DEL

CONTROL DATA
CORPORATION

CONTROL DATA®
CYBER 70
COMPUTER SYSTEMS
MODELS 72, 73, 74, 76
7600 COMPUTER SYSTEM
6000 COMPUTER SYSTEMS

FORTRAN EXTENDED INSTANT
MODELS 72, 73, 74, VERSION 4
MODEL 76 VERSION 2
7600 VERSION 2
6000 VERSION 4

DEL 14.1.72

CONTROL DATA
CORPORATION

CONTROL DATA®
CYBER 70
COMPUTER SYSTEMS
MODELS 72, 73, 74, 76
7600 COMPUTER SYSTEM
6000 COMPUTER SYSTEMS

FORTRAN EXTENDED INSTANT
MODELS 72, 73, 74, VERSION 4
MODEL 76 VERSION 2
7600 VERSION 2
6000 VERSION 4

New features, as well as changes, deletions, and additions to information in this manual are indicated by bars in the margins or by a dot near the page number if the entire page is affected. A bar by the page number indicates pagination rather than content has changed.

REVISION RECORD	
REVISION	**DESCRIPTION**
A	Original printing.
(11-24-71)	
Publication No. 60357900	

Additional copies of this manual may be obtained from the nearest Control Data Corporation sales office.

Address comments concerning this manual to:

CONTROL DATA CORPORATION
Software Documentation
215 MOFFETT PARK DRIVE
SUNNYVALE, CALIFORNIA 94086

INTRODUCTION

This instant outlines the FORTRAN Extended language (version 4.0) for the CONTROL DATA® CYBER 70/Models 72, 73 and 74, and 6200, 6400, 6500, 6600 and 6700 computers, and FORTRAN Extended (version 2.0) for the CONTROL DATA CYBER 70/Model 76, 7600, 7601-1 and 761X computers. Detailed information is contained in the FORTRAN Extended Reference Manual Publication No. 60305600 A. FORTRAN Extended is designed to comply with American National Standards Institute FORTRAN language.

The FORTRAN compiler operates in conjunction with version 3.0 COMPASS assembly language processor under control of the 6000 SCOPE operating system (version 3.4) and 7000 SCOPE operating system (version 2.0).

The following new features are included in FORTRAN Extended:

- LEVEL statement
- IMPLICIT statement
- Hollerith strings in output lists
- Expressions in output lists
- Quote delimited Hollerith strings
- Exclusive OR function
- Messages on STOP and PAUSE statements
- Line limit on output file at execution time
- Syntax scan only during compilation
- Program listings suppressed but reference map produced
- Rewrite in place, mass storage
- Multiple systems texts and local texts for intermixed COMPASS programs

Throughout this document, CONTROL DATA extensions to the FORTRAN language are indicated by blue type. Otherwise, FORTRAN Extended conforms to ANSI standards.

Information which applies only to the CONTROL DATA CYBER 70/Model 76 and 7600 computers is indicated by red type.

Information which applies only to the CONTROL DATA CYBER 70/Models 72, 73 and 74, and 6000 Series computers is indicated by green type.

LANGUAGE ELEMENTS

SYMBOLIC NAMES

Symbolic names are 1–7 alphanumeric characters; the first must be alphabetic.

FORTRAN CHARACTER SET

Alphabetic: A to Z
Numeric: 0 to 9
} alphanumeric

Special:	=	equal	)	right parenthesis
	+	plus	,	comma
	–	minus	.	decimal point
	*	asterisk	$	dollar sign
	/	slash		blank
	(	left parenthesis	≠ or '	quote

Any character of the SCOPE set may be used in Hollerith information and comments. Blanks are significant only in Hollerith fields.

FORTRAN STATEMENTS

Column 1	C or $ or * indicates comment line
1–2	C$ indicates DEBUG statement
1–5	Statement label
6	Any character other than blank or zero denotes continuation, except on comment cards. A DEBUG continuation card must contain C$ in columns 1 and 2.
7–72	Statement
73–80	Identification field, not processed by compiler

Statements may be labeled by an integer constant in the range 1–99999. If a C, $ or * appears in column 1, the remainder of the card is ignored by the compiler, but printed with the source listing as a comment.

$ may be used to separate multiple statements on a card with all statements except FORMAT, OVERLAY or debugging statements.

A character other than zero or blank in column 6 signifies continuation from the preceding card.

Statements are written in columns 7–72; blanks are ignored except in Hollerith fields.

Columns 73–80 may contain identification and serial numbers which are ignored by the compiler but printed with the program listing.

All 80 columns may be used for data input.

CONSTANTS

Constants	Form	Examples
Integer	$n_1 n_2 \ldots n_m$	2
	$1 \leq m \leq 18$ Range: $-(2^{48}-1)$ to $2^{48}-1$	247
	Integer addition and subtraction results may range from $-(2^{59}-1)$ to $2^{59}-1$. Integers used as a DO index or as a subscript must be in the range $2^{17}-2$.	31456932
Real	n.n .n n. n.nE±s .nE±s n.E±s nE±s	
	n Coefficient $\leq$ 15 decimal digits	7.5 3.22 42.E1
	E±s Exponent	314.E05 700.E–2
	s Base 10 scale factor	.5 0.
	Range 10^{-293} to 10^{+322}	
	Accurate to approximately 15 decimal digits	

Constants	Form	Examples
Double Precision	n.nD±s .nD±s n.D±s nD±s	
	n Coefficient ≤ 29 decimal digits	5.834D2 7.D2
	D±s Exponent	9.2D03
	s Base 10 scale factor	14.D−5
	Range 10^{-293} to 10^{+322}	3120D4 1.D0
	Accurate to approximately 29 decimal digits	
Complex	(r1,r2)	(1.,7.54)
	r1 Real part	(−2.1E1,3.24)
	r2 Imaginary part	(0.,−1.)
	Each part has same range as a real constant	(4.0,5.0)
Octal	$n_1 \ldots n_m$B	7777777777B
	1≤m≤20	525252B
Hollerith	nHf	
	nRf	
	nLf	
	≠f≠	
	1≤n≤10 in expression	
	1≤n≤150 in FORMAT statement	
	H left justified with blank fill	6HABCDEF
	R right justified with binary zero fill	7RJUSTIFY
	L left justified with binary zero fill	7LTHE END

Constants	Form	Examples
	A Hollerith string delimited by paired symbols ≠ ≠ can be used anywhere the H form of the Hollerith constant can be used. For example: IF(V.EQ.≠YES≠) GO TO 20 PRINT 1,≠ SQRT = ≠ ,SQRT(.5) 1 FORMAT (A10,F10.2)	≠ABCDEF≠
Logical	.TRUE. or .T. stored as minus one .FALSE. or .F. stored as all zero bits (+0)	LOGICAL X1,Z2 X1 = .TRUE. Z2 = .FALSE.

VARIABLES

1—7 alphanumeric characters; the first must be alphabetic.

A variable not defined in a type declaration is real if the first character of the symbolic name is any letter other than I,J,K,L,M,N and if no IMPLICIT statement appears in that program unit.

Implied Typing of Variables

A—H, O—Z	Real
I—N	Integer

Variables	Form	Examples
Integer	Range $-(2^{59}-1)$ to $2^{59}-1$. As subscript or index of a DO statement, maximum value is $2^{17}-2$. As a result of multiplication or division or conversion from real to integer, or integer to real, maximum value is $2^{48}-1$.	ITEM JSUM KOOL INTEGER X

Variables	Form	Examples
Real	Range 10^{-293} to 10^{+322}, approximately 15 significant digits.	AVAR SUM TUF BETA REAL I
Double Precision	Must be defined explicitly in type declaration. Range 10^{-293} to 10^{+322}, approximately 29 significant digits. Occupies two words in storage.	DOUBLE PRECISION *OMEGA,X,B DOUBLE X,Y
Complex	Must be defined explicitly in type declaration. Occupies two words in storage; each word contains a number in real variable format and each number can range from 10^{-293} to 10^{+322}	COMPLEX A,D COMPLEX P2
Logical	Must be defined explicitly in type declaration. A logical variable with positive zero value is false. A logical variable with value minus one is true.	LOGICAL L3,C LOGICAL L2,R

ARRAYS

An array name may have up to three subscripts. Subscripts may be any valid arithmetic expressions; zero and negative subscripts are not allowed. A non-integer subscript value is truncated to integer. If the number of subscripts in a reference is less than the declared dimensions of the array, the compiler assumes a value of one for missing subscripts.

Examples:

(I,J) (R+3.5*A,B) (I+3,J+3,2*K+1)

To find the location of an element in the linear sequence of storage locations:

Number of Dimensions	Array Dimension	Subscript	Location of Element Relative to Starting Location
1	ALPHA(K)	ALPHA(k)	(k-1)xE
2	ALPHA(K,M)	ALPHA(k,m)	(k-1+Kx(m-1))xE
3	ALPHA(K,M,N)	ALPHA(k,m,n)	(k-1+Kx(m-1+Mx (n-1)))xE

K, M, and N are dimensions of the array.

k, m, and n are actual subscript values of the array.

1 is subtracted from each subscript value because the subscript starts with 1, not 0.

E is length of the element. For real, logical, and integer arrays, E = 1. For complex and double precision arrays, E = 2.

STATEMENT FORMS

The following symbols are used in the descriptions of FORTRAN Extended statements:

v	variable or array element
sn	statement label
iv	integer variable
name	symbolic name
u	input/output unit: 1- or 2-digit decimal integer constant integer variable with value of: 1–99 or display code file name
fn	format designator
iolist	input/output list

Other forms are defined individually in the following list of statements.

Assignment Statements

Form	Examples
v = arithmetic expression	A = B + C
logical v=logical or relational expression	LOGICAL L,M,N L = M .AND.N
v = masking expression	CAT = 5252B .OR. Z
MULTIPLE ASSIGNMENT	
$v_1 = v_2 = \ldots v_n$ = expression	X = Y = Z = (10 + B)/SUM(1)

Control Statements

GO TO sn	GO TO 30
GO TO ($sn_1, \ldots, sn_m$) , iv	GO TO (1,4,7,2), N
GO TO ($sn_1, \ldots, sn_m$) iv	GO TO (3,6,10,1) J
GO TO ($sn_1, \ldots, sn_m$) , expression	GO TO (1,2,9,4), A + B

Form	Examples
GO TO (sn_1, . . . , sn_m) expression	GO TO (3,4,5,6) N + J
GO TO iv,(sn_1, . . . , sn_m)	GO TO LSWITCH, (10,20,30,40)
GO TO iv (sn_1, . . . , sn_m)	GO TO NEXT (1,2,3,4)
ASSIGN sn TO iv	ASSIGN 10 TO LSWITCH
IF (arithmetic or masking expression) sn_1,sn_2,sn_3	IF (I—N) 3,4,6
IF (arithmetic or masking expression) sn_1,sn_2	IF (I*Y*K) 100, 200
IF (logical or relational expression) stat	IF (P.AND.Q) RES = 7.2
IF (logical or relational expression) sn_1 ,sn_2	IF (K.EQ. 100) 60,70
DO sn iv = m_1 ,m_2,m_3	DO 100 I = 1,10,2
DO sn iv = m_1,m_2	DO 2 J = 1,5
sn CONTINUE	100 CONTINUE
PAUSE	PAUSE
PAUSE n	PAUSE 2
PAUSE ≠c . . . c≠	PAUSE ≠ CHANGE TAPE ≠
STOP	STOP
STOP n	STOP 25
STOP ≠c . . . c≠	STOP ≠ END OF RUN ≠
END	END

n — string of 1—5 octal digits

c . . . c — string of 1—70 characters

Type Declaration

Type specifications can be dimensioned.

Form	Examples
INTEGER $name_1, \ldots, name_n$	INTEGER A,B,C(10)
TYPE INTEGER $name_1, \ldots, name_n$	TYPE INTEGER X,Y,N
REAL $name_1, \ldots, name_n$	REAL NEXT,X(5)
TYPE REAL $name_1, \ldots, name_n$	TYPE REAL N,J,CAT
COMPLEX $name_1, \ldots, name_n$	COMPLEX CC,J
TYPE COMPLEX $name_1, \ldots, name_n$	TYPE COMPLEX NON,Z(3)
DOUBLE PRECISION $name_1, \ldots, name_n$	DOUBLE PRECISION DP1,DP2
DOUBLE $name_1, \ldots, name_n$	DOUBLE DP3
TYPE DOUBLE PRECISION $name_1, \ldots, name_n$	TYPE DOUBLE PRECISION CAT,DOG
TYPE DOUBLE $name_1, \ldots, name_n$	TYPE DOUBLE HEN,DUCK
LOGICAL $name_1, \ldots, name_n$	LOGICAL L1,L2
TYPE LOGICAL $name_1, \ldots, name_n$	TYPE LOGICAL LL,LN
IMPLICIT $type_1(ac), \ldots, type_n(ac)$	IMPLICIT REAL (I–N)

(ac) is a single alphabetic character or range of characters in alphabetic sequence represented by the first and last character separated by a minus sign.

External Declaration

EXTERNAL $name_1, \ldots, name_n$	EXTERNAL ABS

Form	Examples
type $name_1$ (d_1)	
TYPE type $name_1$ (d_1)	
DIMENSION $name_1$ (d_1) , . . . ,$name_n$ (d_n)	DIMENSION SUM (10)

d_i array declarator, one to three integer constants; or in a subprogram, one to three integer variables

type INTEGER, REAL, COMPLEX, DOUBLE PRECISION or LOGICAL

COMMON v_1, . . . ,v_n	COMMON A,B,C
COMMON/$blkname_1$/v_1, . . . ,v_n. . . /$blkname_n$/v_1, . . . ,v_n	COMMON/BLK/D,E,F/CAT/X,Y,Z(10)
COMMON// v_1, . . . ,v_n	COMMON//NEXT,JAY(3)

blkname symbolic name or 1–7 digits

// blank common

DATA $vlist_1$/$dlist_1$/, . . . ,$vlist_n$/$dlist_n$/	DATA A,B,C/3.,27.5,5.0/
DATA (var=dlist) , . . . , (var=dlist)	DATA (X=3.),(Y=5.)

var variable, array element, array name or implied DO list

vlist list of array names, array elements, or variable names, separated by commas

dlist one or more of the following forms separated by commas:

constant
(constant list)
rf*constant
rf*(constant list)
rf(constant list)

constant list list of constants separated by commas

rf integer constant. The constant or constant list is repeated the number of times indicated by rf

Form	Examples
EQUIVALENCE (v_1, . . . ,v_n), . . . ,(v_1, . . . ,v_n)	EQUIVALENCE (N,J),(X,Y)
LEVEL n, a_1, . . . ,a_n n unsigned integer 1, 2 or 3 a variable or array name	LEVEL 3,X,Y,Z(3)

Main Programs

PROGRAM name ($file_1$, . . . ,$file_n$)	PROGRAM A(INPUT,OUTPUT)
PROGRAM name	PROGRAM B

Subprograms

FUNCTION name (p_1, . . . ,p_n)	FUNCTION GRATER(A,B)
type FUNCTION name (p_1, . . . ,p_n)	REAL FUNCTION D(X,Y)
type INTEGER, REAL, COMPLEX, DOUBLE PRECISION or LOGICAL	
SUBROUTINE name (p_1, . . . ,p_n)	SUBROUTINE X(C,D,E)
SUBROUTINE name	SUBROUTINE PGM

SUBROUTINE name (p_1, . . . ,p_n), RETURNS (b_1, . . . ,b_m)

SUBROUTINE SUB(X,Y), RETURNS (M,N)

SUBROUTINE name,RETURNS (b_1, . . . ,b_m)

SUBROUTINE SUB2,RETURNS(J,K,L)

Entry Point

ENTRY name	ENTRY BOX

Statement Functions

name (p_1, . . . ,p_n) = expression	ADD(X,Y,C,D) = X+Y+C+D

Subprogram Control Statements

Form	Examples
CALL name	CALL JIM
CALL name ($p_1, \ldots, p_n$)	CALL JIM (A,B)
CALL name ($p_1, \ldots, p_n$),RETURNS ($b_1, \ldots, b_m$)	CALL JOHN (X,Y),RETURNS (N,K)
CALL name,RETURNS ($b_1, \ldots, b_m$)	CALL SUB4,RETURNS (J,K,I)
RETURN	RETURN
RETURN i	RETURN M
i a dummy argument in a RETURNS list	

Specification Subprograms

BLOCK DATA	BLOCK DATA
BLOCK DATA name	BLOCK DATA BD3

Input/Output

PRINT fn,iolist	PRINT 4, A,B,N
PRINT fn	PRINT 20
PUNCH fn,iolist	PUNCH 2, X,Y,Z
PUNCH fn	PUNCH 30
WRITE (u,fn) iolist	WRITE (4,27) X,Y,Z
WRITE (u,fn)	WRITE (2,30)
WRITE (u) iolist	WRITE (3) A,B,C
WRITE (u)	WRITE (6)
READ fn,iolist	READ 100, A,B,C

Form	Examples
READ fn	READ 50
READ (u,fn) iolist	READ (5,100) X,Y,Z
READ (u,fn)	READ (5,100)
READ (u) iolist	READ (3) JN,AB
READ (u)	READ (5)
BUFFER IN (u,p) (a,b)	BUFFER IN (1,1)(R(1),R(512))
BUFFER OUT(u,p) (a,b)	BUFFER OUT(1,J) (B(M),B(N))

a — first word of data block to be transferred

b — last word of data block to be transferred

p — integer constant or integer variable
zero = even parity, one = odd parity

NAMELIST/group $name_1/a_1, \ldots, a_n/ \ldots /group\ name_n /a_1, \ldots, a_n$

	NAMELIST/SHIP/I1,I2,A,B
READ (u,group name)	READ (5,SHIP)
WRITE (u,group name)	WRITE (6,SHIP)
READ group name	READ SHIP
PRINT group name	PRINT SHIP
PUNCH group name	PUNCH SHIP

a_i — array names or variables

group name — symbolic name identifying the group $a_1, \ldots, a_n$

Internal Transfer of Data

ENCODE (c,fn,v) iolist	ENCODE (40,1,ALPHA) A,B,C
DECODE (c,fn,v) iolist	DECODE (77,17,CARD) INK

v — starting location of record; variable or array name

c — length of record in characters; unsigned integer constant or simple integer variable

File Manipulation

Form	Examples
REWIND u	REWIND 3
BACKSPACE u	BACKSPACE LUN
ENDFILE u	ENDFILE 4

Format Specification

sn FORMAT ($fs_1, \ldots, fs_n$) 100 FORMAT (I6,F7.3,2I4)

fs_i one or more field specifications separated by commas and/or grouped by parentheses

Data Conversion

srEw.d	Single precision floating point with exponent	2E13.3
srFw.d	Single precision floating point without exponent	F7.3
srGw.d	Single precision floating point with or without exponent	G14.6
srDw.d	Double precision floating point with exponent	2D10.4
rIw	Decimal integer conversion	4I9
rLw	Logical conversion	2L5
rAw	Alphanumeric conversion	A7
rRw	Alphanumeric conversion	4R10
rOw	Octal integer conversion	O5

s optional scale factor of the form:

nPDw.d	2PD18.7
nPEw.d	3PE20.2
nPFw.d	-1PF13.6
nPGw.d	1PG16.2
nP	0P

Form		Examples
	r repetition factor	
	w integer constant indicating field width	
	d integer constant indicating digits to right of decimal point	
nX	Intraline spacing	9X
nH . . . * . . . * ≠ . . . ≠	Hollerith	8H THE END *FINIS* ≠ TEST 7 ≠
/	Format field separator; indicates end of FORTRAN record	
Tn	Column tabulation	T10

OVERLAYS AND DEBUGGING STATEMENTS

Overlays

Form	Examples
CALL OVERLAY (fname,i,j,recall,k)	CALL OVERLAY (4HTEST,1,0,6HRECALL)

i — primary overlay number

j — secondary overlay number

recall — if 6HRECALL is specified, an overlay already in memory is not reloaded

k — L format Hollerith constant: name of library containing overlay

any other non-zero value: overlay loaded from global library set

OVERLAY (fname,i,j,Cn)	OVERLAY (TEST,0,0,C5500)

i — primary overlay number, octal

j — secondary overlay number, octal

Cn — n is 6 octal digits indicating start of load relative to blank common

---

Debug

The D option on FTN control card selects debugging mode; if it is not specified, debugging cards are treated as comments.

DEBUG statements are written in columns 7–72; columns 1 and 2 of each statement must contain C$. Any character, other than blank or zero, in column 6 denotes a continuation line. Columns 3, 4, and 5 of a continuation line must be blank.

C$ DEBUG

C$ DEBUG ($name_1, \ldots, name_n$)

C$ AREA $bounds_1, \ldots, bounds_n$
C$ DEBUG
} within program unit

C$ AREA/name$_1$/bounds$_1$, . . . ,bounds$_n$, . . . ,
/name$_n$/bounds$_1$, . . . ,bounds$_n$

C$ DEBUG (name$_1$, . . . ,name$_n$)
or
C$ DEBUG

} external debug deck

bounds (n_1,n_2) n_1 initial line position; n_2 terminal line position

(n_3) n_3 single line position to be debugged

(n_1,*) n_1 initial line position; * last line of program

(*,n_2) * first line of program; n_2 terminal line position

(*,*) first and last lines of program

C$ ARRAYS (a_1, . . . ,a_n)

C$ ARRAYS

a_i array names

C$ CALLS (s_1, . . . ,s_n)

C$ CALLS

s_i subroutine names

C$ FUNCS (a_1, . . . ,a_n)

C$ FUNCS

f_i function name

C$ GOTOS

C$ NOGO

C$ STORES (c_1, . . . ,c_n)

c_i variable name

variable name .relational operator. constant

variable name .relational operator. variable name

variable name .checking operator.

checking operators:

RANGE	out of range
INDEF	indefinite
VALID	out of range or indefinite

C$ TRACE (lv)

C$ TRACE

lv level number:

0	tracing outside DO loops
n	tracing up to and including level n in DO nest

C$ OFF

C$ OFF ($x_1, \ldots, x_n$)

x_i any debug option

PRINTER CONTROL CHARACTERS

Character	Action
Blank	Space vertically one line then print
0	Space vertically two lines then print
1	Eject to first line of next page before printing
+	No advance before printing; allows overprinting
Any other character	Refer to SCOPE Reference Manual

FORTRAN LIBRARY: INTRINSIC FUNCTIONS

Intrinsic Function	Definition	Number of Arguments	Symbolic Name	Type of Argument	Type of Function	Example
Absolute Value	|A|	1	ABS IABS DABS	Real Integer Double	Real Integer Double	Y=ABS(X) J=IABS(I) DOUBLE A,B B=DABS(A)
Truncation	Sign of A times largest integer $\leq$|A|	1	AINT INT IDINT	Real Real Double	Real Integer Integer	Y=AINT(X) I=INT(X) DOUBLE Z J=IDINT(Z)
Remaindering † (see note)	A1 (mod A2)	2	AMOD MOD	Real Integer	Real Integer	B=AMOD(A1,A2) J=MOD(I1,I2)
Choosing largest value	Max(A1, A2, . . .)	$\geq$2	AMAX0 AMAX1 MAX0 MAX1 DMAX1	Integer Real Integer Real Double	Real Real Integer Integer Double	X=AMAX0(I,J,K) A=AMAX1(X,Y,Z) L=MAX0(I,J,K,N) I=MAX1(A,B) DOUBLE W,X,Y,Z W=DMAX1(X,Y,Z)

Choosing smallest value	Min(A1, A2, . . .)	≥2	AMIN0 AMIN1 MIN0 MIN1 DMIN1	Integer Real Integer Real Double	Real Real Integer Integer Double	Y=AMIN0(I,J) Z=AMIN1(X,Y) L=MIN0(X,Y) J=MIN1(X,Y) DOUBLE A,B,C C=DMIN1(A,B)
Float	Conversion from integer to real	1	FLOAT	Integer	Real	X1=FLOAT(I)
Fix	Conversion from real to integer	1	IFIX	Real	Integer	IY=IFIX(Y)
Transfer of sign	Sign of A2 with |A1|	2	SIGN ISIGN DSIGN	Real Integer Double	Real Integer Double	Z=SIGN(X,Y) J=ISIGN(I1,I2) DOUBLE X,Y,Z Z=DSIGN(X,Y)

† MOD or AMOD (x1,x2) is defined as x1-⌊x1/x2⌋x2, where ⌊x⌋ is the largest integer that does not exceed the magnitude of x with sign the same as x.

FORTRAN LIBRARY: INTRINSIC FUNCTIONS (continued)

Intrinsic Function	Definition	Number of Arguments	Symbolic Name	Type of Argument	Type of Function	Example
Positive Difference	If A1>A2 then A1-A2. If A1 $\leq$A2 then 0.	2	DIM IDIM	Real Integer	Real Integer	A=DIM(C,D) J=IDIM(I1,I2)
Logical Product	Bit-by-bit logical AND of A_1 through A_n	n$\geq$2	AND	any type††	no mode	C=AND(A1,A2)
Logical Sum	Bit-by-bit logical OR of A_1 through A_n	n$\geq$2	OR	any type††	no mode	D=OR(A1,A2)
Exclusive OR	Bit-by-bit Exclusive OR of A_1 through A_n	n$\geq$2	XOR	any type††	no mode	D=XOR(A1,A2)
Complement	Bit-by-bit Boolean complement of A	1	COMPL	any type††	no mode	B=COMPL(A)

Shift	Shift A1,A2 bit positions: left circular if A2 is positive; right with sign extension, and end off if A2 is negative. If A2 is not a constant, with A2<0, and |A2| >63, the result is +0.	2	SHIFT	A1:any type†† A2:integer	no mode	B=SHIFT(A,I)
Mask	Form mask of A1 bits set to 1 starting at the left of the word. $0 \leq A1 \leq 60$.	1	MASK	Integer	no mode	A=MASK(B)

†† For a double precision or complex argument, only the high order or real part will be used.

FORTRAN LIBRARY: INTRINSIC FUNCTIONS (continued)

Intrinsic Function	Definition	Number of Arguments	Symbolic Name	Type of Argument	Type of Function	Example
Obtain Most Significant Part of Double Precision Argument		1	SNGL	Double	Real	DOUBLE Y X=SNGL(Y)
Obtain Real Part of Complex Argument		1	REAL	Complex	Real	COMPLEX A B=REAL(A)
Obtain Imaginary Part of Complex Argument		1	AIMAG	Complex	Real	COMPLEX A D=AIMAG(A)
Express Single Precision Argument in Double Precision Form		1	DBLE	Real	Double	DOUBLE Y Y=DBLE(X)

Express Two Real Arguments in Complex Form	A1+A2i (where $i^2=-1$)	2	CMPLX	Real	Complex	COMPLEX C C=CMPLX(A1,A2)
Obtain Conjugate of a Complex Argument	a-bi (where A=a+bi)	1	CONJG	Complex	Complex	COMPLEX X,Y Y=CONJG(X)

FORTRAN LIBRARY: BASIC EXTERNAL FUNCTIONS

Basic External Function	Definition	Number of Arguments	Symbolic Name	Type of Argument	Type of Function	Example
Exponential	e**A	1	EXP	Real	Real	Z=EXP(Y)
		1	DEXP	Double	Double	DOUBLE X,Y Y=DEXP(X)
		1	CEXP	Complex	Complex	COMPLEX A,B B=CEXP(A)
Natural Logarithm	$\log_e(A)$	1	ALOG	Real	Real	Z=ALOG(Y)
		1	DLOG	Double	Double	DOUBLE X,Y Y=DLOG(X)
		1	CLOG	Complex	Complex	COMPLEX A,B B=CLOG(A)
Common Logarithm	$\log_{10}(A)$	1	ALOG10	Real	Real	B=ALOG10(A)
			DLOG10	Double	Double	DOUBLE D,E E=DLOG10(D)

Trigono- metric Sine	sin (A)	1 1 1	SIN DSIN CSIN	Real Double Complex	Real Double Complex	Y=SIN(X) DOUBLE D,E E=DSIN(D) COMPLEX CC,F CC=CSIN(CD)
Trigono- metric Cosine	cos (A)	1 1 1	COS DCOS CCOS	Real Double Complex	Real Double Complex	X=COS(Y) DOUBLE D,E E=DCOS(D) COMPLEX CC,F CC=CCOS(F)
Hyperbolic Tangent	tanh (A)	1	TANH	Real	Real	B=TANH(A)
Square Root	$(A)^{1/2}$	1 1 1	SQRT DSQRT CSQRT	Real Double Complex	Real Double Complex	Y=SQRT(X) DOUBLE D,E E=DSQRT(D) COMPLEX CC,F CC=CSQRT(F)

FORTRAN LIBRARY: BASIC EXTERNAL FUNCTIONS (continued)

Basic External Function	Definition	Number of Arguments	Symbolic Name	Type of Argument	Type of Function	Example
Arctangent	arctan (A)	1 1	ATAN DATAN	Real Double	Real Double	Y=ATAN(X) DOUBLE D,E E=DATAN(D)
	arctan (A1/A2)	2 2	ATAN2 DATAN2	Real Double	Real Double	B=ATAN2(A1,A2) DOUBLE D,D1,D2 D=DATAN2(D2,D2)
Remaindering†	-A1 (mod A2)	2	DMOD	Double	Double	DOUBLE DM,D1,D2 DM=DMOD(D1,D2)
Modulus	$a^2 + b^2$ for A=a+bi	1	CABS	Complex	Real	COMPLEX C CM=CABS(C)
Arccosine	arccos (A)	1	ACOS	Real	Real	X=ACOS(Y)
Arcsine	arcsin (A)	1	ASIN	Real	Real	X=ASIN(Y)
Trigonometric Tangent	tan (A)	1	TAN	Real	Real	X=TAN(Y)

† The function DMOD (x1,x2) is defined as x1-⌊x1/x2⌋x2, where ⌊x⌋ is the largest integer that does not exceed the magnitude of x with sign the same as x.

LIBRARY SUBROUTINES AND FUNCTIONS

The following utility subprograms are supplied by the system. ANSI does not specify any library subroutines.

Functions

RANF (n) — Random number generator

LOCF (a) — Returns address of a

UNIT (u) — Returns buffer status on unit, u

- −1 Unit ready, no error
- +0 Unit ready, EOF encountered
- +1 Unit ready, parity error encountered

EOF (u) — Checks for end of file

- 0 No end of file encountered

LENGTH (u) — Returns number of words read on previous buffer or mass storage input/output request

IOCHEC (u) — Returns parity status on non-buffer unit

- 0 No read parity error

LEGVAR (a) — Checks variable a

- −1 Indefinite
- +1 Out of range
- 0 Normal

Subroutines

CALL DUMP (a_1,b_1,f_1, . . . ,a_n,b_n,f_n) — Dumps storage and terminates program execution

CALL PDUMP (a_1,b_1,f_1, . . . ,a_n,b_n,f_n) — Dumps storage and returns control to calling program

a first word } of storage area
b last word } to be dumped

f = 0 or 3, octal dump

f = 1, real dump

f = 2, integer dump

f = 4, octal dump

CALL SSWTCH (i,j) — Sense switch test

i 1 to 6. Integer variable or constant

j set to 1 if i is on; 2 if i is off. Integer variable

CALL REMARK (H) — Dayfile message

H Hollerith specification
≤ 80 characters

CALL DISPLA (H,k) — Displays name and value

H Hollerith specification
≤ 80 characters

k Variable or expression

CALL RANGET (n) — Current value of RANF

n Symbolic name to receive seed

CALL RANSET (n) — Initial value of RANF

n Dummy argument

†SECOND(t) or CALL SECOND (t) — Elapsed central processor time

†DATE(a) or CALL DATE (a) — Returns current date in format bMM/DD/YYb

†TIME(a) or CALL TIME (a) — Returns current time in format HH.MM.SS

CALL ERRSET (a,b) — Maximum number of errors

Sets maximum number of errors, b, before fatal termination; count is kept in a.

CALL LABEL (a,u) — Tape label information

CALL MOVLEV (a,b,n) — Moves data between extended core storage and central memory, or SCM and LCM

a and b — Variables or array elements

n — Integer constant or expression

Transfers n consecutive words of data between a and b. a is starting address of the data to be moved and b is starting address of receiving location.

CALL OPENMS (u,ix,lngth,t) — Opens mass storage file

CALL READMS (u,fwa,n,k) — Transmits data from mass storage to central memory

CALL WRITMS (u,fwa,n,k,r,s) — Transmits data from central memory to mass storage

CALL STINDX (u,ix,lngth,t) — Changes file index in central memory to base specified in call

CALL CLOSMS (u) — Writes index from central memory to file and closes file

---

†SECOND, TIME and DATE can be used as functions or subroutines. The value is always returned via the argument and the normal function return.

u — Unit number

ix — First word address in central memory of index

lngth — Length of index buffer: Number index, lngth $\geq$ (number of records in file) +1; name index, lngth $\geq$ 2* (number of records in file) +1

t — Index type: Number index 0; name index 1

fwa — First word address in central memory of data buffer area

n — Number of 60-bit words in data record to be transferred

k — Index key: Number index $1 \leq k \leq$ lngth-1; name index, k may refer to any 60-bit quantity except ± 0

r — Rewrite in place request

r = +1 in-place rewrite

r = −1 in-place rewrite if new record length does not exceed old record length; otherwise, write at end-of-information

r = 0 normal write at end-of-information

Parameter may be omitted if no subindex flag parameter is required. Default value is zero (normal write).

s — Subindex flag, may be omitted; default value is zero.

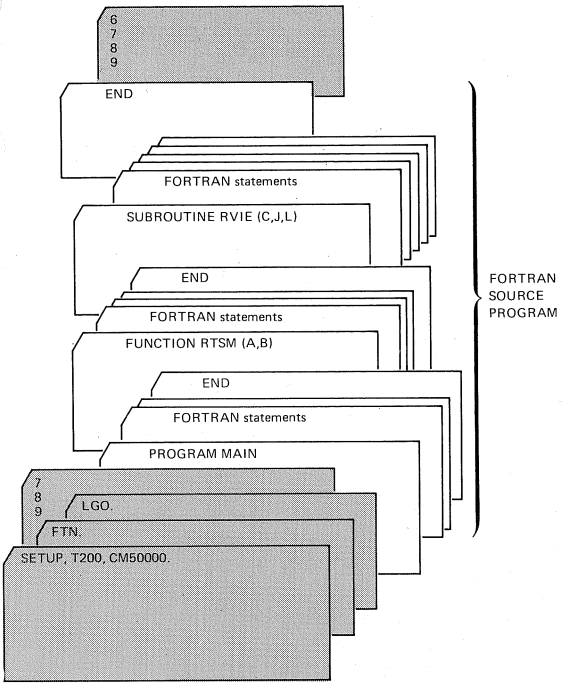
s = 1 index control word entry for record contains a special subindex marker flag

s = 0 subindex marker flag is not included in control word

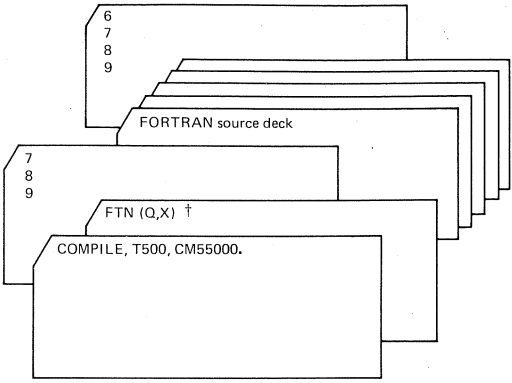
SAMPLE DECK STRUCTURES

FORTRAN SOURCE PROGRAM WITH SCOPE CONTROL CARDS

In the following sample deck SCOPE control cards are shaded. Refer to the SCOPE Reference Manual for details of these cards.



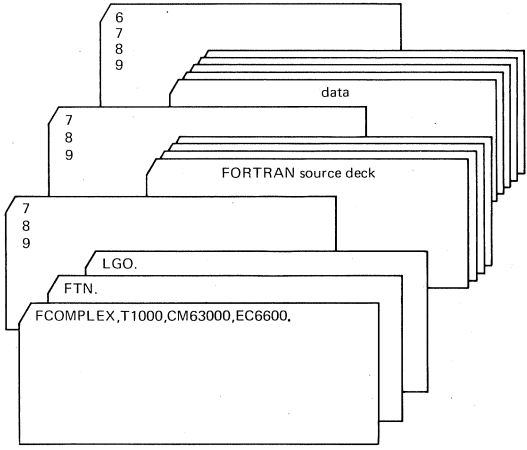
COMPILATION ONLY



† Q–Full semantic and syntactic scan of program. All diagnostics and complete reference map printed

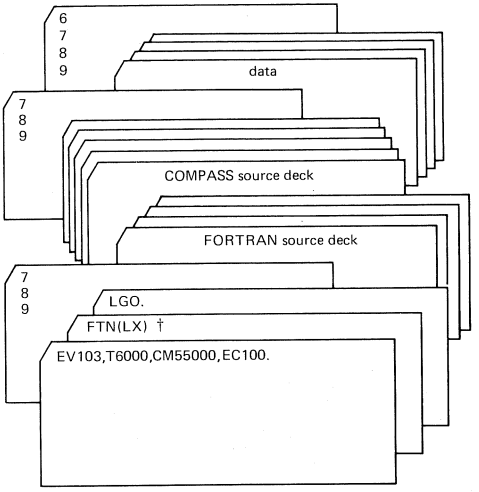
X–Warnings printed for non–ANSI usage

COMPILATION AND EXECUTION



FORTRAN COMPILATION WITH COMPASS ASSEMBLY AND EXECUTION

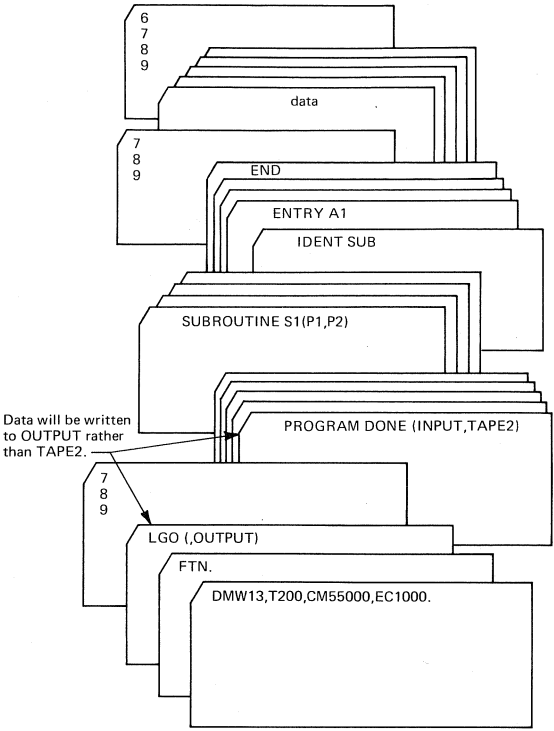
FORTRAN and COMPASS program unit source decks can be in any order. COMPASS source decks must begin with a card containing the word IDENTb in columns 11–16 and terminate with a card containing the word ENDb in columns 11–14 (b denotes a blank). Columns 1–10 of the IDENT and END cards must be blank.



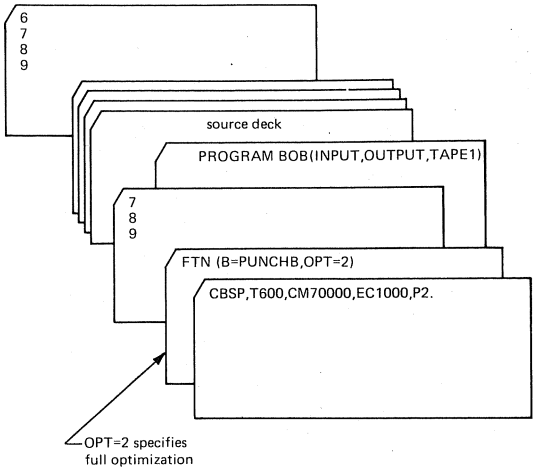
† L – Source program diagnostics, and short reference map listed

X – ANSI diagnostics listed

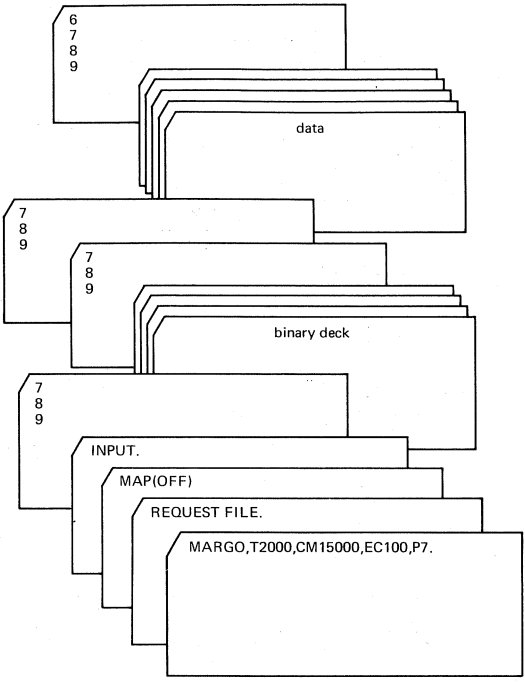
COMPILE AND EXECUTE WITH FORTRAN SUBROUTINE AND COMPASS SUBPROGRAM



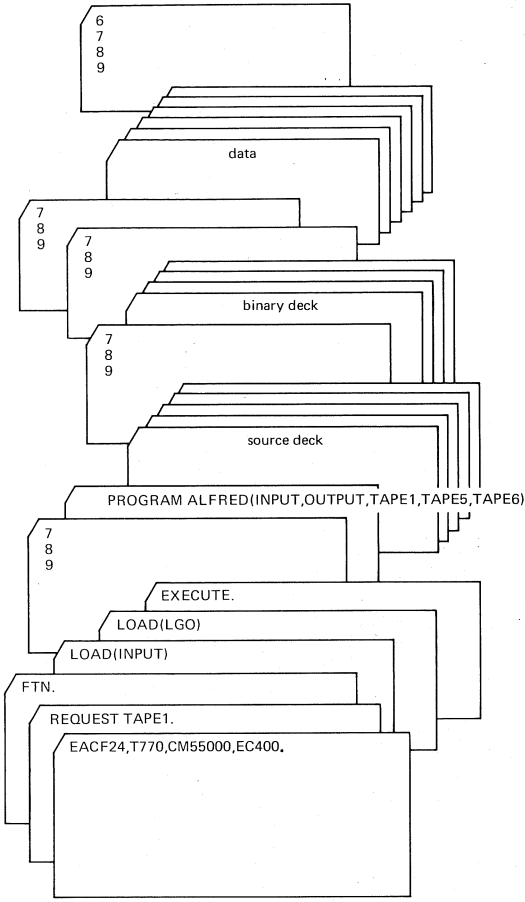
COMPILE AND PRODUCE BINARY CARDS



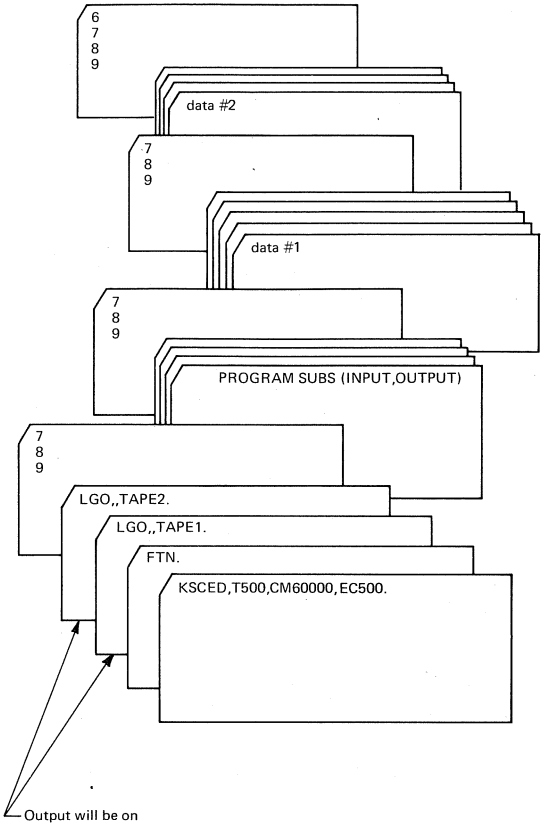
LOAD AND EXECUTE BINARY PROGRAM



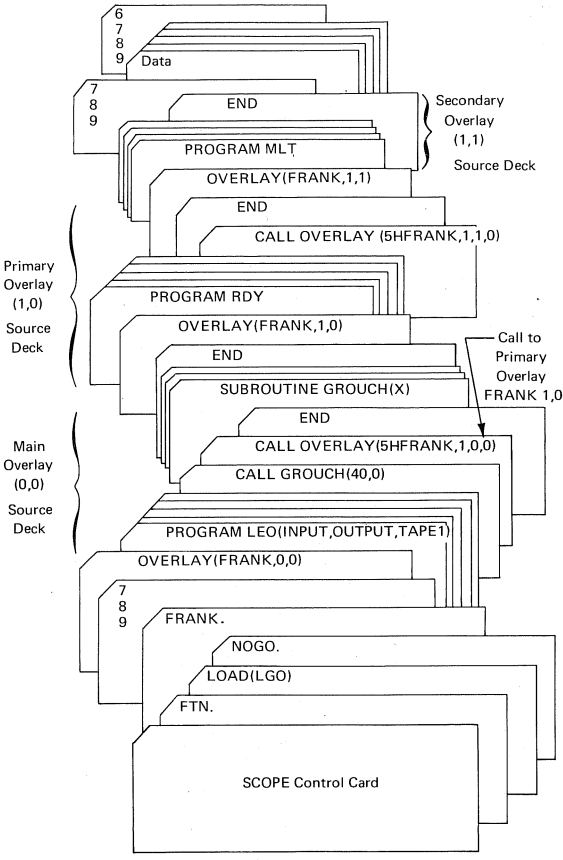
COMPILE AND EXECUTE WITH RELOCATABLE BINARY DECK



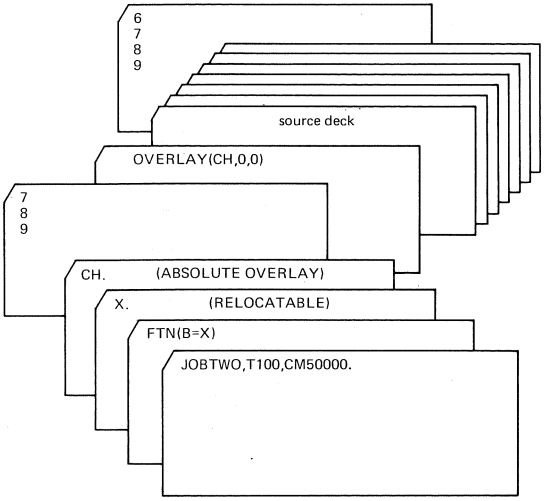
COMPILE ONCE AND EXECUTE WITH DIFFERENT DATA DECKS



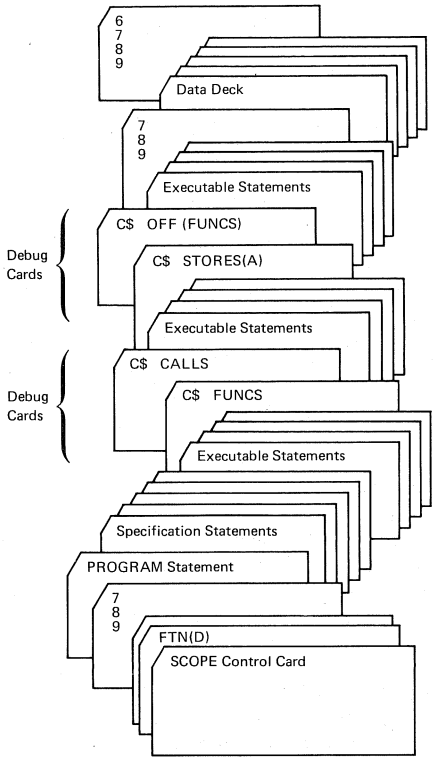
PREPARATION OF OVERLAYS



COMPILATION AND TWO EXECUTIONS WITH OVERLAYS

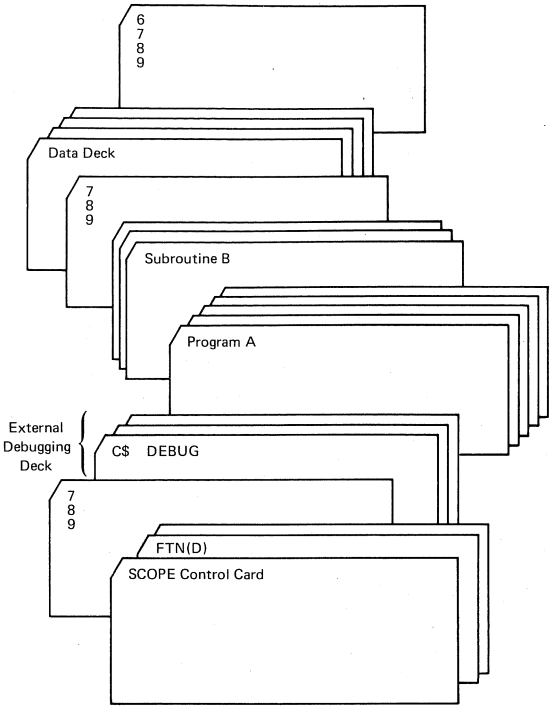


INTERSPERSED DEBUGGING STATEMENTS



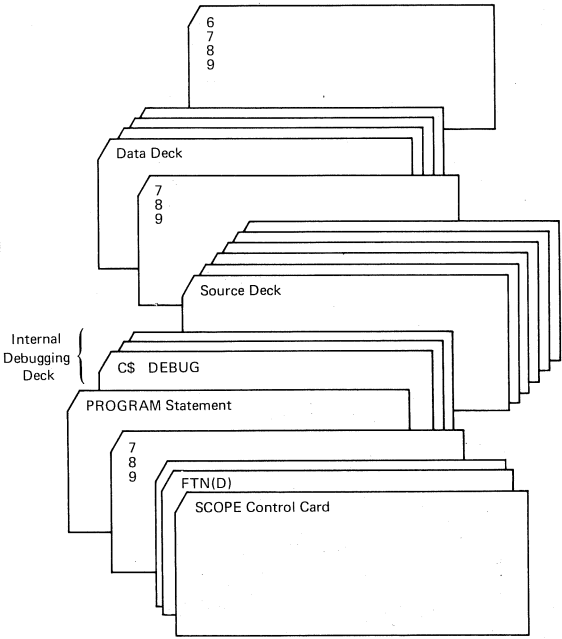
Debugging cards are interspersed; they are inserted at the point in the program where they will be activated.

EXTERNAL DEBUGGING DECK



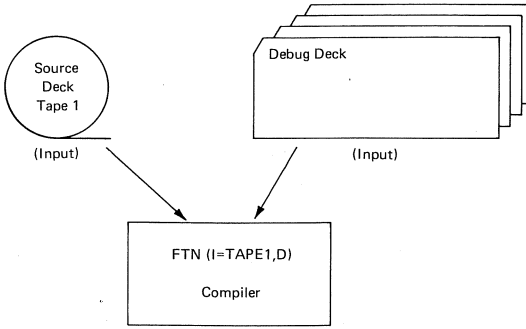
The external debugging deck is placed immediately in front of the first source line. All program units (here, Program A and Subroutine B) will be debugged (unless limiting bounds are specified in the deck). This positioning is particularly useful when a program is to be run for the first time, since it ensures that all program units will be debugged.

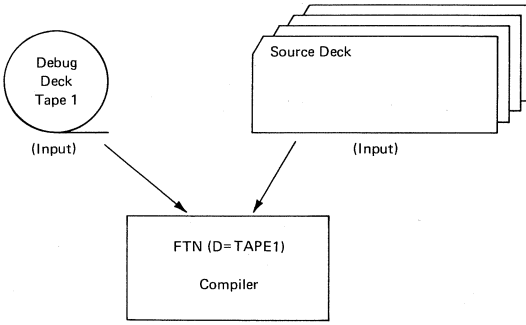
INTERNAL DEBUGGING DECK



When the debugging deck is placed immediately after the program name card and before any specification statements, all statements in the program unit will be debugged (unless limiting bounds are specified in the deck); no statements in other program units will be debugged. This positioning is best when the job is composed of several program units known to be free of bugs and one unit that is new or known to have bugs.

EXTERNAL DECK ON SEPARATE FILE





The debugging deck is placed on a separate file (external debugging deck) named by the D parameter on the FTN control card and called in during compilation. All program units will be debugged (unless the program units to be debugged are specified in the deck). This positioning is useful when several jobs can be processed using the same debugging deck.

FORTRAN CONTROL CARD

FTN. comments

FTN ($p_1, \ldots, p_n$) comments

FTN,$p_1, \ldots, p_n$. comments

The optional parameters, $p_1, \ldots, p_n$, may appear in any order within the parentheses. All parameters, with the exception of the list control options, must be separated by commas.

FTN. is equivalent to FTN (I=INPUT,L=OUTPUT,B=LGO,S=SYSTEXT, R=1,OPT=1).

Parameters

I SOURCE INPUT PARAMETER (Default I=INPUT)

I=lfn	lfn is name of source input file; this form must be used if source input file is other than INPUT.
I=INPUT	Source input is on file INPUT
I only	Source input is on file COMPILE Compilation stops when an end-of-record (section) or end-of-file (partition) is encountered.

B BINARY OBJECT FILE (Default B=LGO)

B	Generated binary object code is output on file LGO.
B=lfn	Generated binary object code is output on file lfn.
B=0	No binary object file is produced.
G=lfn BG=lfn GB=lfn G	Binary object file is loaded and executed at end of compilation

L LIST CONTROL (Default L=OUTPUT, R=1)

y=lfn	y is the type of listing of the source program; any combination of one to four options selected from L O R X N. Commas must not be used; X and N cannot be specified at the same time. If no options are specified, the source program with informative and fatal diagnostics is listed. If no file is specified, OUTPUT is assumed. lfn is file to receive output listing.
L=lfn	Source program, diagnostics, and short reference map listed (default listing).
L	L defaults to L=OUTPUT.
L=0 or LR=0	Fatal diagnostics and the statements which caused them are listed, all other output, including inter-mixed COMPASS, is suppressed.
L=0,R=1 L=0,R=2 L=0,R=3	Level 1 reference map Level 2 reference map Level 3 reference map } fatal diagnostics and statements which caused them are suppressed.
O=lfn	Generated object code listed; O must not be used if E option is selected.
R=lfn	Symbolic reference map listed.
X=lfn	A warning diagnostic is listed for any non-ANSI usage.
N=lfn	Listing of informative diagnostics is suppressed; only diagnostics fatal to execution are listed.

Example:

LRON = lfn specifies all options except non-ANSI diagnostics are to be listed; LO selects source program and generated object code listing on OUTPUT.

E EDITING PARAMETER (Default E=COMPS)

E or E=lfn	Compiler generated object code is output as COMPASS card images for the SCOPE maintenance program UPDATE. If E is omitted, normal binary object file is produced. O and C options must not be specified if E is selected.

An object code output file is delimited with the card images: *DECK, name and *END (name identifies the program unit).

The object code output file lfn or COMPS is rewound and ready as UPDATE input. No binary file is produced. COMPASS is not called automatically. When the COMPS file is assembled S = FTNMAC must be specified on the COMPASS control card.

T ERROR TRACEBACK (Default T omitted)

T	Calls to library functions are made with call-by-name sequence. Error checking is maximum with full error traceback.
T omitted	Call-by-value linkages are generated. Error checking is minimum; no traceback occurs. Saves memory space and execution time.

Selecting the D parameter or OPT=0 automatically selects T.

ROUNDED ARITHMETIC SWITCH (Default: arithmetic not rounded)

ROUND=op	op is arithmetic operator: + - */. Single precision (real and complex) floating point arithmetic operations use hardware rounding features. Any combination of arithmetic operators can be specified. For example: ROUND = + - /

D DEBUGGING MODE PARAMETER

D or D=lfn	If the debug facility is used, D or D=lfn must be specified, and fast compilation (OPT=0) and full error traceback (T option) are automatic. When the debug parameter is selected, any optimization level other than OPT=0 is ignored. A minimum field length of 61000 should be specified on the SCOPE control card. lfn names the file containing the user debug deck. Default option for D=lfn is D=INPUT.

FTN(D) is equivalent to FTN(D=INPUT,OPT=0,T)

A EXIT PARAMETER

A	If fatal errors occur, compilation terminates and a branch is made to an EXIT(S) control card. If no EXIT(S) control card appears, the job terminates.

Note: S, GT and SYSEDIT parameters are of interest primarily to system programmers.

S SYSTEM TEXT FILE (Default S=SYSTEXT)

S=lfn	Source of systems text information for intermixed COMPASS assemblies is on file lfn. If the GT parameter is GT=0 only, or GT is omitted, the overlay named SYSTEXT is loaded. If parameter is omitted, information is on SYSTEXT overlay.
S=0	When COMPASS is called to assemble intermixed COMPASS programs, it will not read in a system text file.
S=ovlname	The system text overlay, ovlname, is loaded from the job's current library set.
S=libname/ ovlname	The system text overlay, ovlname, is loaded from the library, libname. Libname can be a user library file or a system library.

GT GET SYSTEM TEXT FILE

GT=lfn	lfn is sequential binary file, loads first system text overlay.
GT=lfn/ ovlname	Searches lfn for system text overlay named ovlname, and loads the first encountered.
GT=0 or omitted	No system text is loaded.

Any combination of the GT, S and C parameters must not specify more than seven system texts.

SYSEDIT SYSTEM EDITING (Default SYSEDIT not selected)

This option is used mainly for system resident programs.

SYSEDIT or SYSEDIT=FILES and SYSEDIT=IDENT	All input/output references are accomplished indirectly through a table search at object time. File names are not entry points in main program, and subprograms do not produce external references to the file name. The IDENT specification causes a $ sign to be suffixed to the program name in both IDENT and ENTRY cards if it duplicates the program name of any FORTRAN object library program.

V SMALL BUFFERS OPTION

V	Compiler uses 513-word buffers for intermediate files. Programs with a large number of specifications are compiled with a smaller field length under this option. Since less space is used in the buffers, compile time may increase. On a 7600 control card, V will be ignored.

C COMPASS ASSEMBLY

C	The COMPASS assembler is used for code generated by FTN. If C is omitted, the faster FTN assembler is used. When C is specified, FTNMAC is supplied as additional text for the COMPASS assembly; therefore, no more than six system texts can be specified by GT and S parameters. When C is specified, the SCOPE loader control card LDSET is required: LDSET (LIB=FORTRAN/SYSIO).

R SYMBOLIC REFERENCE MAP (Default R=1)

R=0	No map
R=1	Short map (symbols, addresses, properties)
R=2	Long map (symbols, addresses, properties, references by line number and a DO-loop map)
R=3	Long map with printout of common block members and equivalence groups

PL PRINT LIMIT (Default n=5000)

PL=n	n is maximum number of records that can be written on OUTPUT file at execution time. n does not include the records in source program listing, nor compilation and execution time listings; n $\leq$ 999 999 999
PL=nB	Octal number must be suffixed with a B; n $\leq$ 777 777 777B

Q PROGRAM VERIFICATION

Compiler performs full syntactic and semantic scan of the program and prints all diagnostics, but no object code is produced. A complete reference map is produced (with exception of code addresses). This mode is substantially faster than a normal compilation; but it should not be selected if the program is to be executed. If Q is omitted, normal compilation takes place.

Z ZERO PARAMETER

Z	All subroutine calls with no parameters are forced to pass a parameter list consisting of a zero word. This feature is useful only to COMPASS subroutines expecting a variable number of parameters (0 to 63). For example, CALL DUMP dumps storage on the OUTPUT file and terminates program execution. If no parameters are specified, a zero word parameter is passed. Z should not be specified unless necessary, as execution is more efficient when Z is omitted.

LCM LARGE CORE MEMORY ACCESS (Default LCM=D)

LCM=D	Selects 17-bit address mode for level 2 data (most efficient method for generating code for data assigned to level 2). User LCM field length must not exceed 131,071 words.
LCM=I	Selects 21-bit address mode for level 2 data; this mode depends heavily upon indirect addressing. LCM=I must be specified if user LCM field length exceeds 131,071 words.
	In neither case can a single common block be greater than 131,071 decimal words.

OPT OPTIMIZATION PARAMETER

OPT=m

m=0	Fast compilation (automatically selects T option)
m=1	Standard compilation and execution
m=2	Fast execution

CORPORATE HEADQUARTERS, 8100 34th AVE. SO.
MINNEAPOLIS, MINN. 55440

SALES OFFICES AND SERVICE CENTERS
IN MAJOR CITIES THROUGHOUT THE WORLD

Pub. No. 60357900 Litho in U.S.A.