¬>I4$-) ...

I4 ... 4QL6¬<I3$ ...

Note that the order (not content!) of the three DARMS substrings under each example may be permuted as long as the appropriate Instrument Code is active when the substring appears in the encoding stream.

Linear Decomposition Mode

By using the facilities described below, it is possible to encode an Instrument in a score in several successive passes. This can result in syntactically simpler coding and a lower percentage of error.

The manner in which an Instrument (or section thereof) may be stratified for encoding in Linear Decomposition Mode is determined by the encoder. Certain obvious possibilities grow out of the very nature of musical textures and the physical layout of manuscripts or printed scores:

1. Piano Music: left and right hand (or bass and treble staves) may be encoded separately.

2. Vocal Music: the voice line and text may be encoded separately. Corresponding lines of strophic texts are encoded as a series of layers with differing Pseudo-Space Codes (cf. section B6).

3. Dynamics and natural-language information (tempo and other instructions to the performer) may be embodied in one or more distinct layers of information. It is recommended that at least one layer of music, that is, Note information, be encoded first.

4. A four-part chorale on two staves (S A and T B) could be encoded in four passes (or, for that matter, in two

In any event, Barlines must be encoded in secondary layers of all instances of Linear Decomposition Mode in all Instruments before any use of an '!In' Instrument Code; this makes it possible for the canonizer program to account for suppressed Instruments in a DARMS-encoded score. In sum, therefore, it is always good (and often necessary) practice to encode all barlines in secondary layers.

or three or conceivably more than four passes).

Linear Decomposition Mode offers the possibility of encoding most beam structures, slurs and other information having horizontal extent in the short form, which might not be possible otherwise.

The metrical layout of a stratified passage is implicitly defined in the first (Note) pass by means of Barline and Duration Codes. However, on subsequent passes the re-encoding of barlines is permitted (and, in fact, recommended) for the sake of convenience and accuracy. The encoder may also use Skip Codes of the form ¬/ (C3) for this purpose.

← See opposite page.

Restrictions

1. Notes on the same stem may not be assigned to different layers; however, by using any method described in section B3, any number of simultaneously sounding Notes of an Instrument may be encoded as belonging to a single layer.
2. Ties (D5), Beams (D9) and Slurs (D12) may not have their openings encoded in one layer and their closings in another.
3. No symbol of horizontal extent (e.g., a glissando, etc.) may have its beginning encoded in one layer and its termination in another. Such symbols are described in sections C2 and E8.

Control Symbols

The following symbols (which must be preceded and followed by a Delimiter) establish the beginning and end of Linear Decomposition Mode and the layers within it.

!& Signals the beginning of every layer except the last and indicates that the information immediately following does not complete the representation of the passage. May also appear within the last layer to invoke a new "partial-information" condition.

& Signals the end of each layer (except the last) and the beginning of the next; thus the position pointer is reset at the position in the score corresponding to the beginning of the previous layer. If not followed by the "partial information" code described immediately above, it is understood that the information that follows completes the DARMS representation of the passage in question.

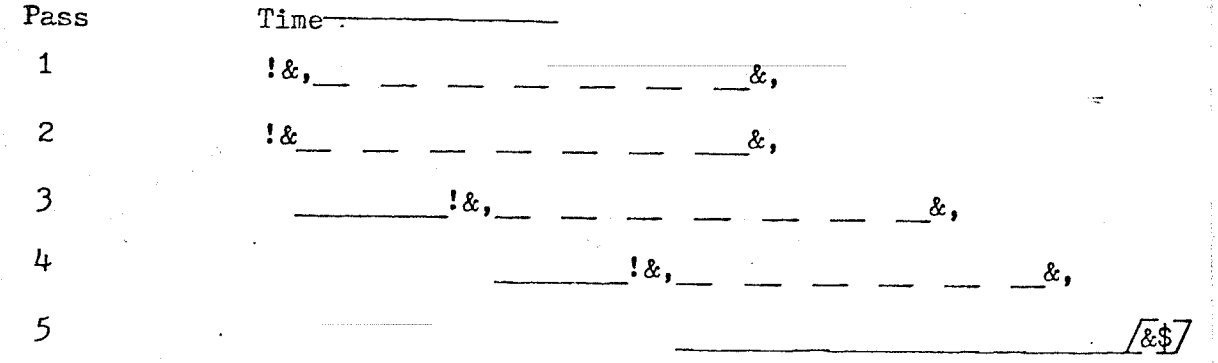
&$ Signals the end of the last layer of Linear Decomposition Mode. The position pointer is automatically set immediately to the right of the right-most position of the stratified passage. Not required unless last stratum does not extend the full length of the longest previously encoded layer. (!)

From the above, it follows that:

1. No layer starts earlier (i.e., in the score, to the left of) the previous layer;

2. A subsequent layer may run beyond the current or a previous layer;

Thus a passage of score being encoded might be logically divided up (and encoded) as follows:



All five passes might just as easily have been terminated at the same time-slice in the score; the example merely indicates that it is not necessary for this condition to occur. (See (2) above).

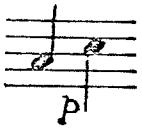
Delimiters and Linear Decomposition Mode Codes

The rules for Delimiters following Linear Decomposition Control Codes are as follows:

1) All invocation points of Linear Decomposition Mode (represented by "!&" without "&," immediately preceding) must be followed by
   a) a Delimiter Comma, and
   b) a code having a Duration attribute (explicit or implicit).

   Thus '7Q !&, 8H' and '7Q &,!&,!F 8H' are illegal.

2) All other occurrences of '!&' and '&' may be followed by a Delimiter Blank or Comma depending on the location of the symbols represented.



!&,4Q 6 & VP

In '!&,4Q 6 & VP' , the blank following the '&' indicates that the first graphic symbol to follow is between the point at which Linear Decomposition mode begins (here the '4Q') and the next left-most symbol in any previously encoded layer. A less flexible but more precise encoding would use the Anchored Skip Code (C3): '!&,4Q 6 &,¬E,VP' . Were the p aligned with the '4Q', the DARMS Code would have been '!&,4Q 6 &,VP' .

3. &$ indicates that the passage encoded in Linear Decomposition Mode has been completely encoded, even if the last layer (i.e., that terminated with &$) does not extend to the right-most position of the passage. What necessarily follows &$ is information for a passage, immediately to the right of the stratified passage, for which nothing has been encoded.

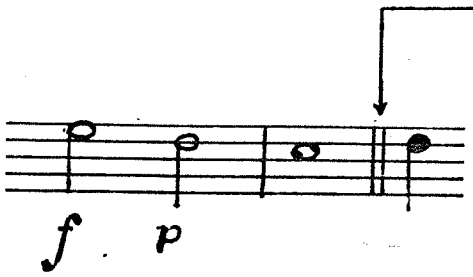
← See opposite page.

Use of the Push Operator

Since not all layers have the same density of information, the Skip Code (C3) is used to "skip over" measures and beats not relevant to the layer being encoded. (It would be wrong to encode such empty spaces as Rests.) Note that within Linear Decomposition Mode the Skip Code is normally preceded by a Delimiter Blank and followed by a Delimiter Comma.

In addition to indicating the position of symbols, the Skip Code can be used to "complete" a layer whose final symbol occurs somewhere to the left of the end of a previously encoded layer. This is done by "skipping" the distance equivalent to the difference in length between the two layers (see example below). The precise representation of this "empty" space facilitates the detection of Duration Code and Barline Code errors, for it is then possible to calculate and compare the lengths of all layers (which will normally, but not necessarily, be equal).

Example:



Position pointer reset here at termination of Linear Decomposition Mode signified by &$. The Delimiter Blank following &$ advances the pointer to the next symbol, in this case the quarter note on the fourth line.

!&,8H 7 / 6W //&,VF ¬H 2 VP ¬/2,&$ 7Q
OR: VP &$ 7Q

Although not illustrated with an example, there is the possibility of encoding multiple "starting points", allowable within the definition of LDM given here. E.g.,

!& —— &
—!& —— &
—— &

Order of Encoding for Notes

There follows an ordered list of attributes of Note events. For syntactically valid encoding, these attributes must be encoded in the sequence given, except when one attribute is "attached" to another (B7). In the case of a note with two stems, the attributes pertaining to one and then, with no Delimiter intervening, the attributes pertaining to the other, are encoded.

1 Open Beam, short form only

2 Space Code (σ)

1. Single (B1), or

2. Vertical Structure

a. Chord Structure I or II (B3) with or without Chord Structure Definer (B4)
b. Chord Structure Reference (B5)
(N.B. Chord Structures may alter the normal order of encoding because they incorporate some of the categories of information below. See section B3.)

3 Accidental (α)

4 Notehead

5 Duration (δ)

6 Tie

7 Stem

8 Tremolo

9 Beam Code (any Long Form Code)

10 Articulation

11 Appropriate Dictionary Codes

12 Ornament

13 Slur

14 Fingering

ordered according to relative proximity to the notehead, closest one first

15 Figured Bass

16 Close Beam (short form only)

N.B. For double-stemmed notes, the recycling must begin with Duration Code or Stem Code as indicated by the arrows. See section D14.

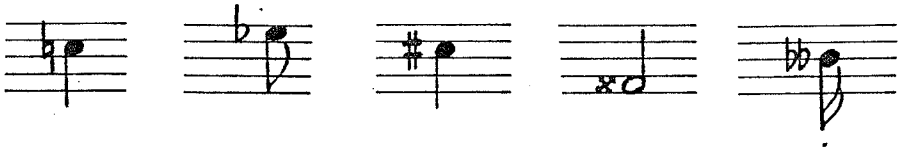
size reduction should be used for small accids - reduc. in size is not an editorial convention, as it is with Clefs.

Accidental Codes[1]

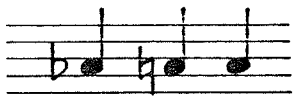
Accidentals are represented as follows:

Symbol	Code (α)
Sharp (♯)	#
Double Sharp (𝄪)	##
Flat (♭)	-
Double Flat (𝄫)	--
Natural (♮)	*

Examples:



6*Q 8-E 6#Q 2##H 5--S



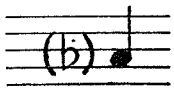
4-Q *Q Q

Except in certain special cases--such as ornaments (D11) or a literal involving an accidental, for example, "B♭ Clarinet" (E6)--an accidental is associated with some graphically subsequent note. The position of an Accidental Code is between the associated Space Code σ and ~~Duration Code δ (σαδ)~~ Notehead Code (σαν).

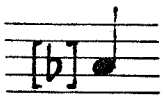
[1] Metalinguistic symbol for accidentals is α (lower case alpha).

If an accidental is enclosed in parentheses or square brackets, it is encoded as (α) or ⟨α⟩, respectively.

Examples:



4(-)Q



4⟨-⟩Q

Editorial accidentals, if placed above the note, are treated as Ornaments (D11).

If in normal position but size reduced or parenthesized, editorial accidentals are encoded as Accidentals, not Ornaments.

Example:

6!'(-)$QL> ; Canonizer must be able to recognize this token.

(Note the use of a Size Reduction Code (F3) that applies, in this case, to the Accidental information only.)

Special Notehead Codes

The code for any of the special noteheads listed below has the general form

Nn

where n is an identifying digit. The table of Notehead Code Identifiers follows.

n	W	H	Q	E	Explanation
0	-	-			Notehead missing
1					Stem missing; no duration; e.g., gives pitch of harmonic
2					Double notehead
3					Triangular notehead
4					Square notehead
5					"x" superimposed on the notehead
6					"x" instead of the notehead; Sprechstimme, percussion
7					Diamond-shaped notehead; centered stem (if any); percussion and 16th century mensural notation.
8					Stem (if any) on side, e.g., for harmonics
R					Rest symbol stands in place of notehead; treated as Note, not rest for purposes of Sigma and

δ: determined by # beams; placed in Note δ-buffer

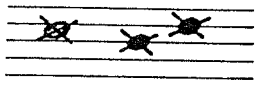
σ: " " default table for rests; " " "σ-"

Delta Propogation (B1 and D4 respectively). However, the value of σ is determined by the rules for rests given in D4.

The Notehead Code (if any) follows the Accidental Code (if any) and precedes the Duration Code.

Combinations of noteheads on the same Space Code are encoded as a concatenation of Notehead Codes.

Example:

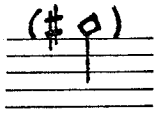


6N1N5H 5N1N5Q 7N1N5

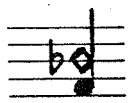
(N) without a digit is a parenthesized notehead. If the notehead itself requires a Special Notehead Code, then that Code--not just its identifier--is suffixed to (N).

Examples:

6(N)HD



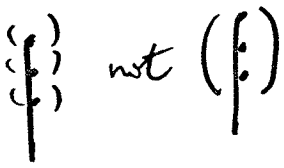
30(#N)N7HD
(Note that (#N7) is illegal; parenthesized accidentals are discussed in section D2.)



2QU1,5-N7HU1

OR

|2|5-N7H|Q

*) 1+2+2Q(N) ⟹  not

Duration Codes[1]
(Notes and Rests)

Duration Codes for Notes occur after the Notehead Code (if any) and before the Tie Code (if any).

An unabbreviated Duration Code consists of 1) a letter specifying the basic durational denomination of a note or rest, 2) a string of periods equal in number (possibly zero) to the number of dots that qualify the value of the letter code, and (where required) 3) a Groupette Identifier.

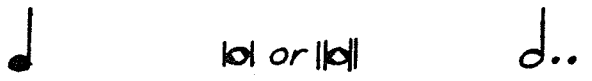
The following table summarizes essential Duration Code information, the underscores indicating mnemonic relationships.

W	Whole
H	Half
Q	Quarter
E	Eighth
S	Sixteenth
T	Thirty-second
X	Sixty-fourth
Y	One hundred twenty-eighth
Z	Two hundred fifty-sixth
.	(dot)
G	Single-grace note (small eighth note, slashed: )

[1] The metalinguistic symbol for Duration Code is δ (lower case delta).

To extend the scale in either direction, the extreme codes are iterated for each additional multiplication (in case of W) or division (in case of Z) by 2. Thus, WW stands for a double whole note or breve, WWW for a quadruple whole note, etc.

Examples (all information except Duration Code is suppressed):



Q WW H..

Non-normative Duration Codes

Durations of Notes (or Rests) that are members of non-normative groupings such as  and  are specified by use of a Groupette Identifier (C5) which is appended to the normative denomination, for example,

Q3 H.6 T..5

Note that the Groupette Identifier follows any dots that may be part of the Duration Code.

Rests

Rests are encoded in the form

$$\sigma R \delta \underline{x}$$

where σ is the Space Code (as determined by the criteria below);

R is the mnemonic for "rest" and

δ is the Duration Code of the rest; and

$\underline{x}$ is a code for a Fermata or some other symbol that can be appropriately associated with a rest.

Parenthesized or bracketed rests are encodable in the form '$(\sigma R \delta \underline{x})$' or '$<\sigma R \delta \underline{x}>$', respectively, where σ, R, δ and $\underline{x}$ are as defined above. If only $\underline{x}$ is parenthesized (bracketed), then the left parenthesis (bracket) is placed immediately in front of $\underline{x}$, e.g., '$\sigma R \delta <\underline{x}>$'.

Space Codes for Rests may be assigned by default, as given in the table below:

Type	Factor Determining Space Code	Default Value
	Space in which the Rest is located	26
	Point of bottom hook	24
	Nub of topmost flag	26 (28 for 32nd and shorter rests)

Note that Rests always have even-numbered Space Codes. → make this an attribute?

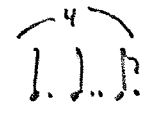
If a Rest is attached to the stem of a beam system, it is treated as a Note having a special notehead (D3).

Delta-Suppression for Notes

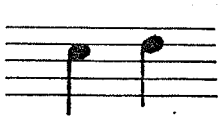
1. If a Note IS NOT part of a beam system, δ may be suppressed

a. if the current δ is exactly the same as that of the most recently encoded Note, OR

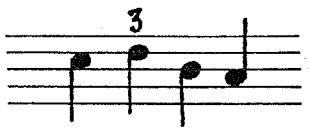
b. the current δ differs from that of the most recently encoded Note only in having more dots than the antecedent, in which case only the additional dots need be encoded for the current Note. The Duration Code for the current note will then be the full value of the previous δ plus any additional dots encoded with the current note.

⇒ 
'Q.1. .E.1'

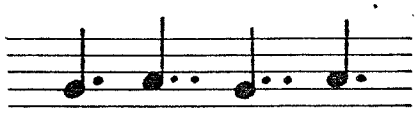
Examples:



6Q 7



!3Q2:H 6Q2 7 6 4Q
(Groupette Definer: section F1)



3Q. 4. 3 4Q.

2. If a Note IS part of a beam system, the letter code of the δ in question is inferrable from the number of active beams and therefore may always be suppressed EXCEPT

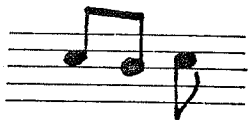
a. when there is a change or cancellation of Groupette Identifier;

b. when the denomination (e.g., Q, E1, etc.) remains the same as that of the most recently encoded Note of the same beam system but is qualified by fewer dots than the antecedent Note.

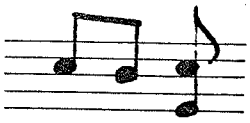
In cases 2a. and 2b., δ must be given in full. In other cases, dots, not being inferrable from Beam Codes (D8), are suppressible according to the rules outlined below.

Note. The full value of δ (explicit or implicitly stated) for the last note or chord propagates to the next note or chord if the note or chord is not part of a new beam structure and δ is omitted.

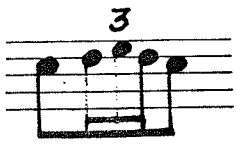
Examples:



(6U 5) 6
6U(B7 5B8) 6



(6U 5) 1+5
6U(B5 5B6) 1+5



!3S4:E ... (6 (7S4 8 7) 6E)

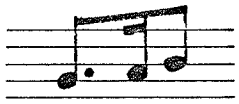
!3S4:E ... 6(B1 7S4(B1 8B1 7B1) 6EB2)

(2. (4 5))

2.(B1 4(B1 5B2))

(5. 6E (7))

5.(B1 6EB1 (B1 7B2))



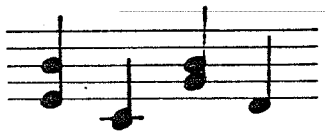
(3. (4) 5)

3.(B1 (B1 4B1) 5B2)

If any of the notes in the examples above were replaced by chords, a Chord Structure representation (B3) could stand in the place of the Space Code without affecting the rules for Delta Suppression for Notes. In DARMS syntax both single notes and chords (several notes on the same stem) are classified as Notes.

(A)

Example:



1+4Q 19 3+2 0

Delta-Suppression for Rests

If δ for a rest is suppressed, it is assumed to be the same as that for the most recently encoded rest. } attribute; provided that there has been already an explicit rest δ given

Dot-Suppression for Notes and Rests

If two successively encoded notes (or rests) have identical Duration Codes (explicit or implicit) except for the number of dots, the following rules apply:

1. If the second δ has more dots than the first, then that part of the second δ identical with the first may be suppressed.

Examples:

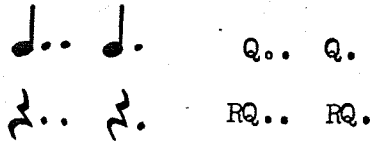
Q. Q.. may be abbreviated as Q. .

RQ. RQ.. may be abbreviated as RQ. R.

Q. RQ. Q.. RQ.. may be abbreviated as Q. RQ. . R.

Note that Notes (single notes and chord structures) and Rests are treated independently of each other in the application of these principles.

2. If the second δ has fewer dots than the antecedent, then no suppression at all is permitted. Thus, in



and

no abbreviation is possible.

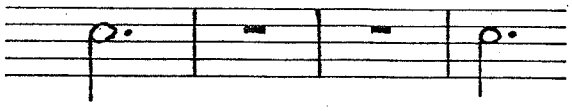
Multiple Measures of Rest

Multiple measures of whole rests may be abbreviated in the form

RnW

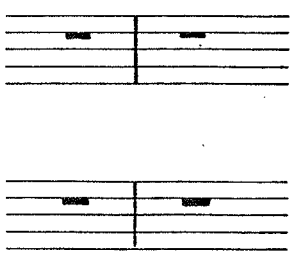
where n is an integer signifying the number of measures. (Note that, conventionally, a whole rest may stand for a complete measure of rest in meters other than 4.) When this method of encoding multiple measures of whole rests is employed all intervening barlines are implicitly included. However, the left-hand barline of the first measure and the right-hand barline of the last measure are not included.

Examples:



6H. / R2W / 6

OR



R2W,76R2W

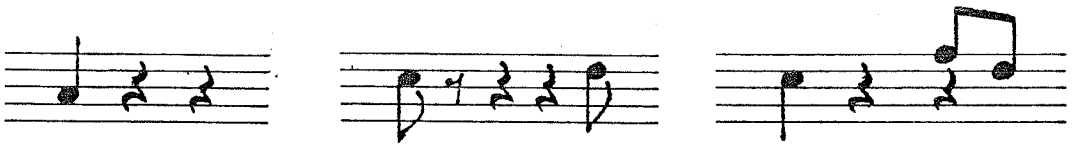
not possible!
This involves backtracking!
'26RW,76RW / 26RW,76RW'

Restriction: when a multiple-measure rest code is employed, there must be, in the current Instrument, no other symbol in the time-slice the code represents with a different duration; this restriction may be circumvented if Linear Decomposition Mode is employed, with the multiple-measure rest encoded in one layer and the other symbols in other layers.

Note that whole rests are encoded at the point at which they take effect, not at that corresponding to their physical placement in the measure.

Successive rests within the same measure may also be encoded in abbreviated form. By this special abbreviation format, the horizontal-position pointer is automatically advanced to where the last rest is located.

Examples:



4Q RQQ 6E REQQ 7 6H RQQ,(9U 7)

Restrictions:

1. If no Space Code prefixes R, then default Space Codes for the Rests will be assumed. If, on the other hand, a Space Code is encoded then all the Rests in that token will be assumed to have the same Space Code even if their defaults are different.

2. This type of abbreviation is not valid for multiples of W and Z. For example RWW stands for a breve, not two whole rests.

Ties

The tie as a graphic category is to be considered a species of slur, since ties and slurs are graphically so similar that often only a consideration of context will enable one to be distinguished from the other. Because of this DARMS permits the encoding of any tie as a slur (D12), although not vice versa. The distinction between slur and tie, since it is not crucial either to music-printing or to musical graphics, may thus be reserved for the music-analytic stage of a project employing the DARMS system. However, because most DARMS users will, in fact, recognize ties as such and may not wish to encode them as slurs--only to have to design subsequently a "tie recognition routine"--a separate Tie Code has been invented.

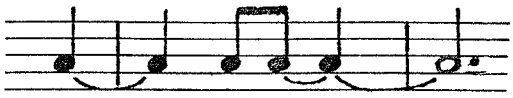
The symbol for a tie is J (mnemonic "join"). All Tie Codes follow the Duration Code (if any) and precede the Stem Code (if any) in the order of encoding.

There are two classes of Tie Codes.

A. Simple Tie Codes are employable only in the case where a tie terminates at the next note in the same Instrument having the same Space Code explicitly encoded or implied by Sigma Suppression. A Simple Tie Code has no Identifier and is stated only on the first of a pair of tied notes. In

canonization a Simple Tie Code will be converted into a Pair of Tie Codes with Identifiers (described below) incorporated into the code for both notes connected by the tie.

Example:

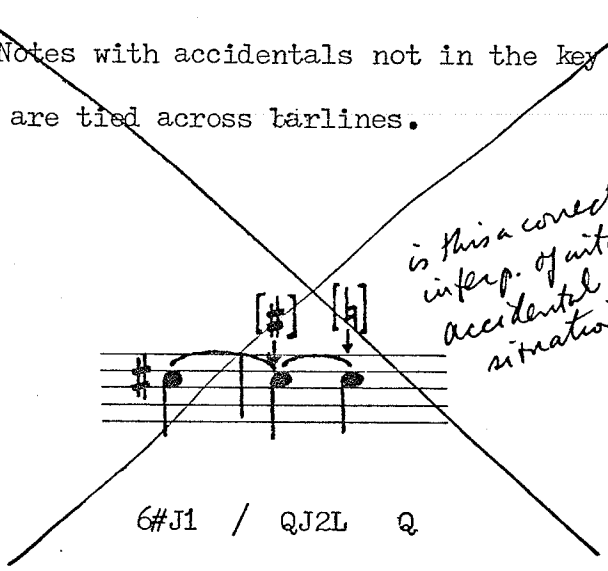


4QJ /1Q1(E EJ) QJ / H.

B. Tie Codes with Identifiers are employed for all other cases of ties. Among the conditions mandating the use of Tie Codes with Identifiers are

1. Notes with accidentals not in the key signature that are tied across barlines.

Example:



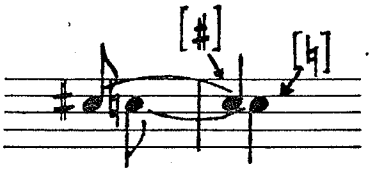
is this a correct interp. of intended accidental situation?!

2/21/83 no problem bec. space codes still match

6#J1 / QJ2L Q

1. Multiple notes with the same Space Code (regardless of accidentals) tied across barlines

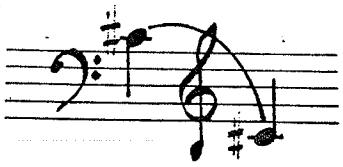
Example:



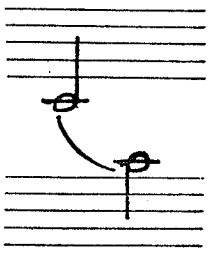
6#EJ1U,,*EJ3 / QJ2U,,QJ4

2. Ties going across a change of stave or clef.

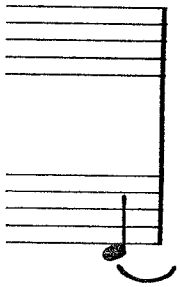
Example:

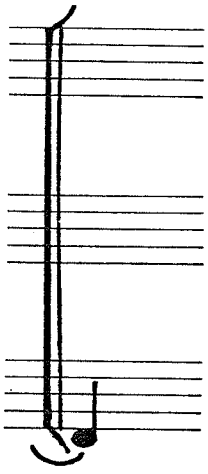


!F 31#QJ5 !G 19#J6



19HJ3 ... 81HJ4

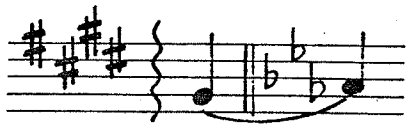




70J1 ... 120J2

3. Ties connecting enharmonically equivalent notes.

Example:



!K4# ... 3QJ1 // !K3- 4J2

Special Ties

1. Dotted and dashed ties : suffixes

a. Ties that are dotted lines are represented by suffixing a colon ("dots") to the Tie Code for the first note of the tie.

b. Ties that are dashed lines are represented by suffixing an equal sign ("dashes") to the Tie Code for the first note of the tie.

Examples:

J: J= J3: J7=

Note that these suffixes follow the identifier, if any.

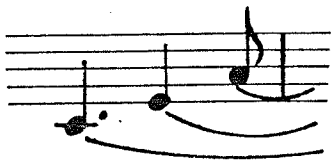
2. Parenthesized and bracketed ties

Ties in parentheses or square brackets are encoded by enclosing any of the Tie Codes described thus far in parentheses or < >, respectively.

3. Ties without terminating notes

Ties without terminating notes are encoded as notes of a definite pitch tied to null symbols (E8) of the same pitch.

Example:



19Q.J 1QJ 4EJ / 19?0,1?0,4?0

OR

19Q.J 1QJ 4EJ / 19+2+3?0

4. Ties with more than two end points

Ties with multiple beginnings and one termination (e.g., ties beginning in each of two or more Qualified Instruments that terminate together in the Unqualified Instrument) or ties with one beginning but several terminations (as in the first, second, ... endings of a passage of music) are handled according to the principles outlined in section C2.

5. Fingering "Ties"

Changes of fingering on a note are sometimes indicated by means of a tie-like line connecting the fingering digits, for example $\overset{\frown}{45}$. Such markings are coded as Ties within the Fingering Code. See section D10.

Stem Codes

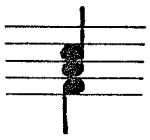
The Stem Code has two functions:

1. To indicate the direction of a stem by the mnemonics

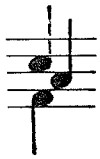
U ("up") and
D ("down"), and

2. To identify those notes in a multi-stemmed structure that belong to the same stem. This is done by suffixing the same integer identifier to the stem-direction letter codes of notes on the same stem. If the same identifier is used with different letter codes, for example D1 and U1, then the stems are assumed to be distinct. Stem Identifiers hold only for a single vertical axis, becoming available for reuse immediately thereafter. In short, a Delimiter Blank (or Double Comma) frees all Stem Identifiers for reuse in that Instrument.

Examples:



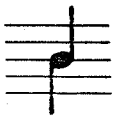
2QD,4U1,6U1



2QD,6U,,4

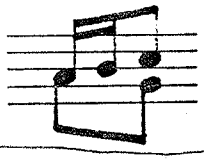
3. Unless a Stem Code <u>with identifier</u> specifies the contrary, a stem is assumed to be associated with only one notehead. However, a notehead may have more than one stem.

Examples:



5QDU
or
5QU,QD
or
5QU,5D
or
5QD,5U
etc.

5QDEU
or
5EUQD



4U((B1D(B3 5UB1) 3DB4),6UB2)

Double-stemmed Notes are discussed further in D14.

4. To encode a stem of the form  or  , a percent sign is suffixed to the Stem Identifier (which may be null). The stem direction letter code (U or D) may not be defaulted in this case.

Examples:

5QD%

4EU%

2+2+2QU%
or
2QU1%,4U1%,6U1%

Stem-Code Suppression

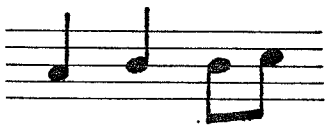
Default values for Stem Codes will automatically be supplied in canonization for all notes not encoded with Stem Codes. This means that failure to encode a Stem Code explicitly will result in the default Stem Code value for that Note being assumed. Stem Code suppression is to be employed only for Notes 1) whose complete Stem Code is either U or D and 2) meeting the criteria for Stem Code suppression defined below. In other words, Stem Codes with Identifiers or other suffix may never be suppressed. For all other cases, however, the simple Stem Code may be suppressed (and, therefore, the appropriate default value will be assumed by the Canonizer) for those single notes and chords meeting the criteria defined below.

1. Unbeamed notes or the first note of a beam system:

   The Stem Code may be suppressed for

   a. notes on or above the middle line having a downward stem;

   b. notes below the middle line having an upward stem.



4Q 5U (5 6)

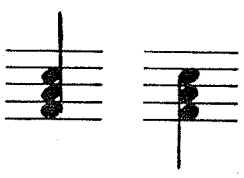
2. Unbeamed chords or the first chord of a beam system:

   The Stem Code may be suppressed if

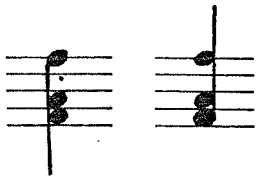
a. the note farthest from the middle line of the stave is in the direction opposite from that of the stem;

b. the outer notes of the chord are equidistant from the middle line and the stem direction is downward.

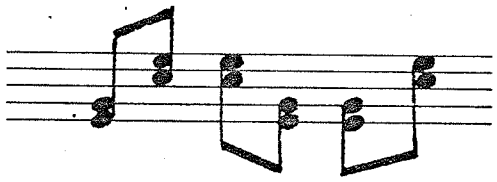
Examples:



2+2+2Q 2+2+2QD



2+2+5Q 2+2+5QU



(1+2 6+2) (6+2 1+2) (1+2D 6+2)

3. <u>All other notes or chords of a beam system</u>:

The Stem Code may be suppressed for all remaining notes and chords of the beam system <u>except</u> where

a. such special stem information (identifier, etc.) is required on a particular note or chord;

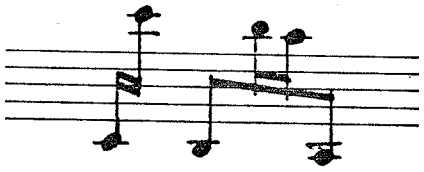
b. there is a change of stem direction, in which case the new direction is encoded with the note at which the change occurs; this new direction

will be assumed for the rest of the beam system unless again countermanded.

Examples:

6QD1,8D1

6QD%



((19 33D)) (18 (32D 31) 17U)

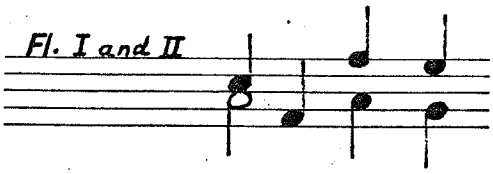
See also the beamed examples in the preceding paragraph.

2 qual. rules for stems

The above rules apply except when the active Instrument Code is of the form In:1 or In:2 and there is no In:m where m > 2. In In:1, Stem Code "U" will always be assumed unless explicitly countermanded in each contrary instance; in In:2 the Stem Code "D" is assumed. See section A.

how do we know there's no In:3 ??

Example:



I2:1,6Q 2 9 8 ... I2:2,4H Q 3 ...

Tremolos

Tremolos are of two types, one-Note and two-Note ("Note" being taken here in the DARMS sense of a single note or chord), and their DARMS encodings have certain features in common with the representation of normal beam systems. The reader is therefore advised to read section D8 before continuing.

A. One-Note Tremolo Codes consist of as many nested pairs of parentheses as there are slashes in the tremolo notation. In their full or "canonical" form they are part of the code for every note to which they pertain, appearing between the Stem Code and Long Form Beam Code (as shown in the example below). However, the encoder is obligated to encode the tremolo only once for a given chord: either in association with a Chord-Structure format (B3-B5) or as part of the code for the first-encoded note of a chord, when the code for each note is separated by a Delimiter Comma. Note, however, that this implies "tremolo propagation" by virtue of "tremolo suppression" for each subsequently encoded note on the slashed stem or, in the case of whole notes, for all notes in that vertical slice. Therefore, it is required that all notes not associated with the tremolo be encoded before the note or notes that are. Tremolo propagation is assumed to exist from

the point a tremolo is detected in the input stream until the position pointer is advanced to a new vertical slice by the recognition of a Delimiter Blank, or Skip Code. or Double Comma

Examples:

2Q(())

21HU1(),24HU1(),26HU1() (Canonical form)
1HU1(),4U1,6U1
|1|4|6|H()
1+3+2H()
(Three valid forms using the formats described in section B3)

Correct: 7QU,2W((())),4

or

7QU,2+2W(((

or

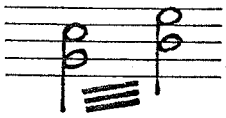
7QU,4W((())),2

Incorrect: 2W,4((())),7QU

or

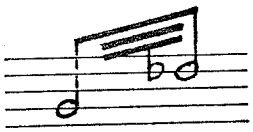
2W((())),7QU,4W

B. Two-Note Tremolos consist of two notes or chords separated horizontally by a tremolo-beam system. The DARMS system provides a Long and a Short Form of two-Note tremolo encoding; however, the Short Form is not usable when ambiguity might arise, most notably when there is another two-Note tremolo or beam system active in the same Instrument. Since two-Note tremolo indications are graphically similar to beam systems, the method of encoding them is somewhat similar, too, as can be seen in the following examples:



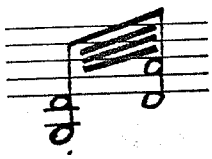
Long Form: 7WB1 (((B1))) 5WB2 3+3HDB1 (((B1))) 5+3HDB2

Short Form: 7WB ((())) 5W 3+3HDB ((())) 5+3HD



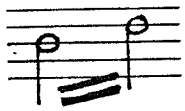
Long: 3H(B1 ((R1)) 7-HUB2)

Short: (3HB ((())) 7-HU)



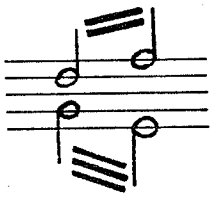
Long: 16HU1(B3,20HU1B3 (((B3))) HU2B4,4U2B4)

Short: (16+4HB ((())) 0+4H)



Long: 5HB1 ((B1)) 7HB2

Short: 5HB (()) 7H

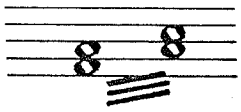


Long: 3HDB1,7UB3 (((B1))),((B3)) 1DB2,9UB4

Short: Not applicable

Long: 4+3HB1 ((B1)) 6+3HB2

Short: 4+3HB (()) 6+3H



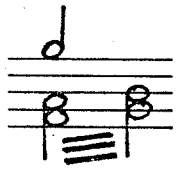
Long: 2WB3,4B3 (((B3))) 4WB4,6B4

Short: 2WB,4 ((())) 4W,6

or

Long: 2+2WB3 (((B3))) 4+2B4

Short: 2+2WB ((())) 4+2



Long: 30HU,2HD1B3,4HD1B3 (((B3))) 3HD1B4,5D1B4
Short: 30HU,2HD1B,4D1 ((())) 3HD1,5D1

From the above examples of two-Note tremolos it will be observed that

1. Each balanced pair of parentheses corresponds to one slash in the tremolo being encoded.

2. Since these slashes (although not physically connected with the two stems) are invariably understood to be associated with a note or chord on either side of the slashes, the terminal points must be designated as such:

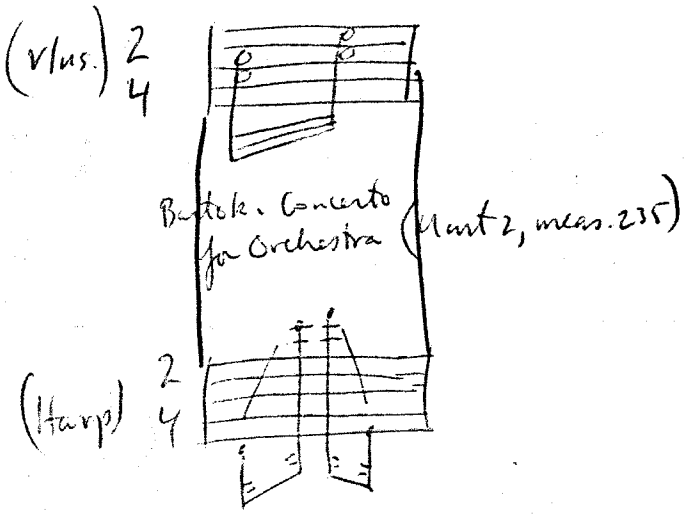
a. in the Long Form by Beam Code Identifiers in the codes for all Notes associated with the tremolo at both terminal points, and

b. in the Short Form by the letter B in the normal position for Long Form Beam Codes within the code for the first Note of the tremolo (see 4. below).

3. If the terminal points are also physically connected as part of a beam system and the Long Form of Tremolo is employed for the encoding, then it uses the same Beam Identifier as the beam itself.

4. When the Short Form is used with Chords, the Tremolo Code letter B must be encoded in the first-encoded note of each

chord and all succeeding notes in that vertical slice must be part of the tremolo chord, whether or not the B is reiterated for each, that is, whether or not Tremolo Code suppression is employed. See the discussion of one-Note Tremolos above.



← encode as a tremolo

← 11/8/82 seems that we need an "Effective Duration" code for Beam systems as a whole.

Beam Codes

All Notes connected by at least one beam are considered to belong to the same beam system. There are two methods of beam encoding, a "long" and a "short" form. The Long Form may be employed for any beam structure. The Short Form can be used only when all Notes and Rests encoded between the opening and closing of a beam system belong to that beam system.

Long Form Beam Codes

1. Every beam system is assigned a Beam System Identifier of the form

Bi

where i is an unsigned integer identifying a particular beam system.

Like other symbols that begin and end in different time-slices, Beams are designated by Identifier pairs incorporated into the Beam Codes for each Note on a Beam System. (These Codes may be suppressed under certain circumstances. See "Beam Code Propagation and Suppression in Chords" below.) All Notes on a given Beam system, except those on the stem that terminates the system, will have the same odd-numbered Beam Code Identifier; those on the final stem will have the

next higher even integer as Beam Code Identifier. When the horizontal-position pointer is advanced beyond this stem, the Beam System is assumed to be terminated, thus freeing the identifier pair for reuse in another Beam Code. (It is recommended that such reuse be delayed to facilitate the detection and localization of encoding and input errors in the data.)

2. All Long Form Beam Codes are positioned immediately after the Tremolo Code, if any, and in front of the Articulation-Fingering-Ornament-Slur complex, if any.

3. The opening of a new beam system or of new beams internal to an already opened system is represented by the Beam System Identifier prefixed with a string of left parentheses of a number equal to the number of new beams, for example, ((B3 .

4. The Beam Code for a beamed Note on which no beams are opened or closed consists of the appropriate Beam System Identifier only (the number of beams being easily inferred), for example, B3 .

5. The closing of a beam system or any beams internal to it is represented by the Beam System Identifier, to which is suffixed a string of right parentheses equal in number to the number of beams closing at that point, for example

B3)) or B4)) ,

depending on context.

Example:

2(B3 2((B3 3B3 4B3 5B3)) 6B4)

Short Form Beam Codes

1. The meaning of parentheses corresponds to that in Long Form Beam Codes.

2. Beam System Identifiers are omitted because they are superfluous: all Notes (and Rests) encoded between the opening of a beam system encoded in Short Form and its termination are assumed to be part of it.

3. Non-Note and non-Rest information (e.g., barlines) may be encoded between Notes of a beam system.

Example:

(6 8 / 7)

The Duration Code is here suppressed according to the Delta-Suppression option discussed below.

4. Left parentheses (beam openings) are inserted immediately in front of the Space Code (or Vertical Structure equivalent). Right parentheses (beam closings) are placed

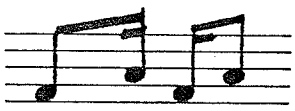
after all other information pertaining to that particular Note, that is, immediately in front of the Delimiter, or, in the case of recycling, before the attributes of the second note are encoded. See sections D1, D6 and D14.

5. Short Form Beam Codes are illegal when a Long Form Beam System is "active" (i.e., not terminated) in the same Linear Decomposition layer of the same Instrument. Short Form Beams may not be opened for the second-encoded stem of a double-stemmed note.

Special Features

1. "Broken Beam". If the beginning or the end of a beam is not attached to a stem, a blank is inserted after the Open Beam Code or before the Close Beam Code.

Example:



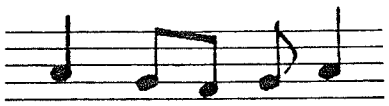
Long Form: 2(B1 (B1 4B2)) 2((B3 B3) 4B4)
Short Form: (2 (4)) ((2) 4)

Long: 4(B1 5B1 B2)
Short: (4 5)

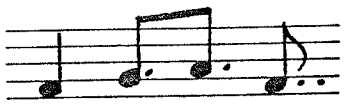
2. Delta Suppression. Because the letter component of Duration Codes can be calculated from the number of "active beams" (except in the case of the tremolo, which is treated separately in section D7), the letter component of the Duration Code of any note on a beam system may be suppressed if the Space Code is not suppressed.

N.B. Regarding the rules for suppressing the letter part of the Duration Code for a dotted note and for propagating dotted values forward, see section D4. The Duration Code (explicit or implied) of a beamed note may be propagated forward to the next Note when the latter is unbeamed.

Examples:



4Q (3 2) 3 4Q



2Q (3. 4) 2.

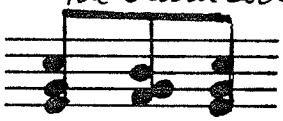
3. Beam Code Propagation and Suppression in Chords. If a chord is beamed, it is not always necessary to encode a Beam Code for each note of the chord.

a. If the chord is encoded having a Chord Structure format (B3), a single Beam Code will suffice for all notes.

b. If a beamed chord is encoded such that each note is a separate token terminated by a Delimiter Comma and incorporates the same Stem Code Identifier,

then it is necessary to encode Beam information for only the first encoded note of the chord. The Beam Code will then be assumed by default for the subsequently encoded notes on that stem. (1/4/83) N.B. Corollary: short form beam codes do not necessary enclose all notes on a beam system; notes on same stem following short beam close code inherit the beam code.

Example:



Long: 1+2+3(B7 2+1+2B7 1+2+3B8)
Short: (1+2+3 2+1+2 1+2+3)

or

Long: 1U3(B7,3U3,6U3 2U4B7,3U4,5U4 1U5B8),3U5,6U5
Short: (1U3,3U3,6U3 2U4,3U4,5U4 1U5),3U5,6U5

4. Beams Spread over Two Systems. Beams spread out over two systems by being broken at the end of the line are encoded as if no break had occurred.

is encoded as

Long: 30((B1 / 8B1 7B1 6B2))
Short: ((30 / 8 7 6))

5. Rests Incorporated into Beams. There are two ways in which a rest may be incorporated into a beam system:

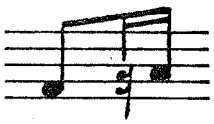
a. as a free-floating rest, in which case it is encoded as a normal Rest;

Example:

Long: 2(B5 RS (B5 4B6))
Short: (2 8RS (4))

b. as a rest associated with a stem (which need not actually touch the rest), in which case the rest is encoded as a special notehead (D3).

Example:



Long: 2(B5 4NRS(B5 4B6))
Short: (2 (4NRS 4))

Articulation Codes

Many of the most common articulation marks may be represented by the one- and two-character codes given in the table below. Other articulation marks (bowings, harmonics, etc.) are to be found in the Dictionary of Additional Symbols (E8).

Articulation marks, (including Dictionary Codes

?/ ?- ?D ?O ?Q ?V ?+ ?; ?,),

Fingerings (D10), Ornaments (D11) and Slur Codes (D12) form a complex that is normally encoded immediately after the Long Form Beam Code (if any). The internal ordering within them, in any given direction, is determined by proximity to the notehead, the closest element being encoded first unless specified to the contrary by means of one or more Attach Prefixes (B7), the complex will be assumed to be closer to the notehead than to the end of the stem (or, in case of a whole note, in the direction opposite the default stem value, had there been a stem).

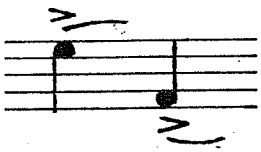
If the Articulation-Fingering-Ornament-Slur complex is distributed into two or more directions, the use of Attach Prefixes (B7) is required for all symbols but those in the default position defined in the previous paragraph. Note, however, that the Attach Code must be appended to the appropriate attribute, most likely the stem, and thus the attached

symbols may be encoded before the Long Form Beam Code and the Articulation and other Codes that are in default position.

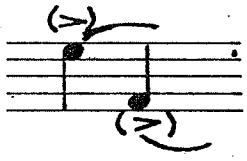
Table of
Common Articulation Codes

Symbol	Code	
•	'	Staccato
▼ or ▲	"	Wedge accent
–	_	Tenuto
>	>	
V or Λ	<	
𝄐	;	
𝄓	;2	
𝄔	;3	
𝄕	;4	
𝄖	;6	
𝄗	;8	

Examples:



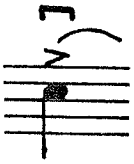
8QL1> 2>L3



8QL1(>) 2(>)L3

D9.3

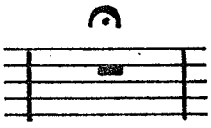
2QUA?D>L



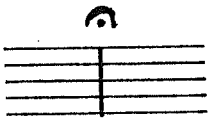
6QD>L?D

6Q'_

8HL3>;



/ RW; /



/;

Fingering Codes

Codes for fingerings, which are introduced by the mnemonic F , are given (when they are required) as part of the Articulation-Fingering-Ornamentation-Slur complex of codes.

Simple Fingering Codes have the form

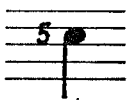
Fcf

where F stands for "Fingering"

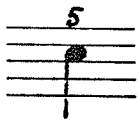
c is a one-character compass direction (default = N) indicating the direction of the fingering from the Note, and

f is a string consisting of one or more of the digits 1 through 5, ordered on the basis of vertical position, the bottom digit being encoded first.

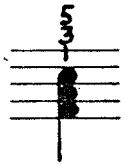
Examples:



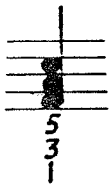
6QFW5



6QF5



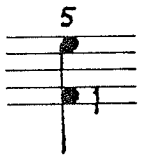
2+2+2QDF135



2+2+2QUFS135

If fingerings are placed in more than one direction from the corresponding notes or chords, a separate Fingering Code must be given for each value of c. The resulting series of Fingering Codes is then concatenated to form a complex of Fingering Codes.

Example:

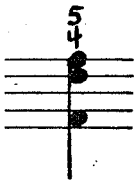


2+6QFE1F5

In order to represent accurately the wide variety of fingering forms found in musical scores, additional information may be required in the Fingering Code. A list of supplementary Fingering Code characters and their meanings therefore follows.

%	Signifies the absence of a fingering digit for a note of a chord. However, % need not be given after the last explicit digit is encoded, being assumed by default. '%' does not exist, i.e., is not a Fingering Code char in Canonical DARMS, if Fingering Codes get distributed to indiv. Notes.

Examples:



2+5+2QF%45

or

|2|7F4|9F5|Q

or

2QD1,7D1F4,9D1F5

2+2+2QFW%3

or

|2|4FW3|6|Q

or

2QU5,4U5FW3,6U5

(NOTE: The above examples illustrate the implications of the various types of Space-Code encoding for chords for the deployment of Fingering Codes. Given that Full Encoding necessitates and Space-Pattern Representation permits the assignment of fingering information to the individual notes of a chord--thus resulting in several Fingering Codes, each simpler than the composite Fingering Code mandated by the Base-Increment Representation--only the Base-Increment Representation will be used in subsequent examples. For further information on the encoding of chords, see section B3.)

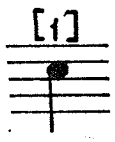
( ) and < >

Left and right parentheses represent left and right parentheses respectively; < and > represent left and right square brackets, respectively. They are encoded in the order corresponding to their position with respect to a fingering digit.

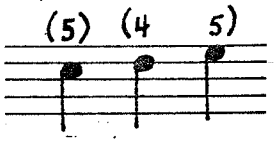
N.B. Because ambiguity may arise if a Fingering Code ends in) or > (which may be taken for a

Short Form Beam Code or Articulation Code, respectively), it is mandated that a Fingering Code ending in either of these characters be terminated with a dollar sign.

Examples:

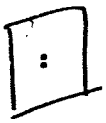


6QF<1>$



6QF(5)$ 7F(4 8F5)$

2+4+2QF1(4)(5)$



Used to delimit the codes for a column (or part of a column) of two or more fingering digits that is bracketed by a parenthesis or has a tie beginning on every digit. (Ties, which are encoded with the character J, are discussed below.)

The rules for the use of the colon (which always implies parenthesized or tied fingerings when associated with a Fingering Code) are as follows, given the general single-column form for Fingering Codes

$$F\left(\left\langle :\underline{f}'_1\underline{f}'_2 \cdots \underline{f}'_n :J \right\rangle\right)\$$$

→ if at end of Fingering Code

where $\underline{f}'_i$ consists of 1) a fingering digit and 2) where required, a meaningful combi-

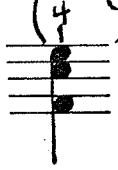
nation of

( ) < > J

that applies to <u>that digit only</u>:

1. The left colon is required if two or more $\underline{f'_i}$ in immediate succession are bracketed by the same left parenthesis. or if ≥2 f_i' in immed. succession are (bracketed by the same left paren) ✓

Example:



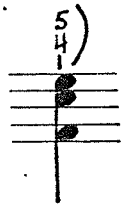
(5 4 1 (are all tied) AND there is ≥1 f_i' to the left of these ≥2 f_i's.

2+4+2QF(:145

2. The right colon is required if

a) two or more $\underline{f'_i}$, delimited on the left by an F, colon or vertical bar, whichever is closest to $\underline{f'_1}$, are bracketed by a single right parenthesis;

Example:



5 4 1)

2+4+2QF145:)$

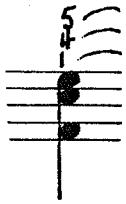
or a Compass direction character

OR

b) all $\underline{f'_i}$ delimited by a colon or F on the left (whichever is closer) and a J on the right have ties. (Fingering ties are discussed below.)

*need to delimit both sides if series of f_i's which are grouped by common tie cond. or paren.

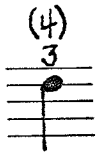
Example:



2+4+2QF145:J

3. In all other cases one or both colons may be suppressed, although it is not incorrect to use a colon on both sides of a parenthesized or tied group of digits. Thus, the Fingering Codes for the three immediately preceding examples could also be F(:145: , F:145:) and F:145:J , respectively.

Additional Examples:

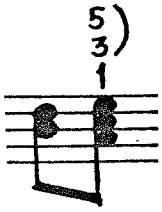


7QF3(4)$

4+2QUF(:12

6+2QF3)(5

6+2QF24(:35:)$

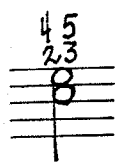


(5+2 4+2+2F1:35:)$)

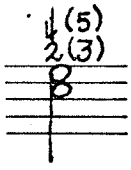
1

Signifies that the fingering digits (and associated modifiers) to follow are in a column immediately to the right of the column encoded before the vertical bar, but nonetheless pertain to the same note or chord.

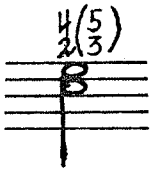
Examples:



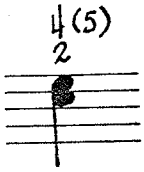
6+2HF24|35



6+2HF24|(3)(5)$

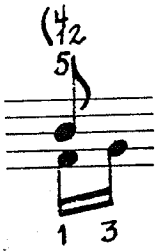


6+2HF24|(:35:)$



6+2QF24|%(5)$

6HOTF2|3|1|3



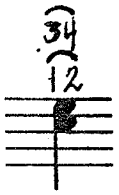
5EUF5(:14|2,((2DFS1 3DFS3))

J

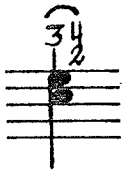
Signifies a tie connecting two fingerings. The J is encoded with the <u>first</u> finger digit for each pair of tied digits. If two or more spatially successive digits in a column have ties, the use of the colon (explained above) is recommended for a more efficient encoding.

Examples:

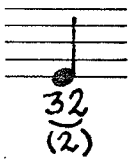
6QF4J|5



6+2QF:13:J|24

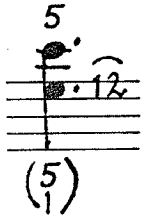


6+2QF13J|24



1QFS(2)3J|2

3QF(24:J



8+5Q.FS(:15:)F%5FE1J|2

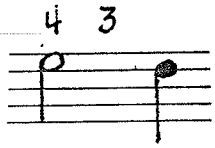
¬d

Links (without any space or other character intervening) two syntactically complete Fingering Codes and indicates that a change in fingering is specified at a point within the duration of the Note but not immediately subsequent to the attack (the latter being implied by the use of the vertical bar as

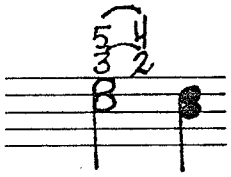
Cumulative δ

explained above). The Push Operator (C3) signifies that the fingering whose code immediately follows ¬d appears at a distance to the right of the last encoded column of fingering equivalent to the duration specified by d , whose meaning is explained in section C3. NOTE: The Push Operator in this context does not irrevocably advance the position pointer, which remains anchored in alignment with the attack position of the Note until advanced by the appropriate Delimiter.

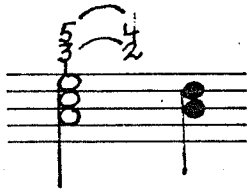
Examples:



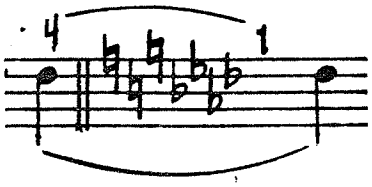
8HF4¬QF3 7Q



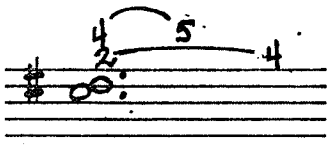
6+2HF35:J¬QF24 5+2Q



4+2+2HF1:35:J¬QF%24 5+2Q



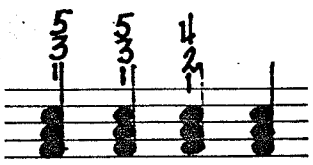
7QJ-F4J7 // !K3*4- 7?0F1J8 Q



6+1#W.F24:J¬HF%5¬HF4

When repeated chords are encoded by means of a Chord Structure Reference with p > 1 (see section B5), the Fingering (if any) is assumed to be given for the first chord only, subsequent chords implied by the Reference being assumed to have no fingerings. In all other situations the encoding must appropriately vary.

Example:



!|3|2+2+2QF135 ||F135 |,,2|F124

Ornaments and Editorial Accidentals

Both standard ornaments and editorial accidentals are treated as DARMS Ornaments. These have the general form

Ox

where x is the code for the sign (or complex of signs) in question.

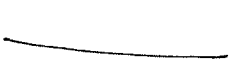
Ornament Codes are normally encoded as part of the Articulation-Fingering-Ornament-Slur complex, all of whose elements, in a given direction, are ordered according to their proximity to the notehead. Thus, if two syntactically complete Ornament Codes appear in this complex, they will be assumed to be vertically aligned, the first encoded being closer to the notehead. Ornament Codes may also be "attached" to these and other Note attributes (see section B7 "Attach Codes for Notes," as well as below, "Ornaments as Attached Symbols.")

If x or any discrete part of x is enclosed in parentheses, all of x or the relevant part of x is assumed to be parenthesized. If < > are encoded in the place of parentheses, then square brackets are represented.

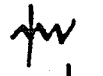
parens, brackets

If x is an integer, the ornament encoded will be assumed to correspond to the entry numbered x in the List of Signs printed in Volume VI of Grove's Dictionary of Music and Musicians Sixth edition (1954), 441-448.

If x is not an integer, it may be any meaningful combination of the codes listed below, by which the most common ornaments may also be encoded.

Sign or Element	Value of x	Sign or Element	Value of x
	N		C+
	M		C-
	L		+C
	J		-C
	%		?
	T		;
	2S		2N
	5S		5N
		sign_y sign_x	x|y

Examples:

Sign	Code	Sign	Code
	LM		C+N-C
	LM-C		C-M-C
	LN?		N1
	N+C		N2
	N-C		M1
	M-C		M3
	C+M		NJ
	C-M		

Remarks on the above examples

1. The code letter C is never used alone, being normally associated with M or N. In these contexts,

   a. the prefix C+ is considered to include the initial upstroke of M or N; and

   b. the suffix -C is considered to include the final upstroke of M or N.

   Thus,  combined with  produces  C+M

   and  combined with produces  M-C

2. Catenation of codes means the symbols are in the same horizontal plane; if two ornaments are vertically aligned, their codes are separated by a vertical bar, the code to the left of it referring to the lower symbol.

Example:

M|?

3. The integer suffix to M and N designates the "valley" (counting from the left) in the ornament through which a slash is drawn.

4. The hook symbols (codes 2N, 2S, 5N and 5S) appear only at terminal points of a line. The code characters 2 and 5 have been chosen because their curved postions resemble the two types of hooks (each of which can be rotated).

Lines with such hooks may be encoded as having horizontal or

vertical extent. The normal function of such symbols is to indicate arpeggiation, and therefore they are encoded as symbols of vertical extent having the form

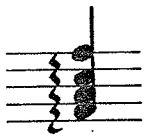
$$\sigma_1|\sigma_2\,?\underline{l}\underline{h}$$

where $\sigma_1|\sigma_2$ gives low and high terminals, respectively, of the entire symbol (hook included);

?$\underline{l}$ is the Dictionary Code for the line (normally $\underline{l}$ = ;), and

$\underline{h}$ is the two-character hook code, as defined above.

Example:



0|9?;2S,,2+2+2+3QU

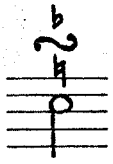
Editorial Accidentals and Accidentals on Ornaments

Editorial Accidentals are not part of the composer's original text and are therefore treated differently than normal accidentals (see section D2). Since they appear in the same position as ornaments (i.e., above or below the note), they are encoded as ornaments. When an editorial accidental appears in the normal position to the left of a note, whether or not it stands in parentheses, and whether or not it is reduced in size, its encoding will be done as for standard accidentals.

Accidentals on Ornaments

Accidentals on Ornaments are encoded as "attached" symbols, using the pertinent ornament as the point of reference. See section B7 for further explanation of attached symbols, and D2 for a listing of Accidental Codes.

Example.



6HO?AS*AN-

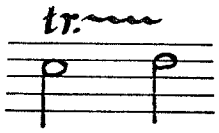
Since accidentals associated with ornaments are normally reduced in size, a Size Reduction Code is not required for encoding such accidentals. If, however, the accidental is of standard accidental size, that is, is not Size Reduced, then it must be encoded as ?|∝| , where ∝ is the Accidental Code for the symbol(s) in question. Such a code will be preceded by the appropriate Attach Prefix in the context under discussion. See section E8 (opening paragraph) for further information.

Ornaments with Horizontal Extent

The horizontal extent of trills can be encoded by means of the Stretch Operator (see section C3). If the distance stretched is equal to the Duration of the Note or tied se-

quence of notes with which it is associated, the Stretch Code for the distance stretched may be suppressed.

Example:



6HO;→H 7

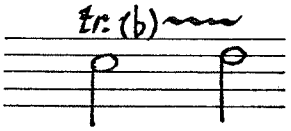
or

6HO; 7

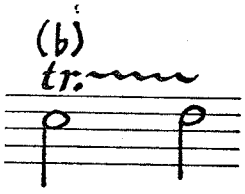
If symbols (such as accidentals) are attached to a trill of more than minimal horizontal extent, they are encoded

a. as a simple Attach Code to the ; (which stands for "tr" with a wiggly line) which will be assumed to have the "tr" as its point of reference;

Examples:



6HO;AE(-) 7

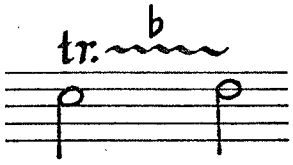


6HO;AN(-) 7

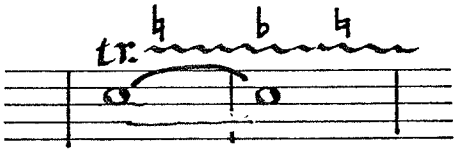
or b. as an Attach Code with a Push Operator internal to it if the symbol represents something occurring subsequent to the point of attack of the Note. There may be more than one Push Operator within the Attach Code, the number being equal to the number of symbols placed subsequent

to the attack point. Note that no Delimiter separates Attach Prefix, Skip Code and the code for the attached symbol.

Examples:



6HO;AN¬Q-^7



/ 6WJO;ANE*AN¬W-¬H* /.6

In the second example it can be seen that

1. the first natural sign is attached northeast of the "tr" sign;
2. the remaining accidentals are attached north of the wiggly line at intervals of a whole and half note from each other (two accidentals being therefore accomodated in one Attach Code); the position pointer is "anchored" (see C3 "Positioning Symbols with the Skip Code.") however, to the attack point of the first note in the trill.
3. the wiggly line extends (and therefore is encoded as stretched) for the full value of two

whole notes; thus, the entire trill is treated as one Ornament Code although it applies to more than one note. This is because the trill line is a single graphic entity.

Ornaments as Attached Symbols

If an ornament is vertically aligned with the notehead the normal Order of Encoding (see section D1) will normally suffice to represent its position with sufficient accuracy. However, if further precision is desired, the ornament may be encoded as a symbol attached to some other attribute (see section B7, "Attach Codes for Notes).

When an ornament is positioned subsequent to the attack point of the Note being encoded, because the ornament is graphically associated with a Note attribute of horizontal extent (slur, beam, etc.), the ornament must be encoded as an ornament attached to the symbol of horizontal extent. For an example, see section B7, "Attach Codes using the Push Operator."

slur without id ends on very next note in next slice which has a note

Slur Codes

The symbol for slur is L (mnemonic "link" or "legato"). Dictionary (E8),

Articulation (D9), Fingering (D10), Ornament (D11), and Slur Codes form a complex whose internal ordering is determined by proximity to the notehead.

As the Tie is considered a species of Slur for purposes of encoding, it is possible to encode all Ties as Slurs, leaving the functional distinction between them to be made at a later stage of processing.

There are two classes of Slur Codes: in next slice which has a note

A. Simple Slur Codes are employable only when a slur terminates at the very next Note encoded for the active Instrument. A Simple Slur Code has no identifier and is given only on the first of the pair of slurred Notes. When converted to canonical form the codes for both Notes will be assigned appropriate Slur Codes with Identifiers.

B. Slur Codes with Identifiers pairs are employable in all contexts involving slurred Notes. The Slur opening will be designated by an odd-numbered identifier and its termination by the next-higher even identifier.

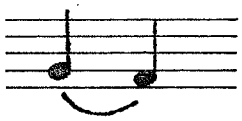
Each time a slur ends the identifier pair becomes available for reuse. However for ease of syntax checking and correction it is suggested that Slur Code Identifiers not be reused in

close proximity to one another. The use of identifiers permits any number of Slurs to be active in the encoding stream.

Multiple slurs on the same note, like multiple articulations, are encoded according to their proximity to the notehead and to other slurs, articulations, and ornaments.

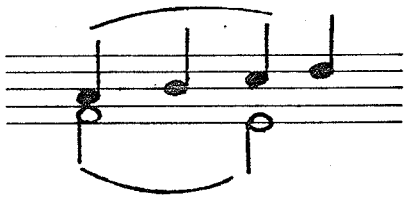
In Linear Decomposition Mode, Slurs may not begin in one layer and end in another. It is possible, however, to encode slurs that begin in one Instrument and end in another. (See section C3, "Codes Incorporating Both the Push Operator and Instrument Codes.")

Examples:



3QL 2

2QL1 3 4L2



2HDL1,4QL3 5U 1HDL2,6QUI4 7U

Special Cases

The precise drawing of slurs is normally an editorial matter, and therefore the encoder need not concern himself with whether the slurs are above stems or below noteheads. However, whenever the shape, direction, vertical extent, or some other

aspect of a slur is unusual or is ambiguously or inaccurately represented by normal Slur Codes, the following additional facilities should be employed to produce more faithful encodings.

1. Dotted and Dashed Slurs

a. Slurs that are dotted lines are represented by suffixing a colon ("dots") to the opening Slur Code.

b. Slurs that are dashed lines are represented by suffixing an equal sign ("dashes") to the opening Slur Code.

2. Intermediate Slur Codes

Intermediate notes embraced by a slur are not generally given Slur Codes, since it is understood that they are affected by the slur previously opened. Incases where this is not obvious, however, sufficiently many of the intermediate notes encompassed within slur i (i being the odd integer Slur Identifier) must have an Intermediate Slur Code of the form

(Li)

in the regular Slur Code position to make the situation clear. To call attention to this abnormal situation, the slur opening (only) must be encoded 2/26/83

!Li

where, again, i is the identifier ~~if one is necessary~~.

need for intermediate slur & conditions for shortform slur are mutually exclusive!

Example (suppressing all other information):

!L2 (L2) (L2) L3

The above represents the Slur Codes of a four note phrase

where the Slur could otherwise be interpreted as not associating the middle notes with those at the terminal points of the slur.

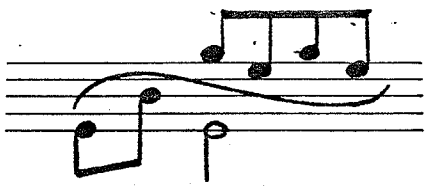
N.B. If a dotted or dashed slur or slur segment is involved, the "break" character must be restated on every Intermediate Slur Code. Otherwise, a solid line will be assumed from the point at which the break character is suppressed. This permits the encoding of slurs that are partly solid lines andpartly dotted or dashed.

3. Slur Direction Indicators

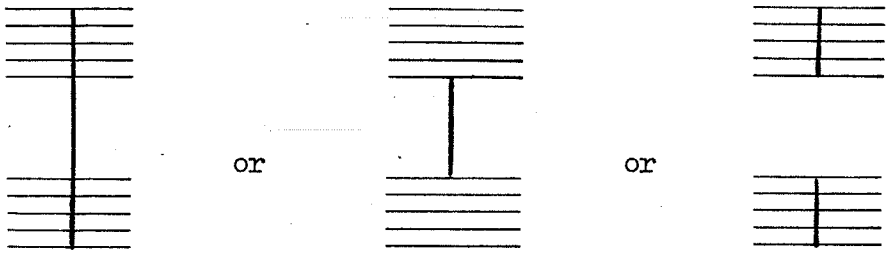
The characters + and - specify that the slur arches upward or downward, respectively. The appropriate symbol is suffixed (when needed) to Slur Codes for the first and any intermediate Notes embraced by the slur.

If the slur twists and thus reverses concavity, a new Slur Direction Indicator is given with the note at which the change of concavity occurs or, if the change occurs between notes, by "pushing" to point of change, as in the example in C3, "Codes Incorporating Both the Push Operator and Instrument Codes."

Example:



(1D!L3+ 5) 1H,(30U(L3-) 8 30 8L4



the single statement of the Barline Code will suffice to cover all staves of the Instrument so long as the staves share a common meter. When this is not the case, the Barline Code applies only to the stave being encoded (as defined by the canonical value of the last Space Code encountered in the encoding stream).

If it is nonetheless necessary to specify a Barline of specific position and extent, this may be done with a code of the form

$$\sigma_1 | \sigma_2 \underline{b}$$

where $\underline{b}$ is any standard Barline Code as defined in the opening paragraph, and

σ_1 and σ_2 are the Space Codes representing the low and high terminal points respectively of the barline.

If, for analytical or data processing purposes (such as more complete error-checking), the encoder wishes to indicate barlines that have been suppressed in the score, he may do so by suffixing an asterisk to the Barline Code.

Examples:

/* /.* //* etc.

A fermata over (under) a barline is represented by '/;' (or '/;2') Cf. D9.3.

Parenthesized or bracketed Barlines are encodable, e.g., '</>', '(71|29/.)', etc.

Codes for Dynamics

The general form of a Dynamic Code is the concatenation of

Pseudo-Space Code (B6) (default 50)
Space Code
Attach Prefix (B7)
Double-Space Code (B2)
} V(v) Literal, if any

where v is 1) any meaningful combination of the "volume" symbols

f p m s z r -

and/or

2) for the representation of hairpin dynamics,

<i crescendo hairpin <

or

>i decrescendo hairpin >

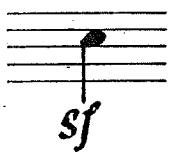
where i is an integer identifier. The parentheses apply only to dynamics in parentheses. Note that words like crescendo are treated as Literals (E6) and are therefore not treated in this section.

Except for minimal hairpins (those applying to a single note), hairpins require two Dynamic Codes incorporating an odd and next higher even integer identifier, respectively: one marking the beginning and one marking the termination.

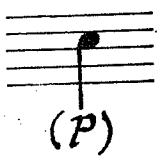
The identifier pair becomes reusable after the hairpin termination is encoded. As in the case of Slur and Beam Codes, the identifiers permit any number of hairpins--all with different identifier pairs, of course, to be active at any given point in the given Instrument. If a hairpin is encoded without identifier, it is assumed to be a minimal hairpin.

Examples:

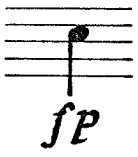
6Q,V<



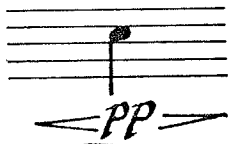
6Q,VSF



6Q,V(P)



6Q,VFP



6Q,V<PP>

2H,V<1 4,V<2

how do we know this isn't bracketed [PP]??

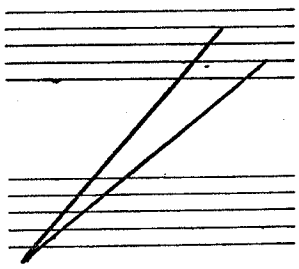
Positioning of Dynamics

Dynamics that are not prefixed with a Space Code, Pseudo-Space Code (B6), or Attach Prefix (B7) are assumed to be in the conventional position under the stave, that is, $\sigma' = 50$. However, one of these codes placed in front of the character V permits the exact or relative vertical position to be represented.

min-hairpins in succession instance must have id-int zero, some such-

Examples:

6H,00VP<1 7,00V<2F



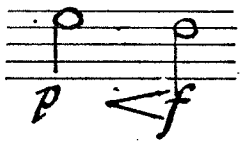
100V<1 5V<2
(all other information suppressed)

Note that the Space Code for the open end of the hairpin above is taken to be approximately midway between the ends of the tines of the hairpin.

Horizontal placement with respect to the previously encoded symbol may be specified by

1. the Delimiter on the left of the Dynamic Code;
2. a Skip Operator (C3); see sections E6 and C1 for detailed information;
3. an Attach Code Prefix (B7).

Example:

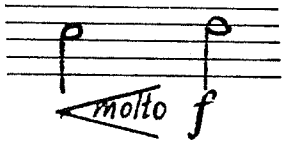


8H,VP,¬Q,V<3 7,V<4F

Hairpins with Literals

If a literal is contained within the extent of a hairpin, its position relative to the left terminal of the hairpin is indicated by means of an Attach Prefix. The entire complex is represented as the concatenation of hairpin representation, Attach Prefix, and Literal Code, respectively. The Skip Code may also be employed for a more precise representation.

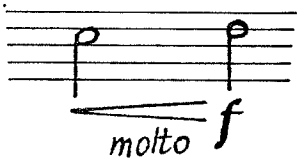
Examples:




```
6H,V<1AE!@MOLTO$  7,V<2F
```

more precise:

```
6H,V<1AE-E!@MOLTO$  7,V<2F
```




```
6H,V<3ASE!@MOLTO$  7,V<4F
```

more precise:

```
6H,V<3AS-E!@MOLTO$  7,V<4F
```

See also "Attach Codes using the Skip Operator" in section B7.

Codes for Literals

This category Literal includes text, tempo indications and other natural-language information.

A Literal is encoded in the following order:

1. Vertical Position — Optional Space Code, Pseudo-Space Code, Double-Space Code (for oversized characters), or Attach Code; 50 is the default value.

2. Typeface — @ (Roman)
   OR
   !@ (Italic)
   (The typeface codes may be used alternately within a literal string to indicate change of typeface.)

3. Literal information — Any meaningful combination of

   1. Letters, digits and the characters
      + - = " ' ( ) ! . , ? / : * ; blank
   2. Control codes — omitted: @ % _ < > # $ _ ¢ | & ¬
      ¢ Letter immediately following is capitalized
      OR
      if shift lock on, release it.
      ¢¢¢ Shift lock mode (all capitals); released at end of Literal or by ¢ within the same Literal.
      ¬d Skip Operator to advance horizontal-position pointer; see section C3.
   3. Codes to represent special typographical features

red-underscore chars have control-meaning

#x Diacritical marks: x (given in the table below) specifies the particular symbol.

%x Sub- and superscript Mode and cancellation, depending on x, which is defined below.

4. |...| , where the string between the vertical bars is any legitimate DARMS representation of a music symbol or symbols.

4. $ Termination of Literal; followed by a Delimiter.

Further Remarks

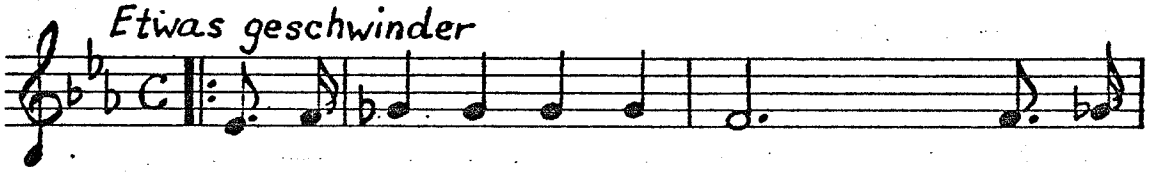
1. For literals associated with dynamics, see E5.

2. The Skip Operator (C3) may be employed within a literal string to specify the temporal distribution of literal information within Linear Decomposition Mode (C4). The Skip Operator will not be taken to be part of the literal text itself. This facility is particularly well suited to the encoding of texts of vocal music, as illustrated below.

3. Pseudo-Space Codes (B6) should be used to indicate the relative placement of each line of text of a strophic song (assuming that each syllable of every stanza is given under its associated note). Thus, if the first stanza is represented as being in the horizontal plane defined by Pseudo-Space Code 50, the second will be below it at 50-1, and so on. This permits the representation of several layers of dynamics and other literal information ("cresc.", etc.), appearing between the stave and the first stanza of text, to be encoded at 50+1, 50+2, and so on, as needed.

Note that, although it is possible to encode a vocal work as a series of vertical slices containing vocal melody, text and accompaniment, it is easier and more economical to encode each stanza of text as a separate "layer" by means of Linear Decomposition Mode, as shown below. Note also that the Pseudo-Space Code prefixing a Literal Code defines the horizontal plane for the entire text of the literal in question.

Example:



Glau-be fest an Gott und Herz! Glau- be
Hof- fe dir Un- sterb-lich- keit, und hie-

```
!G !K3- !MC,00@¢ETWAS GESCHWINDER$ !//: !&, 1E. 2S
  / 3-Q 3 3 3 / 2H. E. 3-S / &,
!&, 50@¢GLAU- ¬E.,BE ¬S,FEST ¬Q,AN ¬Q,¢GOTT ¬Q,UND
  ¬Q,¢HERZ! ¬H.,¢GLAU- ¬E.,BE $&,
  50-1@¢HOF- ¬E.,FE ¬S,DIR ¬Q,¢UN- ¬Q,STERB- ¬Q,LICH-
  Q,KEIT, ¬H.,UND ¬E.,HIE- $&$
```

4. An accent or other diacritical mark is encoded immediately in front of the letter to which it refers

Diacritical mark	DARMS Code
' (acute accent)	#'
` (grave accent)	#G
¨ (umlaut)	#:
˘	#U

^	(circumflex)	#<
¯	(macron)	#_
°		#0 (zero)
·		#.
¸	(cedilla)	#C
v		#V
~	(tilden)	#T
″		#"
’	(high comma, off center)	#,
ø Ł	("O" or "L" cut by slash)	#O/ (letter "O") #L/
æ		#AE
œ		#OE
•	(center-line dot)	#D
ı	(dotless "i")	#I

5. The following special characters appearing in printed texts may also be encoded.

Symbol		DARMS Code
†	(dagger)	#+
‡		#F
§	(section sign)	#S
^	(caret)	#>
¿		#?
¡		#!
—	(long dash)	--
‘	(left upper single quote)	#Q

’	(right upper single quote; apostrophe)	'
“	(left upper double quote)	#W
”	(right upper double quote)	"
‚	(left lower single quote)	#'
„	(left lower double quote)	##
[	(left square bracket)	<
]	(right square bracket)	>
<	(left angle bracket)	#(
>	(right angle bracket)	#)

Examples:

„København”

@##¢K#O/BENHAVN"$

J. L. Důsek

@¢¢¢J. L. D¢#VUSEK$

6. Superscripts and subscripts are introduced by the control codes %+ and %- , respectively. Once superscript (or subscript) mode is established by the statement of either code, all subsequent characters are assumed to be superscripts (or subscripts) until

a. the mode of superscription (or subscription) is raised one degree by a restatement of the same code;

b. the other code is stated, establishing the mode of subscription to superscripts or vice versa;

c. the control code %* is stated, thereby cancelling the most recently stated %+ or %- ;

d. the control code %$ is stated, thereby cancelling all superscript and subscript codes then in effect.

e. the Literal itself is terminated, thereby cancelling all superscript and subscript codes then in effect.

Examples:

Theme A_1 begins here.

@¢THEME¢A%-1%$ BEGINS HERE.$

M^{7a}

@¢M%+7%-A$

One degree of size reduction is automatically implied for each degree of subscription and superscription. Thus, a subscript of a superscript will be assumed to be size-reduced two degrees.

Should circumstances require the representation of a superscript or subscript that is not size-reduced, it may be encoded as a DARMS symbol (specifically, a Literal encoded as a Dictionary Code; see section E8) within the Literal, using the vertical bar convention described below. This means that the superscript or subscript will be encoded in the form

?|@ ... $|

where the dots stand for the superscript or subscript that is not size-reduced.

Example:

Theme A_{1a} begins here.

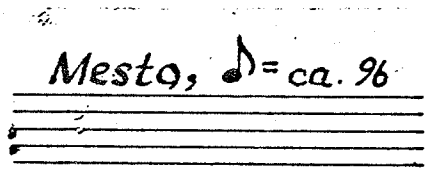
@¢THEME¢A%-|||?|@1A$|||BEGINS HERE.$

The use of the triple vertical bar is explained below.

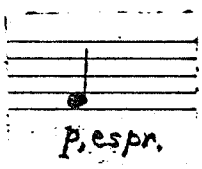
If size reduction of a degree not assumed by the control codes is to be indicated, then an appropriate Size Reduction Code should be employed according to the rules in section F3.

7. Any legitimate DARMS substring may be incorporated into any literal string by enclosing it in absolute value bars and placing it in proper sequence within the literal string.

Examples:



00@¢MESTO,|EU|=CA.96$



2Q,!@|VP|ESPR.$

There are two exceptions to this general rule:

a. a token employing a vertical bar (such as a Chord Structure Reference or Dictionary Code of form ?|...|) may not appear between the "absolute value" bars unless it is bounded on either side by three vertical bars, with no space or other character between the triple bar delimiters and the token they enclose. See the

example in the last paragraph.

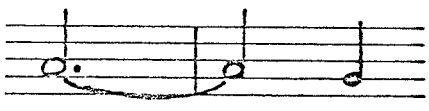
b. Nested literals, that is, Literal Codes within Literal Codes, are not permitted, unless they occur within absolute value bars. Thus,

@...|...@...$...|...$

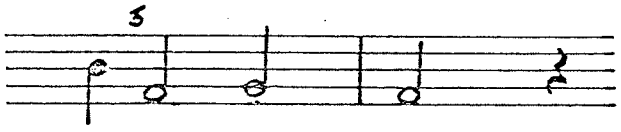
represents the general form of a valid nested Literal Code. The dots stand for missing (or null) information

8. Hyphens between syllables of a word are represented by asingle hyphen, which represents a string of one or more hyphens that extends up to 1) the next syllable or 2) a rest, whichever occurs first. The same is true for the baseline rule (represented by a single underscore character) following the last syllable of a word.

Examples:



(text only) 50@ ... ¢DE-;-√H,UM ... $



way,_____

50@ ... WAY(5)... $ underscore?

example in the previous paragraph.

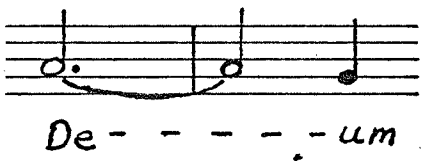
b. Nested literals, that is, Literal Codes within Literal Codes, are not permitted, unless they occur within absolute value bars. Thus,

@...|...@...$...|...$

represents the general form of a valid nested Literal Code. The dots stand for missing (or null) information.

8. Hyphens following all syllables of a word except the last are, of course, encoded as hyphens. However, <u>multiple</u> hyphens bridging widely-spaced syllables may be represented by a single hyphen.

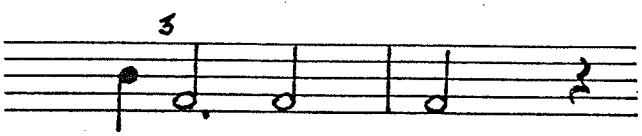
Example:



(text only) 50@ ... ¢DE- —√H,UM ... $

The horizontal extent of a line following the drawn-out <u>last</u> syllable of a word is represented as a line "stretched" the appropriate distance.

Example:



way,_____

50@ ... WAY,—→√H?_... $ (text only)

this is OK (2/14/77)

~~2 commas here.~~

no Delimiter needed!

If for some reason the hyphens or baseline rule extend for a longer or shorter distance than defined above, then a Stretch Code of the form ¬)d , where d specifies the extent of the hyphens or baseline rule, is suffixed to the code character - or _ without any other blank or character intervening. Thus, -¬>/Q indicates that the hyphen string extends one quarter note into the next measure.

Repetition Signs

Repetition symbols, indicating the number of times a measure, or a figure in a measure is repeated, are as follows:

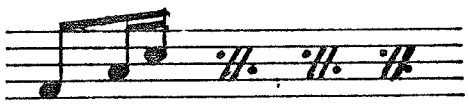
%

%0 (zero)

%1 through %6, depending on the number of slashes.

Parenthesized or bracketed Repetition signs are encodable, e.g., '(%)', '<75%>', etc.

Examples:



(2 (4 6)) %0 %0 %0

((2 6 4 6)) %2 %2 %2

All repetition signs begin with a Space Code $\underline{5}$. The default is the Space Code of the center line of the stave referenced by the last Space Code explicitly or implicitly encountered in the encoding stream before the repetition sign code. Thus, in '... 74$\underline{5}$, %...' or '...!+50,4$\underline{5}$, %...', the value of $\underline{5}$ in the canonical form of the repetition code is 75.

Dictionary Codes

The Dictionary is an extensible facility designed to make available to the encoder four types of codes:

1. Codes for music or graphic symbols that occur rarely or in special contexts only;
2. Codes defined by the user for symbols that are not otherwise found in DARMS;
3. Codes for all DARMS symbols defined elsewhere in this manual, making it possible to represent them outside their normal contexts. This is accomplished by enclosing the non-Dictionary symbol code in vertical bars. For example, an accent listed in the table of section D9 may be represented as being an independent DARMS token (and not an attribute of some Note) by being encoded as

   ?|>| ;
4. Codes to be assigned for music printing applications.

Dictionary Codes have the general form

s?xic

where s is 1. a Space Code (B1) or Pseudo-Space Code (B6), or
2. an Attach Prefix (B7), or
3. a Double-Space Code (B2), required only for symbols of vertical extent perpendicular to the stave, or

Pg. E8.2: add to 4. at top of page: "If the default is an Attach Code Value, this substring will not be supplied explicitly in canonization, but will always be assumed in subsequent processing of the canonical dataset."

4. null, in which case the default value for s, given in the Dictionary Table below, is assumed;

? signifies a Dictionary Code;

x indicates the Dictionary symbol type (see the table below) or any DARMS symbol enclosed in vertical bars.

The value of x may be enclosed by ( ) or < > to indicate that the symbol is enclosed in parentheses or square brackets, respectively;

i is an integer identifier, required only for symbols of horizontal extent or symbols of vertical extent that begin in one Instrument and end in another; and

c is a compass direction, required only for symbols (for example, arrows and some brackets) that can be rotated.

Dictionary symbols may be encoded 1) as part of the code for a Note, Rest, Barline, etc. or 2) as an independent token.

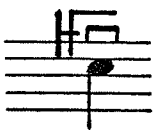
1. If associated with a Note, a Dictionary Code (or complex of Dictionary Codes) will occur in the attribute position corresponding to its meaning. Thus, if the Dictionary symbol is an articulation mark (accent, harmonic, bowing, etc.), the Dictionary Code for the symbol in question will appear in the position of an Articulation Code (D1 and D9). Dictionary symbols may also be "attached" to Note attributes, as explained in section B7.

N.B. Since a token beginning with 'i?', where i is any integer, will be assumed by the DARMS software to be an independent Dictionary code (see (2) below), a Dictionary

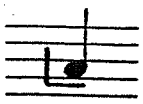
Examples:

Could use examples here for rest barline with dict attachments

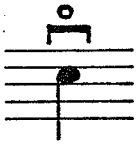
6Q?Q



6Q?D?H



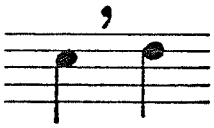
4Q?L



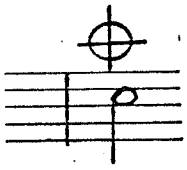
6Q?D?O (letter"O")

2. Should a Dictionary symbol occur in a context independent of a Note, Rest, Barline, etc., it is encoded as an independent token with a Delimiter on both sides of the Dictionary Code.

Examples:



6Q ?, 7



/ ?G,,6H

Symbols in the Dictionary may have a fixed shape or may vary in length horizontally, vertically, or both.

Symbols of Fixed Shape include, in addition to those signs having a standard shape and size, (e.g., downbow mark-

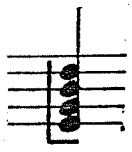
ings), bracket forms whose height and length do not encompass more than the equivalent of one notehead and whose shape corresponds to that given in the table below.

Symbols Variable in Vertical Extent include all those forms whose vertical dimension embraces more than the equivalent of one notehead. For symbols aligned perpendicular to the stave, vertical extent is specified by a "Double-Sigma" prefix (B2) of the form

$$\sigma_1 | \sigma_2 \quad ,$$

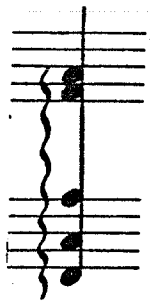
where σ_1 and σ_2 are Space Codes defining, respectively, the low and high vertical limits of the symbol being represented. Symbols which are not aligned perpendicular to the stave are treated as having horizontal extent (see below). If the symbol occurs in a different vertical slice from the one containing the chord or whatever the symbol refers to, the Double-Comma Delimiter (C1) is used to indicate their separation. In this way, the DARMS representation will capture the conditions of graphic separation and musical association.

Examples:



19|8?L,,1+2+2+2Q

1|7?:



70|5?;,,70+4+5QU1,2+2U1 OR→ 2+2+46+4+5QU1

Symbols Variable in Horizontal Extent are encoded by means of identifiers, as described in section C2. A subset of these, for example, horizontal lines, may also be encoded with the Stretch Code (C3), in which case the complete information for the symbol, including its horizontal extent, will be given when the left terminal of the symbol is encoded. No obliquely positioned symbols may be encoded with the Stretch Code. } but see ex. on C3.7

N.B. [Two Dictionary Codes of horizontal extent simultaneously active in the same Instrument may not be assigned the same pair of identifiers regardless of whether they represent the same or a different symbol.] However, as soon as the termination of such a symbol is encoded, the identifier pair employed becomes available again for any Dictionary Symbol of horizontal extent.

TABLE OF DICTIONARY SYMBOLS

x	Symbol	Explanation	Fixed Size	Horiz.	Vert.	Default	Note No.
0 (zero)	null symbol						14
1-9	unassigned	reserved for music-printing purposes					
10-99	unassigned	reserved for user-defined symbols					
100--	unassigned	reserved for future DARMS extensions					
A l i c	→	arrow	X	X	X	AE	1
B	unassigned						
C	𝄵	meter signature	X			25	
D	⌐	downbow	X			AN	2
E	unassigned						
F	⌜	angle bracket (for a note or literal)	X	X	X	ANW	3
G	𝄋	segno	X			00	
H	⌜	Hauptstimme	X	X		ANW	4
Ic	[	square bracket	X	X	X	AN	6
J	⌟	angle bracket	X	X	X	ASE	3
K	unassigned						
L	⌞	angle bracket	X	X	X	ASW	3
M	unassigned						
N	⌜	Nebenstimme	X	X		ANW	4
O	o	offen (horn); harmonic (string)	X			AN	5

P	Ped. or ⌊‾‾	pedal (down)	X	X		see 100 note 10	10
Q	ϕ ϕ	snap pizz.	X			AN	
R	unassigned						
Sc	⎨	brace	X	X	X	AW	6
T	⌉	angle bracket	X	X	X	ANE	3
Uc	∪	organ pedal: heel	X				7
V	∨	up-bow	X			AN	2
W	unassigned						
X	unassigned						
Y	unassigned						
Z	unassigned						
¢	ϕ	meter signature	X			25	
*	✳ or ⌊‾‾⌋	piano pedal release	X	X		see note 10	10
¬	⌉	termination of ⌐ or ¬ if graphically separate	X			ANE	4
+	+	stopped (horn)	X			AN	
@	○	rehearsal number/letter	X			00	8
#	□	rehearsal number/letter	X			00	8
,	,	Luftpause	X			00	
;	~~~	wiggly line see D11.4, ex.		X	X	AE	
:	•••••••	dotted line		X	X	AE	

4. Parenthesized or Bracketed Slurs

Slurs enclosed in parentheses or square brackets are indicated by enclosing the Slur Code for the slur opening only in ( ) or < > , respectively. Note that both left and right parentheses or brackets will be used (indeed, must be used to avoid ambiguity) and that the entire code for the slur opening is enclosed within them.

5. Slurs with Multiple Beginnings or Multiple Endings

See section C3.

6. "Slurred" Fingerings

Changes of fingering on a Note indicated by slur-like symbols connecting two fingering digits are treated as a special case of tie. See section D10.

Summary

The following summarizes the order of encoding within Slur Codes. Of course, not all elements will be present in a given Slur Code.

(or < ! L identifier = : + -) >

Figured-Bass Codes

Figured-Bass symbols appear either above or below the bass melody with which they are associated. They are encoded immediately in front of the Short Form Beam Close, if any.

A Figured-Bass Code has the general form

¢cx

where ¢ indicates Figured Bass (mnemonic: "chord");

c is the compass direction N or S (default N) indicating that the figures are respectively above or below the relevant stave; c in any case need be given only for the first-encoded figure and if and when the value of c changes; and

x is the code for the particular Figure.

In this discussion, the proper noun Figure refers to the complex of music-notational symbols specifying a given chord in a score. Figures such as $\begin{matrix}6\\ \flat\end{matrix}$ and $\begin{matrix}4\sharp\\ 2\end{matrix}$ are then said to consist of (a column of) several Elements, each Element consisting of one or more symbols. Therefore, 4♯ and 2 are the two Elements comprising the Figure $\begin{matrix}4\sharp\\ 2\end{matrix}$.

The rules for the encoding of Figured-Bass information are as follows:

1. Numeric elements are represented by numeric characters.

Figure	Code	
6	6	
10	|10|	(Regarding the vertical bars, see Rule 3 below.)

2. Graphically distinct accidentals (i.e., those not incorporated physically into the numeric figure as shown in Rule 5 below) are represented by the corresponding DARMS Accidental Codes (D2).

Figure	Code	
♭	-	
𝄪	|##|	(Regarding the vertical bars, see Rule 3 below.)
etc.		

3. Elements whose representation requires two or more characters are encoded within vertical bars to preclude ambiguity. However, in some instances of multi-character Elements (see Rule 5 below), the vertical bars may be suppressed without introducing ambiguity. Colon pairs are used to indicate the vertical extent of bracketings (Rule 7, below).

Figure	Code
6♯	|6#|
10♭	|10-|

4. Figures consisting of two or more Elements, vertically aligned, are encoded from bottom to top, that is, the reverse of the order in which Figures are "read" in musical discourse. Thus, a "six-four" is encoded as "four-six".

Figure	Code
6 4	46
6 4#	|4#|6
6♮ 3	3|6*|

5. Elements that graphically combine the numeric character specifying the interval and an accidental or slash such that the elements are physically connected are encoded in the form

¢i&α

where i is the numeric character representing the interval, and

α is a one-character Accidental Code.

It will be noticed that a single DARMS code may stand for the several forms of a given symbol class; the diversity in shape is due to publication conventions rather than semantic differences; only one form of each symbol will be found in a given score.

A. The following Figures and Codes represent sharped or augmented intervals or cancelled flats:

Figure	Code
1♯	1&#
2♯ or 2+	2&#
3+	3&#
4+ or x4	4&#
5+	5&#

(Figure)	(Code)
8	6&#
7 or 7	7&#
8	8&#
9 , 9 or 9	9&#

B. The following Figures and Codes represent flatted or diminished intervals or cancelled sharps:

Figure	Code
2 or 2	2&-
3 or 3	3&-
4 or 4	4&-
5 , 5 , 5 or 5	5&-
6 or 6	6&-
7 , 7 or 7	7&-
9	9&-

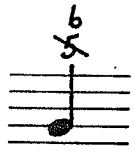
C. The following Figures and Codes represent naturalized intervals:

Figure	Code
2	2&*
3	3&*
4	4&*
5	5&*
6	6&*
7	7&*

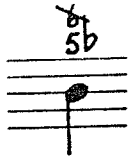
Because the ampersand may not legitimately <u>begin</u>

any Element and will perforce be followed by a one-character Accidental Code, it is <u>not</u> necessary to enclose these three-character codes within vertical bars. In other words, the Figures and Codes given in this section are exempt from Rule 3.

Examples:



3Q¢5&-6

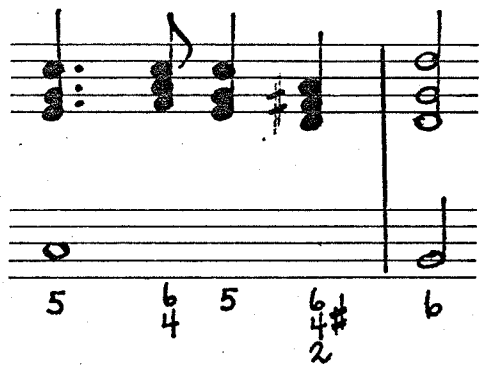


5Q¢|5-|6&#

~~6.~~ ~~A~~ change of Figure within the duration of a note is encoded by the use of the Skip Code (C3), which has the form ¬d and is immediately followed by the syntactically complete Figured-Bass Code for the new Figure. This catenate of codes is itself appended directly to the code for the former Figure. If more than one change of Figure occurs, there will be as many Skip-Code/Figured-Bass-Code pairs, encoded in direct succession, as there are changes of Figure over the given bass note.

Example:

(realization)



4W¢5¬Q.¢46¬E¢5¬Q¢2|4#|6 / 3H¢6

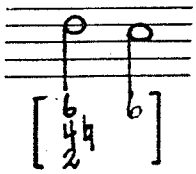
N.B. The upper stave of the example (i.e., the "realization" in an implied treble clef) is not to be considered part of the actual musical text being encoded. For this reason the continuo line is treated as a

Note that, as in the case of Fingering Codes (D10), the Skip Code used within a complex of Figured Bass Codes does not permanently advance the position pointer, which remains "anchored" to the attack point of the (bass) note.

7. Parentheses or square brackets used to enclose part or all of a Figure are encoded in the same way as Fingerings (D10) so enclosed. Thus, when more than two consecutively encoded Elements are embraced by a parenthesis or bracket, pairs of colons are employed to define the extent of the bracketed material. Moreover, when a Figured Bass Code ends in) or > , a $ must be appended to it to preclude it from being taken for a Short Form Beam Closing (D8) or Articulation Code (D9), respectively.

Figure	Code
(6)	¢(6)$
(♭6)	¢(|-6|)$
$\binom{6}{\flat}$	¢(:-6:)$
$\begin{bmatrix}6\\4\\2\end{bmatrix}$	¢24<6>$
$\begin{bmatrix}7\\4\#\\2\end{bmatrix}$	¢<:2|4#|7:>$

Example:

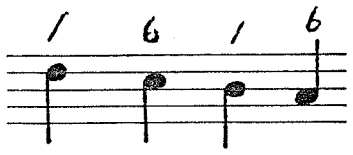
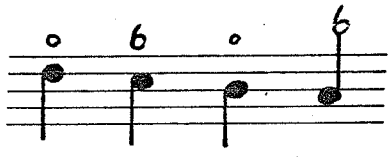
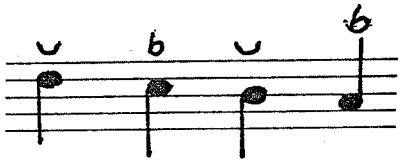


7H¢<:2|4*|6: 6¢:%%6:>$

(Regarding % see Rule 9 below.)

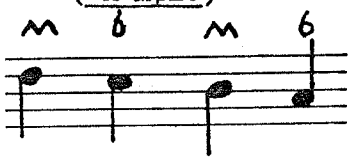
8. This section treats symbols over what Arnold[1] calls "transitional notes"

A. The following symbols are used over notes when the chord indicated by the Figure over the next bass note is actually intended to sound:

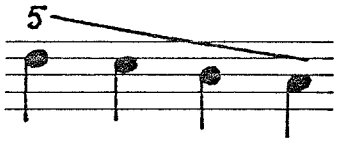
Figure	Code	Example
/	¢/	
o	¢O (letter "O")	
⌣	¢U	

7Q¢/ 6¢6 5¢/ 4¢6

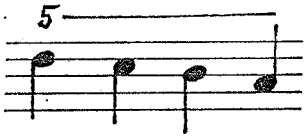
[1] F. T. Arnold, The Art of Accompaniment from a Thorough-Bass, (London: Oxford University Press, 1931; republished in two volumes by Dover Publications, Inc., 1965), 780ff.

(Figure)	(Code)	(Example)
ᨒ	¢M	

B. In cases where a chord is to be continued despite the movement of the bass, a straight line indicating the length of the extension is employed; depending on editorial convention, it may be horizontal or oblique, but in either case it is encoded without Space Codes as a symbol of horizontal extent.



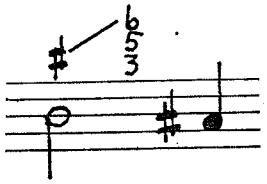
or



7Q¢|5?_7| 6 5 4¢?_8

C. In cases where a chord is anticipated before the bass changes, an oblique line connecting Elements of two successive Figures is employed.

Example:



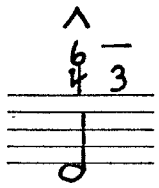
5H¢|#?_3|¬Q¢35|6?_4| 4#Q

Note that, in the encoding of the second Figure, the termination of the line, although it is to the left of the 6, is encoded last. That is, the code for both terminal points of the line is suffixed to the code for the Element with which it is associated. This encoding order is mandated by the ambiguity that would otherwise occur should the code for the line, which terminates in a numeric identifier, prefix the code for an Element that begins with a numeric character.

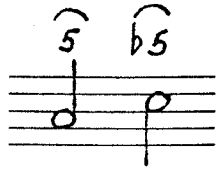
9. Additional Figured-Bass symbols and their codes are summarized below.

Figure	Code	Explanation
-	=	Dash in the position of a normal Element.
null	%	The null symbol may be useful in certain situations to indicate horizontal alignments of successive Figures, or vertical bracketing extending beyond the notated Elements (see example under Rule 7). May be omitted for Elements above the last Element explicitly given in the notation for a given Figure.
∧	V	Symbol used by C.P.E. Bach.
∩	C	Indicates the absence of the 6th (Telemann et al.).
•	°	Used as punctuation terminating an Element (Cavalieri et al.) or to indicate when a change of chord will occur (C.P.E. Bach).

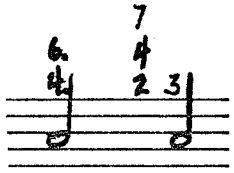
Examples:



20H¢46V¬Q¢3=



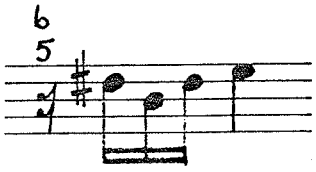
4H¢5C 6H¢1-5|C



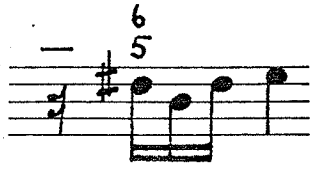
4H¢14.16.1¬Q.¢247 4Q¢3

10. Anticipation: A chord may also be anticipated by means of a Figure over a rest. In such cases the code for the Figure is appended to the code for the rest.

Examples:



RS¢56 ((7# 5 7)) 8Q



RS¢= ((7#¢56 5 7)) 8Q

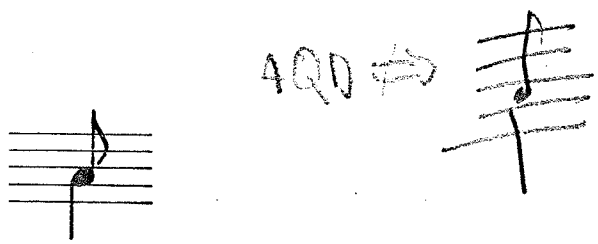
Recycling of Double-Stemmed Notes

Two Notes that share the same notehead (and may differ from each other in all attributes except Space Code, Accidental and Notehead) may be encoded:

A. As two separate DARMS tokens

1. Separated by a Delimiter Comma (C1);

Example:



4ED,Q

or

4Q,ED

(Note the use of Sigma Suppression, B1)

OR

2. In different layers of Linear Decomposition Mode (C4), such that, in the example above, an alternate encoding would be of the form

!& ... 4ED ... & ... 4Q ... &$;

OR

B. As a single DARMS token employing the DARMS facility of "recycling".

Pg. D14.2: Add to #1: "It is recommended (to avoid confusion in the encoder's mind) that the Stem Codes for the first note be supplied, even if the Stem Code may be defaulted."

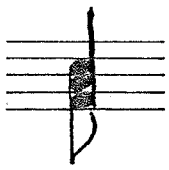
In a recycled representation of a double-stemmed Note:

1. All the attributes pertaining to one Note must be encoded before those pertaining to the other Note;

2. The normal Order of Encoding (D1) is followed for each Note in turn, except that the series of encoded attributes for the second Note must begin with either Duration Code or Stem Code.

3. A corollary of 2. is that the second Note may not be encoded with a Short Form Beam Opening.

4. Recycling is not permitted in either Chord Structure format (B3) unless all notes in the chord have two stems (which is highly unlikely). A chord of the form



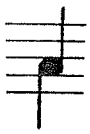
is best encoded as 2+2+2Q,2ED

In this example, the Delimiter Comma and Common Space Code indicate that the noteheads on the first space are congruent.

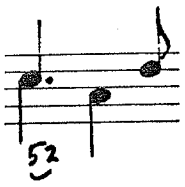
5. Duration Codes for the second Note of a recycled token, when explicitly encoded, must be given in full.

Examples:

4EQD
or
4QDE

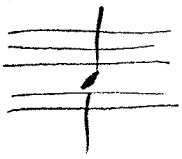


4QUD
or
4QDU
or
4QDQ



6QFS5J|2Q.U 4QD 7EU

Incorrect: 6QFS5J|2.U 4QD 7EU



'4QD' ⇒ must be made clear that there are 2 noteheads and/or 2 stems

* Only shortbeam code allowed is close beams on first note.

Clef Codes

Clef Codes have the general form

$$\sigma!\underline{k}$$

where σ is the Space Code denoting the line referenced by the clef k, which may be G, F, or C. The Clef Code must be followed by a Delimiter (C1).

If the Space Code is omitted the following default values are assumed:

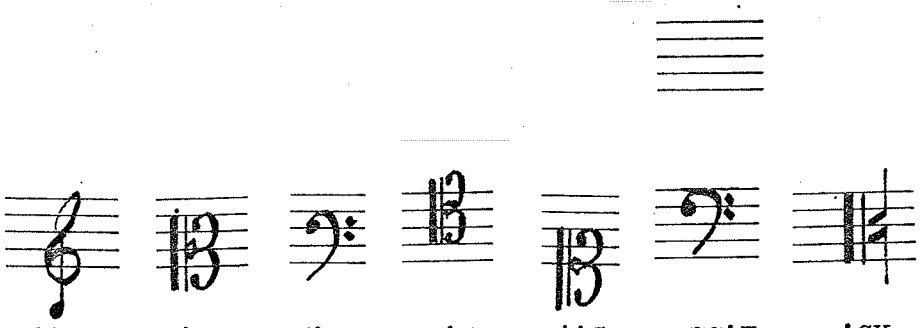
G Clef:	23
F Clef:	27
C Clef:	25

For all other positions, including "normal" clefs on other than the topmost stave of an Instrument, the Clef Code must be introduced with a Space Code.

Examples:

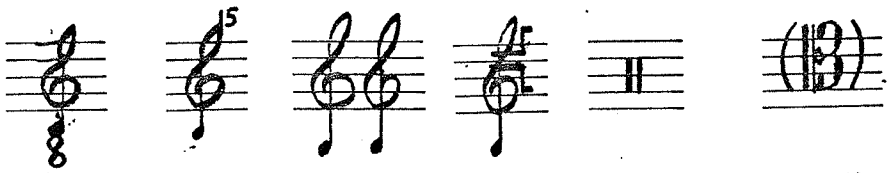
1.	Violin Clef	1!G
2.	Mezzo-soprano Clef	3!C
3.	Baritone Clef	5!F
4.	Tenor Clef	7!C
5.	Soprano Clef	1!C
6.	Bass Clef in normal (two-stave) piano music	77!F
7.	Old-style C Clef	σ!CK

8. Transposing Clefs
   a. with numerals attached — σ!c_n — where c is G, F or C as defined above and n is a signed integer, e.g., !G-8 (down an octave).
   b. compound clefs — σ!c_k1_k2 — where k1 and k2 define the two clef types in their left to right order.
9. Percussion Clef — σ!CH
10. Parenthesized (tenor) clef — σ!C()



1!G 3!C 5!F 7!C 1!C 77!F !CK

Clef Type (as above) 1. 2. 3. 4. 5. 6. 7.

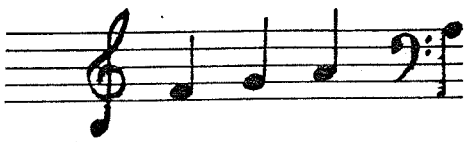


!G-8 !G+15 !GG !GC 25!CH 7!C()

Clef Type (as above) 8a. 8b. 9. 10.

Because of an editorial convention requiring that clefs appearing anywhere except at the beginning of a stave be reduced in size, it will not be necessary to use a

Size Reduction Code (F3) for such clefs, as in the example below.



!G 2Q 3 4 !F 9

If a clef is placed relative to something other than a line of a stave, a Pseudo-Space Code (B6) or Attach Code Prefix (B7) may be substituted for the Space Code of the Clef Code.

Example:

73!GAS!F,!G

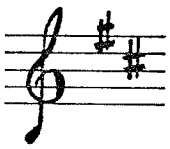
Key Signature Codes

Standard key signatures are encoded in the form

!K$\underline{n}\alpha$

where $\underline{n}$ is the number of symbols α (see section D2) in the key signature. If a key signature is in standard mode it is assumed to hold for all staves of the particular Instrument, with conventional positioning of the sharps and flats in question. If $\underline{n} = 1$, the 1 may be omitted from the Code.

Examples:



!G !K2#

!G !K-

All key signatures must be followed by a Delimiter (C1).

Non-standard key signatures are encoded in the form

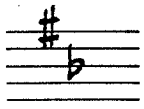
$$!K\alpha_1\sigma_1\alpha_2\sigma_2\cdots\alpha_{\underline{m}}\sigma_{\underline{m}}$$

where $\underline{m}$ is the number of symbols α (sharps, flats, or naturals) in the key signature;

$\alpha_{\underline{j}}$ is the $\underline{j}^{th}$ such symbol (from the left), and

$\sigma_{\underline{j}}$ is the Space Code of each $\alpha_{\underline{j}}$.

Example:



!K#9-5

6/18/83 digit preceding accidental of first signature may not be defaulted in a key change encoding.

Change of a Key Signature

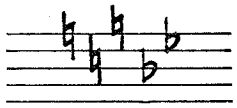
1. For modern practice, with no cancelling natural signs, the new signature is simply encoded where it occurs.

2. For older practice, with cancelling natural signs:

a. !K is followed by a digit and the asterisk, the digit representing the number of natural signs.

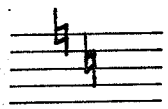
b. This is followed, without intervening blank, by no appropriate to the new key signature.

Example:



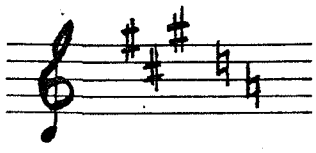
!K3*2-

3. To go from a key signature with accidentals to a key signature without accidentals, the following suffices:



!K*

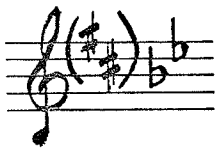
4. To go from a key signature with n sharps or flats to one with fewer symbols of the same kind:



!G !K3#2*

Parenthesized or Bracketed key signatures employ parentheses or < > to enclose that portion of the code to the right of !K that corresponds to the parenthesized or bracketed symbols.

Examples:



!G !K(2#)2-

!G !K<->

Meter Signature Codes

The general form of a Meter Signature Code is

σ!M$\underline{r}$

where σ stands for the Space Code (normally suppressed in favor of default value of 25) of the implied fraction line, and

$\underline{r}$ stands for the meter, normally represented as a ratio.

In rare instances a Pseudo-Space Code (B6) or Attach Code Prefix (B7) takes the place of σ.

If only one Meter Signature Code is given, it will be assumed to hold for all staves embraced by the Instrument Code in force. Staves within a multi-staved Instrument with a meter signature different from that of the topmost stave must be given their own Meter Signature Code, the Space Code of which will unequivocally identify the stave in question.

Examples (signature in normal position):

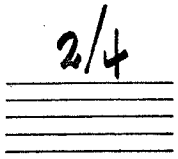

!M2:4


!M3:4


!MC


!M¢

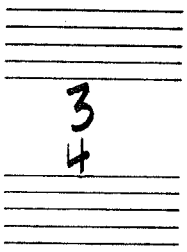
Examples of unusually placed meter signatures:



00!M2/4

OR

AN!M2/4



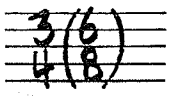
50!M3:4

OR

80|20!M3:4

A Meter Signature Code terminates with the first Delimiter (blank or comma). For a signature made up of two distinct signatures, two separate Meter Signature Codes, separated by a blank to indicate relative horizontal position, are required.

Example:



!M3:4 !M(6:8)

Note also that parenthesized meter signatures are represented by enclosing the value of r in parentheses. Square brackets would be similarly represented by the characters < and > .

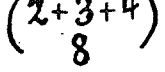
Further examples.

For purposes of illustration, assume that the following meter signatures pertain to the second stave of music for an Instrument and that this stave has a different meter from that of the topmost stave of the same Instrument.

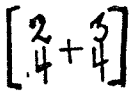
75!M2:4

75!M2/4

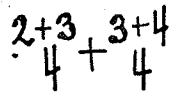
75!M(2:4)



75!M(2+3+4:8)

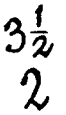


75!M<2:4+3:4>



75!M2+3:4+3+4:4

75!M3:H.



75!M3-1/2:2

If a meter signature is suppressed in a score, but for subsequent data processing purposes the encoder wishes to supply one (as he can also supply "missing" barlines), he can do so by encoding the appropriate meter signature in the form

```
σ!M*r
```

where σ and r are as defined above.

Barline Codes

The Barline Code is made up of the following components, whose ordering within it represents the left-to-right order of barline information being encoded.

/	—	Normal (thin) barline
!/		Thick barline
:		Repeat symbol
/. (N.B.)		Dotted barline (note that ":" usu. indicates "dotted")
/=		Dashed barline

Examples:

// !// /!/ !//: :/!/

The encoder should not encode barlines at the left-hand margin, nor normally be concerned with specifying, in the body of the DARMS code, the vertical extent and placement of barlines, although such information will be specified as part of the input to the DARMS system software.

Thus, whether the barline be of the form

÷	- - - - - -	dashed line		X	X	AE	
_	———	solid line		X	X	AE	
'	8va . . . . ¬	"at the octave"	X	X		00	9
"	15ma . . . . . ¬	"at the fifteenth"	X	X		00	9
<c	V	organ pedal: toe	X				7
%	𝄋	dal segno	X			00	
>	⌒	intermediate piano pedal release	X			see note 10	10
)	⌐———	first or other ending, not the last		X		00	11
(	⌐———	final ending ("ultima volta")		X		00	11
/	/	accent (Schoenberg)	X			AN	
-	⌣	accent (Schoenberg)	X			AN	
|. . .|		any DARMS music symbol not in the Dictionary					12
&i		dummy symbol (patch marker) (or reference ")					13
!	unassigned						
.	unassigned						
$	unassigned						
!	~~unassigned~~						

N.B. The symbols marked with an "X" in the "Horiz." and "Vert." columns are those that may be extended horizontally or vertically.

The following correspond to the references given in the "Note No." column of the Table of Dictionary Symbols.

(1.) Encoded as a Symbol of Fixed Shape, the arrow is assumed to have a shaft of minimum length (as shown in the table). The general form of such a "minimum arrow" is

s?Ac

where s and c are as defined at the beginning of the section.

If the arrow is perpendicular to the stave and is of more than minimal length, the form of its DARMS representation is

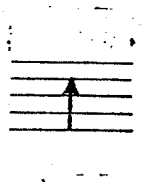
$\sigma_1|\sigma_2$?Alc

where $\sigma_1|\sigma_2$ is as defined above,

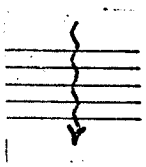
l designates the type of shaft: _ solid (default)
: dotted
= dashed
; wiggly , and

c is S, N or SN, depending on the number and direction of arrowheads.

Examples:



1|7?AN



18|31?A;S

1|9?A=SN

If the arrow has horizontal extent, two codes are required, each having the form

s?Alic

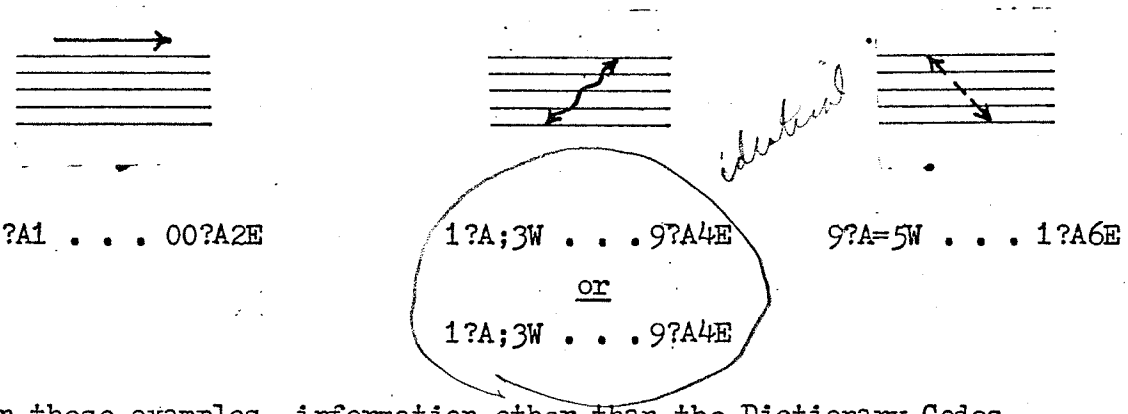
where s and l are as previously defined,

i is an integer identifier, and

c may be one of two legal values: E or W. Because the precise angle of obliqueness is determined by editorial rather than musical considerations, it is unnecessary to specify with greater precision the direction of the arrowhead. If no arrowhead is present at the point represented by the code, then c is omitted.

A value for l need only be given in the code for the left terminal point of the arrow; if not given here, a solid-line shaft is assumed.

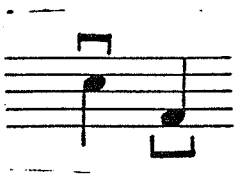
Examples:



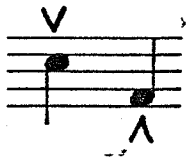
(In these examples, information other than the Dictionary Codes has been suppressed for purposes of clarity.)

(2.) Bowing marks ⌐ and V normally appear above the stave. If they appear below the stave (which can be indicated by "attaching south"), their inversion is assumed.

Examples:



6Q?D 2QAS?D

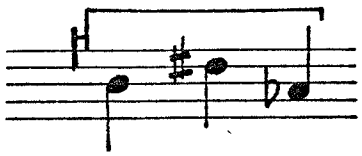


6Q?U 2QAS?U

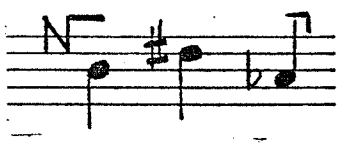
(3.) These bracket codes with right angles are not to be employed for groupette representation. A summary of groupette bracket codes is given in section F1. For further information on bracket code ?I , see note 6 below.

(4.) Haupt- and Nebenstimme signs are treated as symbols of horizontal extent (C2) if they graphically encompass more than one note along the horizontal axis; therefore, the second statement of the code will signify termination of the Haupt- or Nebenstimme. However, there is an alternate notation, involving the two symbols of fixed size which are graphically separate; in this case, the termination of either Hauptstimme or Nebenstimme condition is indicated by the Dictionary symbol assigned code character ¬ .

Examples:



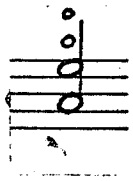
5Q?H3 7# 4-?H4



5Q?N 7# 4-?¬

(5.) In the case of multiple harmonics, the code letter O is stated as many times as there are harmonic symbols.

Example:



4+4H?OO

(6.) The bracket ?I and brace ?S may be represented in various degrees of rotation, but the manner of indicating the rotation of these symbols in their minimal extent is slightly different from that when they have horizontal extent.

a. If the symbol is of minimal extent, the compass direction c may have up to three characters.

b. If the symbol has horizontal extent (and thus requires the use of an identifier pair for its coding), the value of c will be a one-character compass direction: S or N if the bracket or brace is exactly parallel to the stave and E or W in all other cases. (The rationale for the single-character compass direction is the same as that explained in note 1 above.)

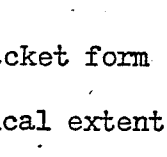
If the symbols are encoded with an Attach Prefix and the direction of attachment is the same as the value of c within the bracket or brace code, then the latter compass direction may be suppressed.

Examples:

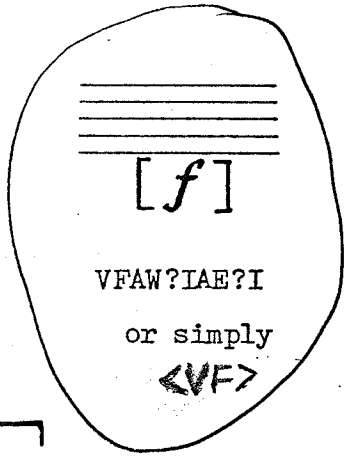
c = N c = E c = E c = E c = S etc.

Bracket form ?I and brace form ?S may have either horizontal or vertical extent, depending on the rotation, but not both simultaneously. In this they differ from the bracket forms ?F , ?J , ?L , and ?T (Note 3, above), which may have both horizontal and vertical extent. Of course, the bracket and brace may be represented as symbols of fixed shape, in which case they embrace no more than a single notehead or character.

Examples:



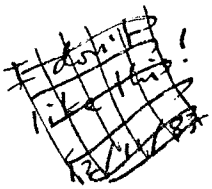
4QAW?I

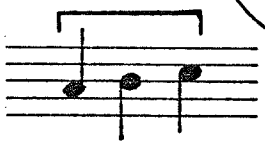


VFAW?IAE?I

or simply

<VF>





4QANW?I3N 5 6ANE?I4N

or

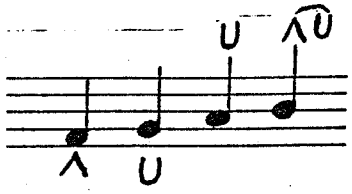
00?I3N,,4Q 5 6,,00?I4N

error *

(7.) Organ pedal markings (∪ and ∧) appearing above the stave refer to the right foot, those below to the left foot.

Unlike upbow and downbow markings (Note 2 above), the symbols below the stave are not inverted forms of those above the stave. The placement of the symbols is signified by the characters N and S suffixed to the code for the pedal symbol.

Example:

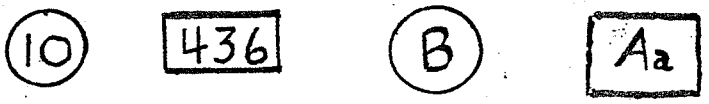


2Q?<S 3?US 4?UN 5U?<NJ|?U

Q Q bec. ?: may be independent Dict. code

✓ (8.) The oval and rectangles surrounding a rehearsal letter or number are assumed to be wide enough to encompass the information inside; it is never necessary to treat them as symbols of horizontal extent. Appended to the code character @ or # is the literal information inside the oval or rectangle. If the literal information is alphabetic (rather than numeric) then the information is encoded as a Literal (E6). [However, if all rehearsal letters throughout the dataset are capitals it is only necessary to set the shift lock mode in the first rehearsal letter code; it will then be assumed to hold for all rehearsal letters until cancelled by ¢ (if at all).]

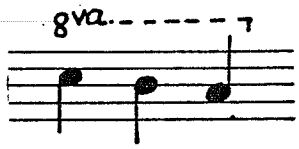
Examples:



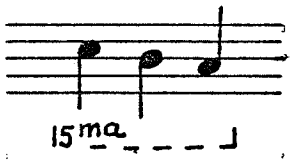
?@10 ?#436 ?@@¢B$?#@¢AA$

(9.) The use of the octave transposition signs 8^{va} and 15^{ma} may be best illustrated through musical examples. Note that the terminal points of the symbol should be encoded as being positioned above or below the first note to which they refer. Note the use of paired identifiers (C2).

Examples:



6Q?'4 5 4?'5



6QAS?"5 5 4AS?"6

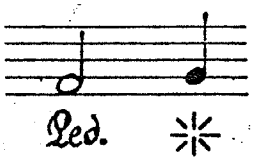
or

50?"5,6Q 5 4,50?"6

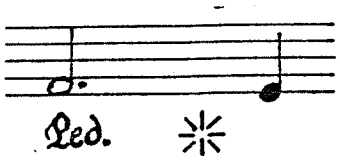
✓ (10.) Piano pedal markings, both "pedal-down" and "pedal-up," should appear directly below the beat or fraction thereof to which they apply. Therefore, to attach a pedal, say, "southeast" of a note would not be good encoding style. Some guidelines are given in the examples that follow. The s default for pedal indications is "below the bottommost stave of the Instrument," that is, normally 100.

There are two species of pedal markings: the first uses discrete symbols of fixed size to indicate down and up pedal.

Examples:



2H?P 3Q?*



2H.?P,¬H,?* 1Q

or

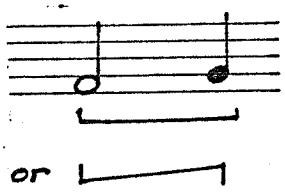
2H.?P!AE¬H?* 1Q

typo?

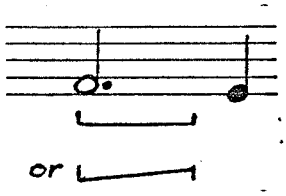
yes, not a typo but it should be made clear that this kind of "nested attach" is not attached to an attached thing but to

The second uses a continuous symbol of horizontal extent that indicates intermediate pedal releases within a "pedaling phrase." In this case the beginning and end of the phrase use the same codes as above except that an underscore is suffixed to both. Intermediate pedal release with immediate repedaling is indicated by ?>.

Examples:



2H?P_ 3Q?*_



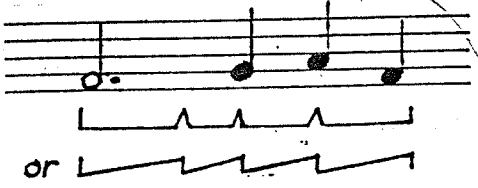
2H?P_,¬H,?*_ 1Q

or

2H.?P_!AE¬H?*_ 1Q

why no identifier as in '?'1'?

Because you can only have one pedaling phrase at one time in an Instr. you turkey!



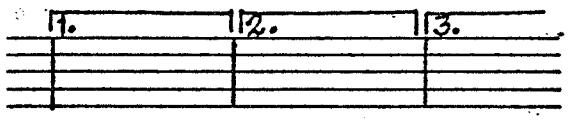
2H.?P_,¬H,?> 3Q?> 4?> 2?*_
Q Q

or

2H.?P_!AE¬H?> 3Q?> 4?> 2?*_

(11.) The encodings of multiple endings will consist of the representations of the endings (treated as symbols of horizontal extent with appropriate identifiers) and a literal internal to the code (as it is internal to the symbol) positioned between the ?) or ?(and the identifier.

Example:



?)@1.$1 ... ?)2 ... / ?)@2.$3 ... ?)4 ... / ... ?(@3.$5 ...?(6 ...

(12.) For an explanation of codes of the form

?!...!

see the introductory discussion of Dictionary Codes at the beginning of this chapter.

(13.) The Dictionary Code ?&i, where i is an integer identifier (odd or even) that ~~is never subsequently~~ may be employed with ~~any~~ some other Dictionary symbol within the dataset, is used as a dummy code to be subsequently replaced by some other string. This facility permits the encoder to pass over places of encoding difficulty which can be "patched up" when the problem has been solved.

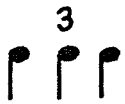
(14.) "The Null symbol can stand for a null (invisible) note to which ties and other symbols may be related (cf. D5.4f.: "Ties Without Terminating Notes" and the topmost example on D10.10). Thus a Null Note may be encoded with attributes such as Space Code, Duration Code, Tie Code, Dictionary Code and Slur Code."

Chord Analysis Symbols

The form of these DARMS symbols is still to be announced.

Groupettes

A groupette is a non-normative subdivision of a metrical unit. For example



is a groupette.

Groupette Definers

The essential information for each distinct groupette type is encoded as a Groupette Definer at some point before the first appearance of the groupette notation. Such a definer is preceded and followed by a Delimiter; the terminating Delimiter, if a Blank, will not advance the position pointer (C1).

Notice that, unlike Equate Strings or Chord Structures, which are defined at the first appearance of the object denoted, Groupette Definers establish durational proportions that must be defined before the first occurrence of the corresponding groupettes. Therefore, it is recommended as good encoding practice that all Groupette Definers be encoded at the very beginning of the data string, before the musical text itself is encoded. Although this involves pre-scanning of the score by the encoder, it will facilitate referencing in case irregularities or errors in the data are discovered or

suspected.

The form of a Groupette Definer is

$$!m\delta_1 i:n\delta_2 j@...\$$$

where m is the number of Notes of duration δ_1 that occupy the same time span that n Notes of duration δ_2 would normatively occupy. The integer Groupette Identifier i uniquely identifies the Definer. The Groupette Identifier j is given only in the definition of a nested groupette as described below. The suffix of the form @...$ is required only when the groupette notation involves one of the special bracketings listed below under "Groupette Bracket Codes." Otherwise, a groupette of the form $\overset{m}{[\ \ \ ...]}$ is assumed, where m is given in the Groupette Definer.

Examples:

!5Q7:4Q !3E1:1Q

Abbreviation Rules

1. If both δ_1 and δ_2 are the same and j is not used, δ_2 may be suppressed;
2. If n = 1, then it may be suppressed.

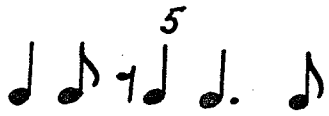
Examples (compare with examples above):

!5Q7:4 !3E1:Q

Representing Notes and Rests in Groupettes

When a Note or Rest belonging to a groupette is encoded, the identifier which refers to the correct Groupette Definer is suffixed to the letter-dot Duration Code. Thus, for an

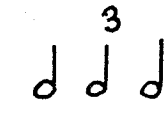
instance of the first groupette formation defined above, such as

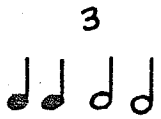


the code would be (omitting Space Codes, etc., for purposes of simplification)

Q7 E7 RE7 Q7 Q.7 E7

Note that although an identifier is defined in terms of two specific durational units, the use of the identifier extends to all normative (i.e., binary) subdivisions of the defined non-normative denomination. Thus, given the Groupette Definer !3H4:W (suppressing all but Duration Codes)

 is represented as H4 H4 H4 and

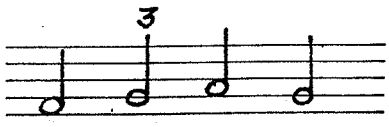
 as Q4 Q4 H4 H4 and

 as Q4 Q4 W4

all being equivalent in duration to a normative whole note.

Delta Suppression is applicable to Duration Codes that include Groupette Identifiers.

Example:



2H4 3 4 3H

Note that the identifier is propagated forward with the letter code; when the normative half-note returns, it must be explicitly given on its first occurrence.

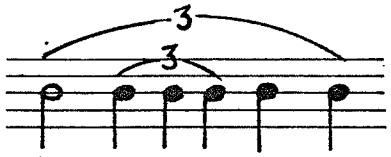
Nested Groupettes

Groupettes may be nested. Nested Groupette Definers are recognizable by the fact that the Duration Codes to the right of the colon have numeric Groupette Identifiers appended to them. Nesting may be to any depth.

Example:

	Groupette Definer
A: Outermost groupette	!3H1:2H or !3H1:W
B: Quarter-note triplet replaces half-note in groupette A:	!3Q2:1H1 or !3Q2:H1 groupette 2 defined in terms of groupette 1 (above): 3 quarter notes labelled "2" take the place of one half-note labelled "1" .

A and B combined



Code: 5H1 5Q2 5 5 5Q1 5

(suppressing groupette bracket information for purposes of simplification here)

Groupette Bracket Codes

As stated above, a groupette of the form 
(where the integer m is given in the Groupette Definer) will normally be assumed. However, special types of bracketing can be represented by codes of the forms given below.

~~!3E1:Q@(3)$~~ !3E1:Q@(3)$

!3E1:Q@(3$

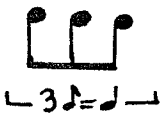
!3E1:Q@3($

!3E1:Q@<3>$

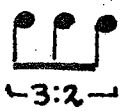
!3E1:Q@<3$

!3E1:Q@3<$

!3E1:Q@<3>¬$ (bracket is pushed to end of normal physical space for  .)



!3E1:Q@<3|EU|=|QU|>$



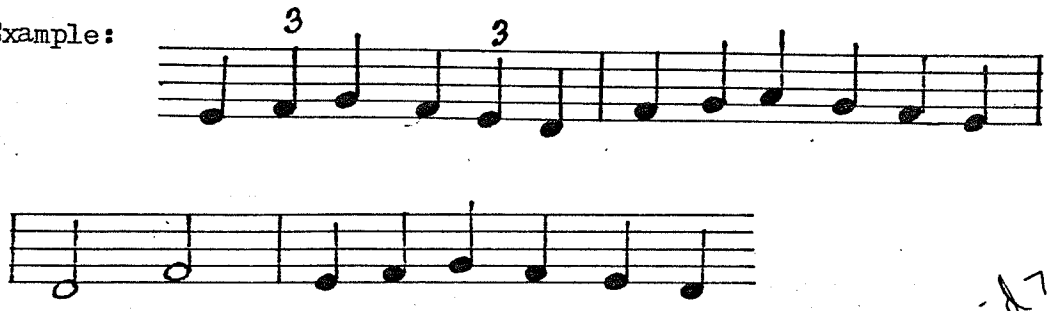
!3E1:Q@<3:2>$

etc.

If groupettes are present in a score but the groupette notation is suppressed, the point at which the groupette <u>notation</u> ceases may be indicated by a definer of the form

!mδ_1i:nδ_2j*

Example:



!3Q1:H 1Q1 2 3 2 1 0 / !3Q1:H* 2 3 4 3 2 1 /

OH 2 / 1Q1 2 3 2 1 0

[id?
(i)

Why not just '!(1*' ?
(just the identifier)

Ossia Control Codes

"Ossia" staves provide alternate versions of portions of the principal musical text of an Instrument and are positioned above or below the relevant portions of the principal text. To indicate ossia material and its disposition in a score a number of Ossia Control Codes, all of which begin with the two-character prefix !- , are available to the encoder.

Encoding Information on Ossia Staves.

The general form of the code to indicate the opening of ossia staves is

$$!-j^{\pm}\sigma$$

where j is an exact multiple of 50, such that $j/50$ is equal to the number of staves in the ossia brace,

+ or - indicates that the ossia brace being referenced is, respectively, above or below the principal stave defined by σ , and

σ designates the middle line of the principal stave to which the ossia text refers. In this context, therefore, σ must be an odd multiple of 25, that is, 25, 75, 125 . . . .

N.B. The default value for $\pm\sigma$ is +25.

Thus, !-50-75 represents the beginning of a single

ossia stave positioned below the second principal stave of the current Instrument; the coding that follows refers to that ossia stave until the ossia mode is cancelled.

Note that codes of this form permit the referencing of any ossia brace or stave, even if there be several such braces or staves simultaneously deployed above and below principal staves of an Instrument. Once a particular ossia brace has been referenced by such a code, the Space Codes 01-49 will refer to the topmost stave of that brace, 51-99 to the next lower, etc.

ossia C ldur
ldur C ossia
ldur = ossia ?

Because a principal text and its ossia variants provide a natural stratification of the information for a given Instrument, Linear Decomposition Mode provides natural and convenient procedures for encoding such material. Note that all abbreviations, in particular, "2-suppression" (B1) and Stave Transposition (F4) may be employed within the code for an ossia brace.

Ossia Mode itself is a separate stratum. Do not encode

Note, also, that a code of the form !+50 (a Stave Transposition Code) does not cancel a code of the type !-50 (Ossia Code). The two codes, although somewhat similar in appearance, are categorically distinct and unrelated.

Since ossia staves are normally size-reduced one degree, a Size Reduction Code (F3) will, in most cases, precede the code opening the ossia brace or stave, for example,

!'!-50 or !'!-100-75 .

This signifies that the ossia stave, as well as the symbols on it, will be size-reduced.

Ossia Mode Cancellation (temporary exit from ossia Mode, but a return to Ossia Mode will occur)

The Control Code

!-0±σ

where ±σ is as defined above, cancels the ossia mode; unless a new Instrument Code is stated, the information that follows will be assumed to pertain to the principal stave(s) of the active Instrument. [However, neither the suppression of the ossia stave or brace nor cancellation of size reduction (if any) is thereby implied.] within Ossia Mode only

Ossia Stave Suppression.

Once an ossia stave or brace is opened, it is assumed to continue throughout the score unless its suppression is specifically indicated by an Ossia Suppression Code of the form

!-±m;σ1,σ2,... $ → '$' needed because of commas in text

where m is the Space Code of the middle line of the principal stave (m = 25(2s-1) where s = 1 for the topmost stave, s = 2 for the next stave below, etc.) with which the ossia stave or brace is associated; and

σ1,σ2, ... specifies the Space Codes of the middle lines of the individual staves within the ossia brace that are to be suppressed.

Since such specificity will rarely be necessary, there are useful defaults of which to take advantage:

1. A code of the form

!-m$

will indicate suppression of <u>all</u> staves in the ossia brace attached to the principal stave defined by <u>m</u>.

2. The code

!-$

will indicate suppression of <u>all</u> ossia staves active in the Instrument. It is expected that this code will be the cancellation code most often employed.

Size Reduction and Enlargement

DARMS provides facilities for representing symbols that are size-reduced or -enlarged.

Size Reduction of one or two degrees may be established as a mode or may be specified for as small a unit of information as a single Attribute within the code for a Note.

A Size Reduction Mode is indicated by one of the following two codes, bounded on both sides by a Delimiter:

!'	Lesser reduction: for cues, ornamental configurations (except single grace-notes to Notes of normal size; see section D4);
!"	Greater reduction: for grace notes to cues.

The Size Reduction Mode thus established remains in force until

1. normal size is re-established by means of the code

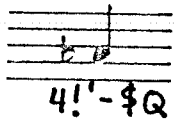
   !$ (preceded and followed by a Delimiter); or

2. the other Size Reduction Mode is established by a statement of the corresponding code.

Intra-token size reduction is made possible by imbedding, within the code for a Note, etc., the operators !' or !" in front of the code for the size-reduced element(s) and $ immediately following it. Note that no Delimiters separate the

Size Reduction Codes from the information to which it refers.

Example:



In the case of Attached Symbols (B7), the Size Code immediately follows the Attach Prefix. A Size Code may be prefixed to an Ossia Code, but not to an Instrument Code.

The size of symbols larger than normal is indicated by means of a Double-Space Code Prefix; see section B2 for further details. Note that a symbol encoded with a Double-Space Code Prefix is assumed to have normal proportions, and therefore is not enlarged only along its vertical axis.

If the Size Code is used to indicate an ossia stave (which is normally size-reduced), it is used as a prefix to the code establishing ossia mode. For the use of a Size Code in conjunction with a "stretched" symbol, see C3.6f.

Stave Transposition Code

The Stave Transposition Code facilitates the encoding of the second and subsequent staves of a multi-staved Instrument (e.g., piano) by reducing the given or implied Space Code by some multiple of 50. This can bring the Space Codes for the segment in question to the range 01-49, where, in turn, they may be further abbreviated by all the means described in section A1.

Specifically,

!+50

signifies that all subsequent Space Codes (not merely those for notes) are in actuality equal to 50 greater than their encoded value. This means that

!+50,!F 6E 9 30 9RE

is equivalent to

77!F 76E 79 80 79RE .

Taken over a larger span, the saving in characters made possible by the use of "2-suppression" (B1) is considerable.

This condition is terminated by either

1. a cancellation operator of the form

!+0

which simply cancels the Stave Transposition Mode, or

2. another Stave Transposition Code, for example,

!+100 (or !+150, etc.) specifying a different increment. Note that increments are always multiples of 50 and that they are absolute, not cumulative values; that is, only the most recently stated code is in force.

Doubling Mode

When notes are doubled at a fixed difference in Space Codes on the same stem (except for whole notes), an abbreviated form of representation may be used. The symbol

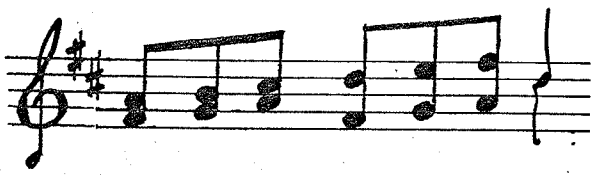
!|+n|

establishes the Doubling Mode at the arithmetic difference n between the Space Codes of the doubling note and the note from which it is referenced. Thus, given !|+7| all notes subsequently encoded (until otherwise indicated) will be assumed to be doubled at the (musical) interval of an octave above. Doubling Mode is cancelled by the symbol

!|+0|

or changed to another interval if a new doubling symbol is stated with a new value of n. The example below demonstrates the use of the doubling operator, which must be set off by a Delimiter on both sides.

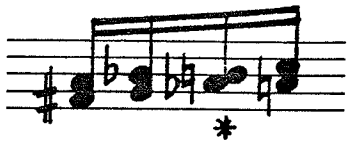
Example:



!G !K2# !|+2| (2 3 4) !|+5| (2 3 4) !|+0| 7QUD

Enharmonic Equivalents

Note that the interval being doubled must be notated so that the arithmetic difference between the Space Codes of the two notes is constant. If in a series of thirds there occurs



doubling at the third cannot be used for the interval marked with an asterisk.

Accidentals

Doubling Mode does not "assign" accidentals to the implied note except in the case of octaves and octave multiples (e.g., when the interval is represented by +7, +14, +21, etc.). This default assumption for octaves can be overridden by following n with an asterisk when establishing Doubling Mode, e.g.,

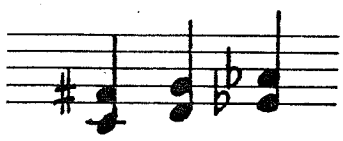
!|+7*|

Here and in the case of all non-octave intervals, the following rules pertain to the encoding of accidentals:

1. An accidental on the note that is doubled is given in the normal way;

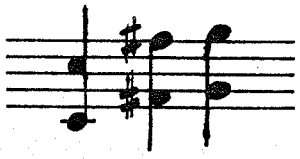
2. An accidental on the doubling note is stated as in a Chord Structure Reference, i.e., $\alpha\sigma$, is followed by a comma, and is placed directly in front of the encoding for the note of which it is the doubling.

Examples:

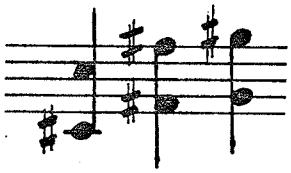


*

!|+3| #2,19Q 0 -4,1-



!|+7| 19Q 2# 3



!|+7*| 19#Q #9,2# #30,3

* B-E format more efficient

Equate Codes

Equate Codes simplify the encoding process by allowing a passage that recurs in identical or musically transposed form to be so designated, on its first appearance, by means of an Equate String Definer. Subsequent occurrences of the passage may then be represented by an Equate Code Reference.

The form of an Equate Code Definer is

$$=\underline{i},\underline{r}=\underline{s},\underline{l}=\$$$

where $\underline{i}$ is the Equate Code Identifier assigned by the encoder,

$\underline{s}$ is the DARMS string associated with $\underline{i}$, and

$\underline{r}$ is the number of occurrences in immediate succession of string $\underline{s}$. If $\underline{r} = 1$ then it and the preceding comma may be suppressed, thus producing the form $=\underline{i}=\underline{s}=\$$.

The minimal (and probably most frequently used) form of the Equate Code Reference is

$$=\underline{i}\$$$

which represents string $\underline{i}$. However, the general form of the Equate Code Reference is

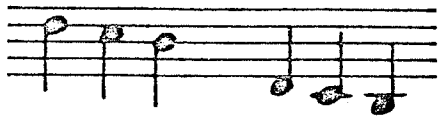
$$=\underline{i}*\pm_{\Delta},r\$$$

where $\underline{i}$ is the Equate Code Identifier,

* indicates that string $\underline{i}$ is to be transposed according to principles explained in the appendix to this chapter.

$\pm\Delta$ is the increment that is to be added or subtracted from all Space Codes in s if s recurs at one octave (Δ=7) or two (or more) octaves (Δ=14) above(+) or below(-) the defined disposition of string s; if Δ is zero, then the signed Δ may be suppressed.

Example:

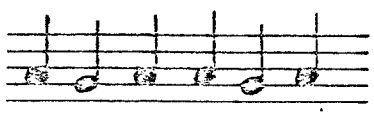


=4=7Q 6 5=$ ... =4-7$ ...

and r is the number of occurrences, in immediate succession, (as modified, if at all, by transposition operators in the Equate Code Reference) of s. If r is 1, then it and the preceding comma may be suppressed.

When the information represented by Equate String References is converted to canonical form, Equate String References will be replaced by the DARMS strings they represent before other abbreviations are expanded. It is therefore particularly important to consider carefully whether or not a Delimiter (D1) is required as the last character of the defined strings.

For example, given a repeated passage such as



the Equate String Definer

=5,2=4Q 3 4 =$

will first be expanded to

4Q 3 4 4Q 3 4

(with one trailing blank)

before other abbreviations are expanded. However, if the string is defined as

=5,2=4Q 3 4=$

then the unintended string

4Q 3 44Q 3 4

will be generated.

Sigma and Delta Propagation with respect to Equate Code

Because of ambiguity that might otherwise result, the following rules govern the suppression and forward propagation of Space Codes and Duration Codes at the beginning and end of a defined string:

1. There is no propagation of Space Code or Duration Code across the prefix (=i=) of an Equate String Definer, which means that in the defined string these codes cannot be suppressed for the first Note and that the Duration Code cannot be suppressed for the first Rest. Therefore, a DARMS string of the form

4Q 5 6H =5=Q 4E ... =$

is illegal because the Space Code for the first note in string 5 is lacking; it will not be assumed to be 6.

2. There is propagation of Space Code and Duration Code values across the =$ terminating the defined string. Therefore, in the case of

```
=5=4Q 5 6 7 =$ H
```

the Space Code of the last note in the defined string will be assigned to the first note to the right of the $, since the Space Code of that note has been suppressed. In other words, the terminating $ in Equate String Definers has no effect on the rules for Sigma Suppression (B1) and Delta Suppression (D4).

Matrix Definers and References

The form of these DARMS symbols is still to be announced.

cf. the opportunities for use
in Beethoven Op. 110

Comments

By means of a Comment Code of the form

Ks$,

where s is a string consisting of any combination of characters except the dollar sign,

comments that are not themselves part of the score being encoded may be inserted into a DARMS text.

Comment Codes may occur at any point within any DARMS Music Symbol or Control Code, thereby enabling the encoder to comment on any aspect of the musical text, its representation in DARMS or the DARMS control mechanisms employed. Such comments should be immediately appended to the codes for the feature or control being commented upon, without any Delimiter intervening. The system software will maintain the association of the comment and DARMS code immediately preceding. A comment may also appear as an independent DARMS token with a Delimiter on both sides; in this case the comment will not be associated with any specific piece of information.

Example:

DARMS text without comments: 6Q>

DARMS text with comments: 6KC# IN THE 1932 VERSION$QKDOTTED IN 1932 VERSION$>KSTACCATO IN 1932 VERSION$