CONTROL DATA CORPORATION • DEVELOPMENT DIV • SOFTWARE DOCUMENT

DOCUMENT CLASS GED PAGE NO i

PRODUCT NAME FORTRAN Extended

PRODUCT NO. C012 VERSION 1.0 MACHINE SERIES 64/6600

DEPT NO 254 PROJECT NO 4P6X1 CHANGE NO 1 DATE 8 August 1966

SUBMITTED / REVIEWED / APPROVED

J F Clark 8/5/66 | 8-8-66 | M.D.B. 8/11/66 | RAZ for CEMiller 8-23-66

PROJ MGR DATE 8-8-66 | DEPT MGR DATE | DIR DATE | MAB DATE

Pierre P. Chin 8/9/66

DRB Chairman Date

R C Gunther 8/12/66

SPM Date

8/12/66

QA MGR. Date

C Conn 8/12/66

DOC. MGR. Date

RECEIVED
AUG 15 1966
INTERNAL DOCUMENT DISTRIBUTION

DISCLAIMER: THIS DOCUMENT IS A WORKING PAPER ONLY, AND DOES NOT NECESSARILY REPRESENT ANY OFFICIAL INTENT ON THE PART OF CONTROL DATA.

Formerly Product Number C010 - 64/6600 FORTRAN Version 3.0

1.0 General Description

The language implemented for this product will be ASA FORTRAN with extensions. The compiler will operate in conjunction with 64/6600 COMPASS Version 1.0 under 64/6600 SCOPE Version 3.0 operating system. Extensive explicit diagnostics will be produced to facilitate debugging. In addition, upon user request, information type diagnostics will be given pointing out non-ASA usage. The user will also have his choice of two degrees of optimization.

2.0 Source Language

2.1 Extensions to ASA FORTRAN as described in the ASA document X 3.9-1966. Use of these extensions may be made fatal to execution as an installation option.

2.1.1 Variable identifiers may be up to seven characters long.

2.1.2 An ENTRY statement permitting multiple entries to a subprogram.

2.1.3 DOUBLE may be substituted for DOUBLE PRECISION in the declarative statement.

2.1.4 Octal constants are indicated by a trailing B and may contain up to 20 octal digits.

2.1.5 .T. may be substituted for .TRUE., .F. for .FALSE., .A. for .AND., .O. for .OR. and .N. for .NOT. whenever these constants or operators may legally appear.

2.1.6 An element of an array may be referenced with fewer subscripts than dimensioned. This facility will operate the same as in 3600 FORTRAN.

2.1.7 Hollerith constants are permitted to occur in expressions and DATA statements. If the constant exceeds one word in size only the first word will be used as an operand when referenced in an expression.

2.1.8 The $ statement separator will be implemented as it is in 3600 FORTRAN.

2.1.9 The following facilities will be provided in the FORMAT statement, the first three of which conform to the 3600 FORTRAN implementation:

1. O for octal conversion
2. R conversion for alphanumerics
3. Asterisks to delimit Hollerith strings
4. T for column assignment in the form Tn where n specifies a column on the output or input record. This facility may be used to space forwards or backwards; if backwards spacing is used subsequent information will replace information which was already assigned to the following columns.

2.1.11 A two branch logical IF of the following form will be provided:

IF (L) n_1, n_2

where L is a logical expression and n_1 and n_2 are statement labels. If the value of the expression is .TRUE. control will be transferred to n_1, if .FALSE. statement n_2 will gain control.

2.1.12 A two branch arithmetic IF will be provided with the following form

IF (E) n_1, n_2

where E is an arithmetic expression. Control passes to n_1 or n_2 if the value of E is nonzero or zero, respectively.

2.1.13 Complex, double, real and integer operands may be mixed in an expression. Mode conversions are performed when an operation is encountered involving operands of different types. The hierachy is complex highest then, respectively, double, real and integer.

2.1.14 Any expression may be used as a subscript expression. Each expression in the subscript is evaluated as a normal arithmetic expression and then converted to integer type; the array element address is then evaluated using these values.

2.1.15 Masking replacement statements and operators are implemented as in 3600 FORTRAN.

2.1.16 The assigned GO TO statement may be used without the list required by ASA FORTRAN.

2.1.17 Implied DO notation may be used in the DATA statement, however, only one subscript may vary.

Example: (A(I,2), I=1,10)

2.1.18 The 3600 FORTRAN form of the DATA statement is permitted with the restriction as stated in 2.1.17.

Example: ((B(I,4), I=1,4,2) = 8.1, 9.1)

2.1.19 Short list notation, i.e. array name only, is permitted in the DATA statement.

Example: (C = 28 (1.4, 3.1))

2.1.20 A subroutine may return to any labeled statement in the calling subprogram. This facility is provided through the following three statement modifications:

1. To the CALL statement a list of statement labels to which the called subroutine may return may be appended.

   Example: CALL A (X,Y), RETURNS (15,24,73)

2. To the SUBROUTINE statement a list of variables of corresponding number is appended.

   Example: SUBROUTINE A (B,C), RETURNS (U,V,W)

3. To the RETURN statement may be appended any of the variable names which occur in the subroutine RETURNS list. On executing the RETURN statement control will be transferred to the statement label associated with the variable upon subroutine entry.

   Example: RETURN V would return control to statement label 24 in the calling subprogram.

2.1.21 Records stored on mass storage may be randomly referenced with the statement READ/WRITE MS (FNM (I), FMT) list; where FNM is the file name, I is the record ordinal within the file and FMT is an optional FORMAT statement label.

2.1.22 Common block may be declared to be stored in ECS (Extended Core Storage) by preceding the common block name with an asterisk in the COMMON statement.

Example: COMMON / *AB/ CAB (80,428)

2.1.23 The following statement will be provided to permit the transfer of data between central memory and ECS:

READ/WRITE ECS (A,B,N)

where A is the variable name which defines the central memory first word address, B is the variable name which defines the ECS first word address and N is the number of words to be transferred.

2.1.24 NAMELIST I/O facilities will be provided. This requires the incorporation of the NAMELIST declarative statement and the READ/WRITE (f,n) where f is the file name and n is the NAMELIST name. The implementation of this facility will correspond to that described in the IBM FORTRAN IV manual (File No. 7090-25 Form No. C28-6390-2, minor revision November 1965) with three exceptions:

1. The NAMELIST declarative statement must follow all specification statements.
2. Formal parameters may appear in the NAMELIST list.
3. The NAMELIST declarative statement is not required to precede a reference to it.

2.1.25 The BUFFER IN/OUT statement will be implemented as in 3600 FORTRAN except that a file referenced with this statement may only be referenced with this statement.

2.1.26 If the flow reaches the END statement in a procedure subprogram it will perform the function of a RETURN statement.

2.1.27 READ, PRINT and PUNCH statements are permitted.

2.1.28 The use of a $ or * in place of the C on a comment card will be permitted.

2.1.29 ENCODE/DECODE WILL BE IMPLEMENTED AS IN 3600 FORTRAN

2.2 Incompatabilities with 64/6600 FORTRAN Version 2.0

2.2.1 The prefixes FORTRAN II, FORTRAN IV and FORTRAN VI may not be attached to subprogram header cards since only one language will be processed by the compiler.

2.2.2 Assembly language subprograms which are intermixed with FORTRAN subprograms will be identified by an IDENT card rather than an ASCENT card.

2.2.3 Mixed mode arithmetic will not include logical operands as no numerical value will be associated with them.

2.2.4 In evaluation of an expression, mode conversion will take place only when an operation is to be performed involving operands of different types. The operand of lower type is converted to the type of the higher; not necessarily the highest in the expression.

2.2.5 A false relational expression is assigned some positive value; a true relational is assigned some negative nonzero value. This permits faster evaluation of relationals. Version 2.0 uses plus or minus zero to indicate false or true relationships.

2.2.6 All I/O status checking will be done through library functions, e.g. GO TO (1,2), EOF (I) rather than IF(ENDFILE, I) 1,2.

2.2.7 All specification statements must precede the first DATA statement, statement function or executable statement.

2.2.8 0 type of octal constants will not be provided.

2.2.9 In evaluating logical expressions the entire statement is not necessarily evaluated e .g. for A .AND. FUN (B), if A is false FUN of B will not be evaluated.

2.2.10 READ/WRITE INPUT/OUTPUT TAPE will not be implemented.

2.2.11 Using an S trailing a statement number in an actual parameter list will not be permitted.

2.3 Compatability with other FORTRANS

FORTRAN Extended Version 1.0 should achieve a high degree of compatability with other FORTRAN systems by conforming to the ASA standard. Most incompatabilities which occur will be brought about by differences in word length and in I/O. Compatability with the standard 3000 series FORTRANs will be quite high as most of their facilities have been incorporated into the system; it should be noted, however, that all DO loops will be executed at least once.

2.4 User Options

The following options will be provided by the FORTRAN system:

1. Choice of two degrees of object code optimization. The lower degree will provide faster compile time at the expense of object code speed and compactness. The higher degree will produce reasonably compact code with significantly improved execution speed.
2. Diagnostics pointing out non-ASA usages in the source program will be available.
3. A listing of the source statements making up each subprogram will be available.
4. COMPASS subprograms may always be intermixed with FORTRAN subprograms.
5. The user may specify the location of all input and output files at compile time.
6. Cross Reference listing of the assembled code will be available.

3.0 Compiler Output

The compiler will be capable of producing a source listing, a listing of diagnostics, and assembly code in proper format for assembly by 64/6600 COMPASS.

4.0 Object Program Characteristics

4.1 Optimizations

The following is a list of optimizations which will be carried out if the user selects the standard degree of optimization, although those followed by an asterisk will be performed if the lesser degree of optimization is selected.

4.1.1 Elimination of redundant operations where possible.

4.1.2 Evaluation of array element address by the index function method.*

4.1.3 Critical path analysis of instruction sequences to maximize parallel operation.

4.1.4 Reformation of subexpressions to permit extended parallelism.*

4.1.5 Evaluation of constant subexpressions at compile time.*

4.1.6 Determination for each reference to a formal parameter at compile time as to whether it should be referenced by address substitution indirect addressing.*

4.1.7 Elimination of common remote parameter lists.

4.1.8 It will be noted whether or not an innermost DO loop fits in the stack and the resultant code will reflect this change in timing.

4.1.9 Simple constants will be formed by sets rather than loads.*

4.1.10 Branches to the next instruction will be eliminated.*

4.1.11 Presetting of arrays with a constant pattern will be specially handled at load time to avoid the generation of a large binary deck.*

4.1.12 Inline evaluation of some functions.*

4.1.13 Branch evaluation of relationals (see 2.2.10)*

4.2 Communication between FORTRAN - Produced and Assembly-Produced Subprograms

Communication between COMPASS and FORTRAN subprograms will be easily accomplished either through COMMON or by utilizing the normal FORTRAN linking mechanism. The use of the linking mechanism will be quite simple since COMPASS macros will be

available to form the required lists.

5.0 Compiler Characteristics

5.1 Language

The compiler will be written in COMPASS.

5.2 Structure

FORTRAN Extended Version 1.0 will be a two pass compiler which produces assembly code. The passes are described below.

5.2.1. Pass 1 - Pass 1 consists of two overlays, called phases, and a controlling overlay which is loaded by the main control program. It contains the FORMAT processor, statement scanner, diagnostic processor and the list manipulating routine. It initially calls in Phase 1 to handle all declarative statements and produces assembly code for them when the first executable statement is encountered. Phase 2 is then loaded in and processes all of the remaining statements producing a register free notation called R-list.

5.2.2 Pass 2 - Pass 2 issues the diagnostics and then processes the R-list and generates the remaining assembly code associated with the subprogram.

6.0 Software and Hardware Interfaces

6.1 Software

The final system will require the 64/6600 SCOPE Version 3.0 operating system and the 64/6600 COMPASS Version 1.0 assembler. During the development of the system, use will be made of the available versions of the operating system, assembler and compiler. The portion of the compiler dealing with random access to the disc and ECS cannot be checked out until the facilities are available through the target operating system.

6.2 Hardware

The minimum machine required will be the same as that required by 64/6600 SCOPE Version 3.0.

7.0 Timing Estimate

Due to the orientiation of the compiler toward producing efficient object code it is difficult to estimate compiling rates though the figure of 10,000 cards per minute seems attainable.