INTERNATIONAL COMPUTERS AND TABULATORS LIMITED

Scientific Programming Department
ICT 1900 Series

Fortran Note 11
20.9.65.

FORTRAN IV LIST SYSTEM STRUCTURE

This note gives in detail the structure of all lists currently in use in The ICT FORTRAN IV Compiler.

The List System in the Compiler - Structure

The 'List System' of the compiler contains three types of list.

1. Local lists are produced and used by the compiler during compilation of a segment; they are lost when compilation of that segment ends.
2. A Consolidated Cue list is produced and used by the Consolidator and exists throughout the compilation of a complete program.
3. An Overlay list is produced during the Compilation of the Program. It is used by the Compiler during the compilation of each Program segment and therefore exists throughout the compilation of the complete program.

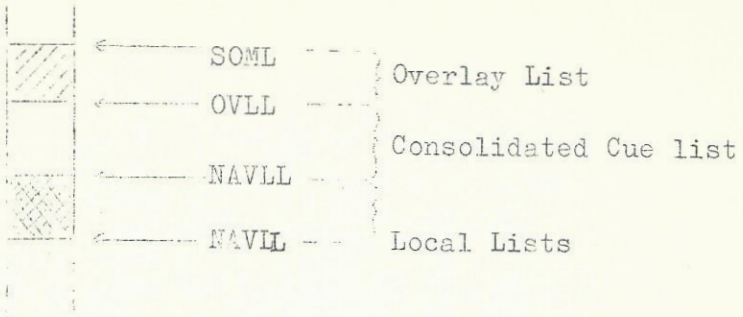
These Lists are stored in the top of store in the following order

1. Overlay List
2. Consolidated Cue List
3. Local Lists

In the compilation of a complete program, the Overlay List is the first list generated. The list is completed before any Consolidated Cue List is produced. The list is completed before any Consolidated Cue List is produced. The Overlay list is started at a location SOML. The Overlay List pointer OVLLL points to SOML at the start of the generation of the Overlay List and points to the word following the end of the final Overlay list item inserted by the time it is required to start forming the Consolidated Cue List. The Consolidated Cue List Pointer NAVLL is set to the value of OVLLL before the first item is generated for that list.

The Consolidated Cue list grows during the Compilation of a complete program, but does not grow during the list of the Local Lists for any segment. Because of this, the Local Lists for any segment can start at the current value of NAVLL (which points to the word following the end of the last item put into the Consolidated Cue List). The Local Lists Pointer is NAVL and for any segment this is initially set to the current value of NAVLL.

The layout of List Store during the compilation of a segment is shown in the following diagram.



Structure of the Individual Lists.

1. Overlay List.

This is a simple chained list which is addressed through the index word ΘVNDX

Each list item is divided into three parts as follows

Word 1. Control Word. This is a c/m to the previous item put into the list. 'c' is the length in words of the 'name' of the previous item and 'm' is the address of the Control Word im the previous item.

Word 2. Data Word

Rest of Item. 'Name' of Item.

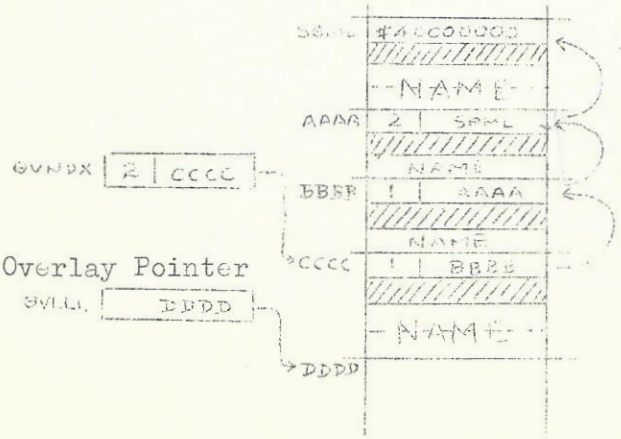
The Index Word is the Control Word for the next item to be put into the list.

The first item put into the list has a control word of #40000000

The Overlay List Pointer is ΘVLLL

The following diagram gives an example of the Overlay List Structure.

Overlay Index Word



2. Consolidated Cue List.

This is a simple chained list which is addressed through the index word QNDX. The Consolidated Cue List Pointer is NAVLL

This list is identical in form to the Overlay List except that the Control word for the first item put into the list is zero instead of # 40000000.

3. Local Lists.

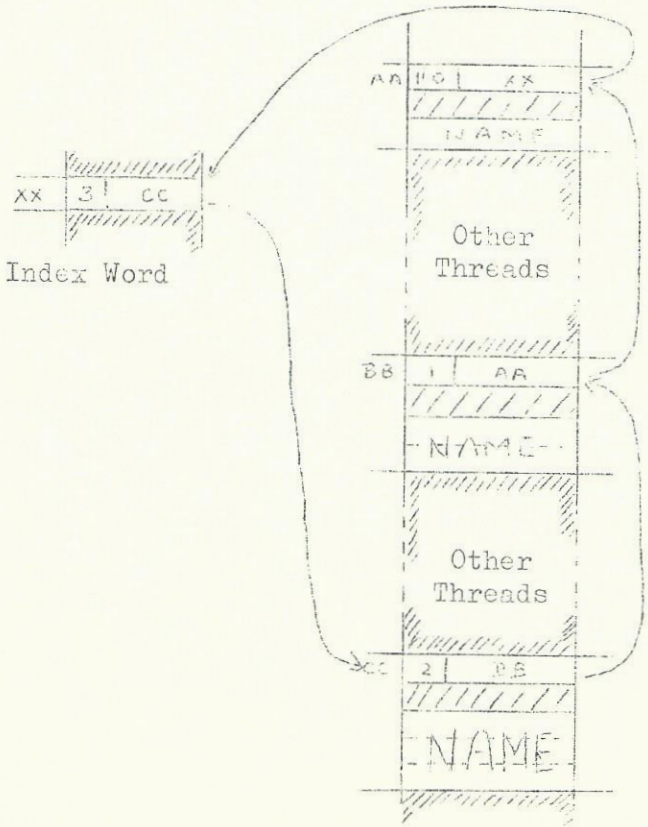
There are three different types of list used in the Compilation of a segment - Main List, Cue List and DØ list.

a). Main List

This is a multiple thread, simple chained list which is addressed through a series of index words (one per thread) starting at INDE. Each complete thread is identical in form to the Overlay List except that the Control Word for the first item put into a thread contains #400 in the counter and the address of the index word for the thread in the modifier.

Although there is one index word per thread there is only one Main List Pointer (NAVL). Thus the items in all threads are intermingled. This is of no consequence as a thread item is always addressed through the thread index word which is connected by address to all items on the same thread.

The following diagram gives an indication of the Thread Structure in the Main List.



b). Cue List.

This is a simple chained list which is addressed through the index word QFST. The structure of this list is slightly different to that of the other lists described so far.

Each list item is divided into two parts as follows.

Word 1. Control Word. This is a c/m to the next item in the list. If this is the last item put into the list then the control word is zero. 'c' is the length of the name of the next item in the list and 'm' is the address of the Control Word (Word 1) of the next item.

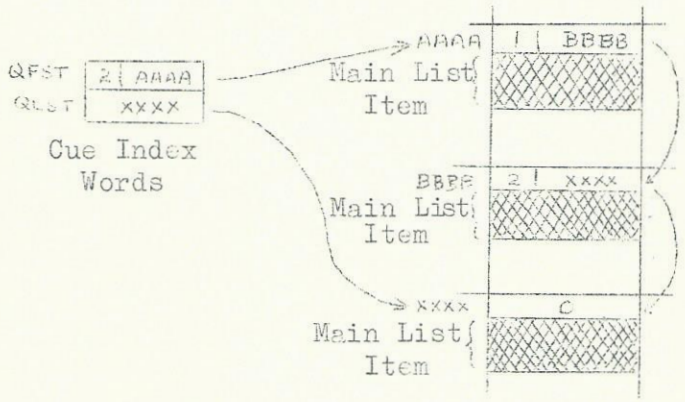
Rest of item. A complete Main List item.

The 'name' of a Cue List item is the same as the name of the Main List item it contains.

The Index Word QFST is the control word for the first item put into the Cue List.

An auxiliary index word QLST contains the address of the last item put into the Cue List, and is used for inserting a new list item.

Since a Cue List item contains a Main List item, after the generation of a Cue List item, NAVL points to the first word following that item. The following diagram gives an indication of the structure of the Cue List.



The Cue List is simply part of the Main List with an additional control word for each item. It is used to generate Cues during the production of the segment Leader. During segment compilation the compiler finds items by using the Main List Index words, not the Cue List index word.

c) DØ List.

The DØ List is divided into two independent lists called the Dead DØ list and the Live DØ list (DDØ list and LDØ list)

Each of these lists is a simple chained list containing 7 word list item, the first word being a control word and the remainder being data words for use by the DØ statement section of the Compiler.

The DØ lists are the only lists which may fluctuate in size during the compilation of a Segment. At the beginning of a DØ statement (or in an implied DØ of a READ or WRITE statement) an entry is made in the LDØ list. On completion of the DØ range this entry is removed from the LDØ list and transferred to the DDØ list. The basic LDØ list item was made in the first place by transferring the last DDØ item to the LDØ list. If the DDØ list was empty, the LDØ list item was made starting at the address specified by NAVL.

The Index Word for the DDØ list is VLST.

The Index Word for the LDØ list is DLST.

Both are initially zero.

The action of putting an item into the LDØ list and if necessary deleting an item from the DDØ list as follows.

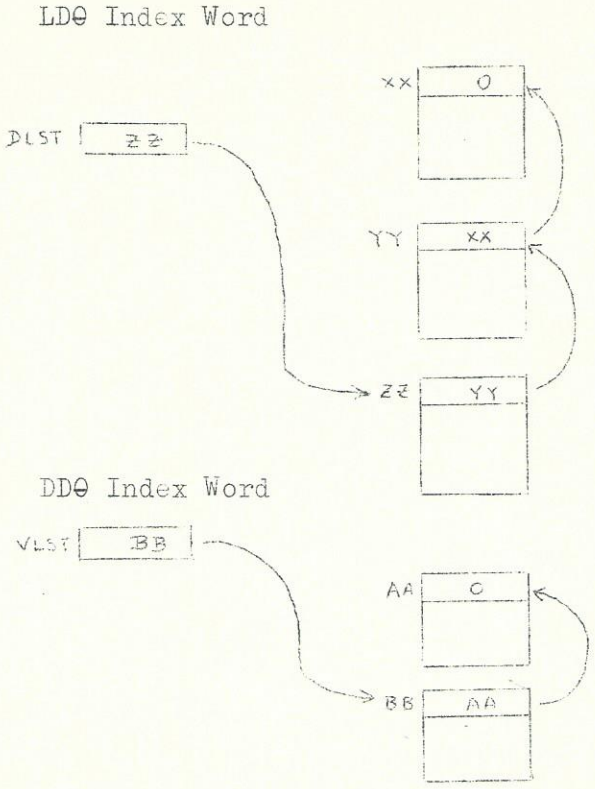
1. If VLST is zero, then the address specified by NAVL is set to the value of DLST and DLST is set to NAVL. NAVL is incremented by 7.
2. If VLST is not zero then the address specified by VLST is set to the value of DLST and DLST is set to VLST. VLST is set to the original contents of the address specified by VLST.

The action of putting an item into the DDØ list and deleting the last item put in the LDØ list is as follows.

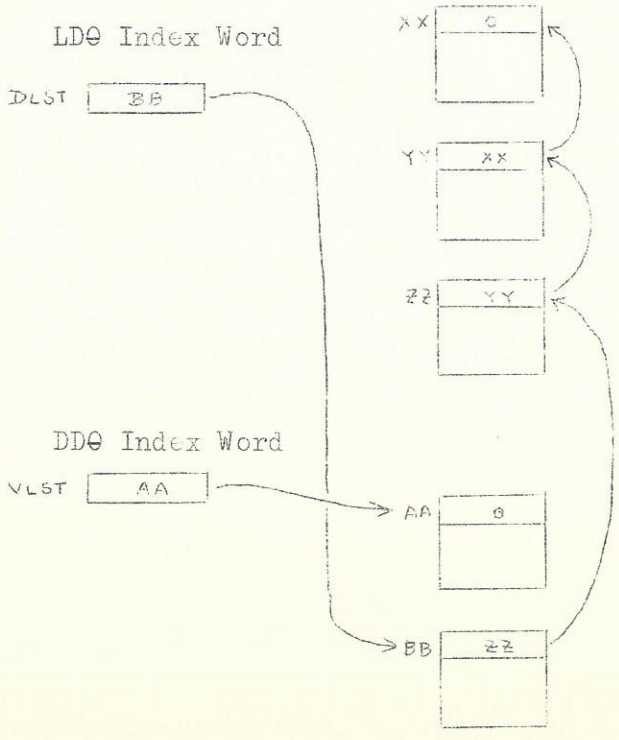
The Contents of VLST is stored in the first word of the list item to be deleted (specified by DLST) and the contents of that word having been saved. The contents of DLST replaces the contents of VLST, and the saved word replaces the contents of DLST.

To illustrate the generation of DDØ and LDØ items consider the examples on the following page.

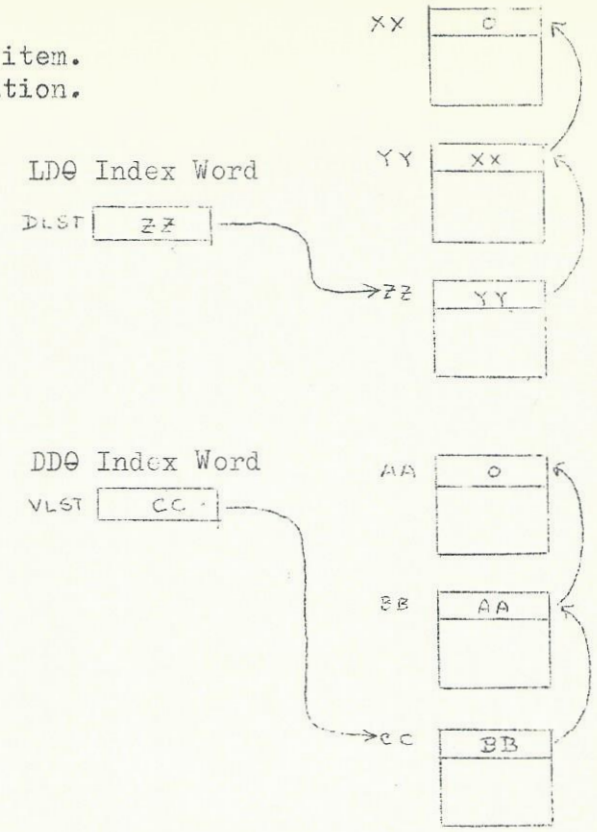
1. Adding a LDΘ item.
a) Original Situation



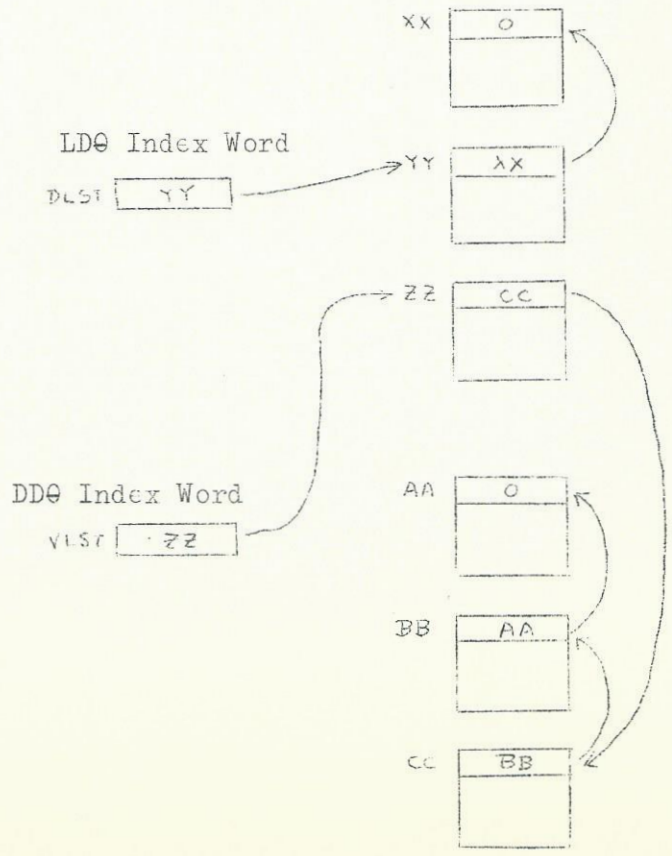
b). After inserting a LDΘ item (and hence Removing a DDΘ item).



2. Deleting a LDΘ item.
a). Original Situation.



b). After Deleting a LDΘ Item (and hence Inserting a DDΘ).



Local Lists in the Program Description

The Compilation of the Program Description may involve the generation of some Local and Overlay Lists at the same time. Because of this, the Local lists are not generated in the normal list area but are formed in the 'Polish' area of the compiler normally used for the analysis of arithmetic operations. At the start of the Program Description, NAVL is set to PNPI (the start of this area) which allows this area to be used for temporary list space.

V.K. Taylor