

August 28, 1957

Preliminary Report: Proposed Specifications for FORTRAN II for the 704

General

FORTRAN has now had some use at a large number of installations and has been used heavily for some time at at least six installations. A large number plan to use the system for the majority of their programming work. Difficulties in using FORTRAN have fallen into two categories (1) Difficulties arising in the initial period of putting the system into use, and (2) Difficulties arising from the properties of the system. Problems in the first category are being solved by on-the-spot expert assistance to those installations experiencing an undue amount of difficulty. The consensus of opinion concerning difficulties in category (2) is that the principal areas needing improvement are (a) facilities for debugging source programs and (b) facilities for creating and using subroutines. FORTRAN II will contain improved facilities in both these areas along the lines described below.

Debugging Facilities

The main improvement contemplated is the incorporation of a general and expandable diagnostic procedure in the translator. The purpose of this procedure is to find and to print a description of every detectable error in a source program. Thus stops in the translation will be eliminated or reduced to a minimum and ambiguity of the reason for a failure to translate will be progressively reduced. Error print-outs will often include values of parameters helpful in locating the trouble.

A variety of routines which are helpful in debugging object programs during execution can be added to the library of FORTRAN and included in an object program during translation by making use of the improved subroutine facilities described below.

Subroutine Facilities

A) Calling for subroutines

The FORTRAN II translator will accept an unlimited number of different statements. All statements presently in the system and a few others to be described below will be recognized and translated in the normal way. All others will refer to subroutines and must have the following form:

NAME (A, B, C, I, X)

where NAME denotes the name of the desired subroutine to be executed. Enclosed in

parentheses are the names of variables and/or arrays which the subroutine is to operate on and also the names of variables and/or arrays which are to receive the results produced by the subroutine. Variables may be fixed ~~or~~ floating point. After the subroutine has been executed the next executable statement in the FORTRAN program will be executed.

Examples

DPADD (A, B, C)

"DPADD" might be the name of a SAP - coded subroutine which adds the double-precision floating point number at A and A - 1 (in FORTRAN notation: A(1) and A(2)) to the number at B and B - 1 and puts the double precision result into C and C - 1. Thus in the FORTRAN program, A, B and C must be the names of vectors of length two or more. A(1) might contain the high order part and A(2) the low order part of the first operand. B(1) and B(2) would be the second operand and C(1) and C(2) the result.

INVMTX (X, N)

"INVMTX" may be the name of a subroutine compiled by FORTRAN which inverts a matrix whose DIMENSION entry is X(20, 20) and whose actual dimension is given by N and which replaces X by its inverse. The facilities for making a subroutine from a FORTRAN program are described below.

B) Use of subroutines, Binary Symbolic Subroutine Loader

All routines produced by FORTRAN II, both master routines and subroutines, will be loadable by the proposed Binary Symbolic Subroutine Loader. This BSS Loader will enable assembled programs in relocatable binary to retain symbolic references to subroutines at lower levels. As a result a routine and all its subroutines can be compiled or assembled independently. At execution time the relocatable binary decks of the main routine and all subroutines (all starting at relocatable 0) may be stacked in the card reader in any order, loaded by the BSS loader, and run.

The BSS Loader will load absolute and relocatable binary cards and accept the usual control cards plus one which will give the symbolic name of the routine and various other information. The first n words of a routine in relocatable binary form which makes symbolic reference to n other routines will contain the BCD symbolic names of the routines referred to. Reference to a routine B is made from routine A by transferring to that word of routine A which contains the BCD name of routine B. The BSS Loader will replace the symbolic name of B within the routine A by a transfer to the proper entry point in B after B has been assigned a location.

Since FORTRAN will compile routines which make symbolic reference to other routines, each routine in a program may be compiled or recompiled independently from the others. In fact, routines may be compiled which refer to subroutines which have yet to be written.

C) Generating a subroutine with FORTRAN II

Two new statements will be available. One of these is RETURN. The other is:

SUB DEF

meaning "subroutine definition". An example of their use is given by the following program.

```
SUB DEF ZERO 15 ( A, N)
DIMENSION A( 15, 15 )
DO 1 I = 1, N
DO 1 J = 1, N
1 A( I, J ) = 0.0
RETURN
```

Compilation of this program gives a relocatable binary deck of a subroutine which will clear the upper left hand $N \times N$ submatrix of any matrix X whose DIMENSION entry is X(15, 15). For example, if the following program is compiled and its binary deck is loaded together with the deck for ZERO 15 the result will be to clear matrices X, Y and Z:

```
DIMENSION X( 15, 15 ), Y( 15, 15 ), Z( 15, 15 )
ZERO 15 ( X, 15 )
ZERO 15 ( Y, 15 )
ZERO 15 ( Z, 15 )
STOP
```

The entire program following SUB DEF is assumed to be the subroutine. A subroutine may refer to as many other subroutines as desired and at any depth.