INTERNAL USE ONLY

INTERNATIONAL COMPUTERS AND TABULATORS LIMITED

Scientific Programming Department — FORTRAN NOTE 10

ICT 1900 Series — 31.8.65.

A Proposed Method of Describing and Implementing Program Overlays for the FORTRAN IV (ICT) Compiler.

This note supersedes FORTRAN NOTE 3 of 27.3.65.

A Proposed Method Describing and Implementing Program Overlays for the FORTRAN IV (ICT) Compiler.

General

If there exists a suitable "backing store" for the object program. There will have to exist a facility through which more than one segment of a program may be compiled to occupy the same main store locations, being read from the backing store to the main store when necessary.

The notes which follow give a proposal for program overlay facilities within the framework of FORTRAN IV (ICT) without adding to the FORTRAN Language.

For the purpose of this document the following terms are now defined.

1. Overlay Segment - A program segment which is obeyed from the same area of main store as another segment or segments and can therefore not co-exist in the main store with the other segment or segments.

2. Overlay Unit - A series of Overlay Segments which are always in the main store at the same time.

3. Overlay Area - An area of main store into which a number of Overlay Units may be loaded - At any one time an Overlay Area may contain only one Overlay Unit.

4. Overlay Area Number A small integer which 'labels' a particular Overlay Area.

5. Overlay Unit Number - A small integer which 'labels' a particular Overlay Unit within an Overlay Area.

FORTRAN Overlays

A FORTRAN IV (ICT) program may consist of a main section of program with no overlay segments; or it may contain a main section of program which may not be overlaid and a number of overlay segments in two or more Overlay Units which may be obeyed in one or more overlay areas. No one segment may be obeyed from more than one overlay area. At any time an overlay area contains only one overlay unit.

For FORTRAN IV (ICT) it is proposed that the only segments which may not be overlay segments are SUBROUTINE or FUNCTION segments which have more than 20 formal arguments, MASTER segments and routines which are integrally associated with the compiler (e.g. %FAP4, %FINOUT).

A subroutine or a function segment which has more than 20 formal arguments may not be called from an overlay unit

All Overlay Segments which belong to a particular Overlay Unit are described to the FORTRAN Compiler by a statement in the 'Program Description' of the form.

OVERLAY (a,u) Sub1,Sub2, -----Subk

where 'a' is the Overlay Area Number, and 'u' is the Overlay Unit number within that Area in which the 'Subi' are to co-exist. 'Subi' are the names of the routines which comprise the overlay unit.

(For the basic concept of the 'Program Description' see FORTRAN Note 2. 'A proposed Method of Describing a FORTRAN program to the FORTRAN IV (ICT) Compiler').

Program Communication with Overlay Segments.

Within FORTRAN segments, at the source program level, there is no indication whether a given segment is or is not an overlay segment. Thus, regardless of the 'status' of a segment the programmer will still write.

CALL Sub (Argument list)

to enter the segment 'Sub'; and from within that segment, the statement

RETURN

will cause the original segment to be re-entered at the statement following the CALL statement.

The actual communication between a segment and an overlay segment which is being called proceeds via a special overlay subroutine (%FOVL).

If in a routine 'A' a statement of the form CALL B ($a,---a_k$) is encountered, where 'B' has been defined previously in an OVERLAY statement, the following sequence is compiled.

and if A, B are not routines in the same overlay area, unit

CALL 1 1 %FOVL
X_B
X_A
N
Standard FORTRAN Calling Sequence for 'B'

where X_B indicates the Overlay Area and Overlay Unit for which 'B' is defined, and X_A gives the Area Unit for routine 'A'.

X_A and X_B are each one word long and have the following form

top 6 bits - zero
next 8 bits - Overlay Area (Binary Integer)
l.s.10 bits - Overlay Unit (Binary Integer)

'N' gives the number of words in the Standard FORTRAN Calling Sequence for 'B'. This will now be called the Overlay Calling Sub-sequence. 'N' may not be greater than 21.

If A,B are in the same overlay unit, only the standard FORTRAN calling sequence is compiled

The Operation of %FOVL

This routine ensures that the correct Overlay Unit is in store when a call for an Overlay segment is obeyed. It also keeps track of The Unit from which calls to Overlay Segments have been made so that on execution of a RETURN statement the information is available to ensure that the original calling segment is in store.

%FOVL is basically divided into two parts - that processing the Overlays CALL and that processing the RETURN.

%FOVL uses a push-down list to determine the 'level' of Overlays and to keep the original Overlay Unit and Area numbers and the return address for each Overlay Call. If the overlay call was made from 'permanent' program (non-overlay), then the Overlay Unit and Area numbers are both zero. This list is in common area %FOVC2.

a) Processing the Overlay Call

On entering %FOVL for an Overlay Call, the Overlay Calling Sub-sequence is moved to an area of %FOVL (called %FOVC1) so that the last word in the sub-sequence is stored in the last word of the area. In this way, after the subsequence has been obeyed from this area, a RETURN statement will always return to the same location in %FOVL. (see b) - Processing the RETURN).

After the sub-sequence has been moved, the actual return address in the calling segment is calculated and stored along with the Overlay Area and Unit number for the calling segment in the push-down list (the Overlay Stack).

At this stage %FOVL uses the 'Mark II Overlay Object Program Routine' (M2OPR) to ensure that the Overlay Unit containing the called routine is in store. On exit from M2OPR, the calling sub-sequence is obeyed from the area %FOVC1.

b) Processing the RETURN

All Overlay Calling Sub-sequences are obeyed from the %FOVC1 area such that the return from the called segment is to the same word in %FOVL.

On return to this word from the called segment, %FOVL ensures that the Overlay Unit and Area which is in the top item of the Overlay Stack is in store.(by use of M2OPR) and notes the 'return' address stored in that item. The top item in the stack is then removed and a branch to the return address is obeyed.

The Overlay Stack

The Overlay Stack is stored in the Area %FOVC2 such that the first item put in that stack is stored starting at the first word of the area. Each item which is put in the stack is two words long and is as shown below.

Word 1. Area/Unit for Calling routine.
Word 2. Return Address in Calling routine.

The 'Stack Pointer' (FOSP) initially contains the address of the start of the area %FOVC2. The action of putting an item into the Stack is to store the two words starting at the address given by FOSP and then incrementing FOSP by two.

The action of processing the top item in the stack is to decrement FOSP by two and then look at the item addressed by FOSP. The item is also effectively removed from the stack.

V. K. Taylor.