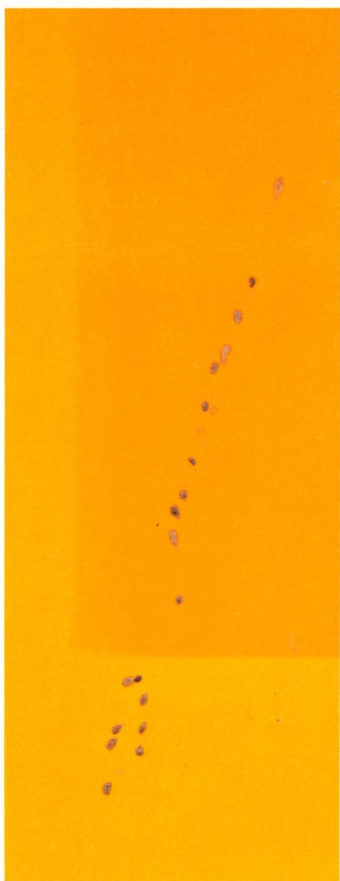# Honeywell

FORTRAN POCKET GUIDE

SERIES 60 (LEVEL 66)/6000

SOFTWARE

Honeywell FORTRAN
POCKET GUIDE

SERIES 60(LEVEL66)/6000

SUBJECT:

FORTRAN Programming Information.

SOFTWARE SUPPORTED:

Series 60 Level 66 Software Release 2
Series 6000 Software Release H

DATE:

January 1975

ORDER NUMBER:

DD82, Rev. 0

PREFACE

This guide is based on material extracted from the reference manual, <u>FORTRAN</u>, Order Number DD02. Refer to DD02 for full explanations of arguments, definitions, etc. contained in this guide.

CONTENTS

STATEMENTS

Assignment

    Arithmetic

        v = e

                v — variable name or array element name
                    of arithmetic type
                e — arithmetic expression

    Logical

        v = e

                v — logical variable name or array element
                e — logical expression

    Character

        v = e

                v — character variable or array element
                e — character constant, variable, function,
                    or array element

    Label

        ASSIGN k TO i

                k — statement number
                i — nonsubscripted integer variable

ABNORMAL

    ABNORMAL
    ABNORMAL $a_i, \ldots, a_n$

           a_i — function subprogram name

BACKSPACE

    BACKSPACE f

        f — file reference

BLOCK DATA

    BLOCK DATA

          .
          .
          .

    END

CALL

    CALL s
    CALL s $(a_1, \ldots, a_n, \$b_1, \ldots, \$b_n)$

        s – name of SUBROUTINE
        a_i – actual argument
        b_i – optional alternate return

CHARACTER

    CHARACTER $*b\ a_1\ *s_1\ (k_1)/d_1/, \ldots, a_n\ *s_n\ (k_n)/d_n/$

        $*b$ – optional length and must be specified as
            unsigned integer constant
        a_i – variable, array, or FUNCTION subprogram
            name
        $*s_i$ – maximum number of characters ($\leqslant 511$)
            contained by element
        k_i – dimension information
        d_i – initial data value(s)

COMMON

    COMMON $a_1, \ldots, a_n$
    COMMON $/x_1/a_1 \ldots /x_n/a_n$

        x_i – symbolic name for labeled common block
        a_i – variable name, array name, or array
            declarator

COMPLEX

    COMPLEX $a_1\ *s_1\ (k_1)/d_1/, \ldots, a_n\ *s_n\ (k_n)/d_n/$

        a_i – variable, array, or FUNCTION subprogram
            name
        $*s_i$ – optional size (bytes)
        k_i – dimension information
        d_i – initial data value(s)

CONTINUE

    CONTINUE

        Most often used as nonoperational statement for
        placement of statement number.

DATA

    DATA $k_1/d_1/, \ldots, k_n/d_n/$

        k_i – list of names of variables, arrays, array
            elements, and implied DOs
        d_i – list of constants specifying initial data

DECODE

    DECODE (a, t) list

        a – character scalar, **array**, or **array** element
        t – FORMAT statement number, character
            scalar, or array name
        list – as specified for READ statement

DIMENSION

    DIMENSION $v_1(i_1)/d_1/, \ldots, v_n(i_n)/d_n/$

        v_i – array declarator
        i_i – one to seven unsigned **integer constants**,
            parameters, or variables specifying array
            size and dimensions
        d_i – optional initial data value(s)

DO

    DO n i = m_1, m_2
    DO n i = m_1, m_2, m_3

        n – statement number of executable statement
        i – nonsubscripted integer variable
        m_i – any valid arithmetic expression

DOUBLE PRECISION

    DOUBLE PRECISION $a_1\ *s_1\ (k_1)/d_1/, \ldots,$
                     $a_n\ *s_n\ (k_n)/d_n/$

        a_i – variable, array, or FUNCTION subprogram
            name
        $*s_i$ – optional size (bytes)
        k_i – dimension information for allocating
            storage to arrays
        d_i – initial data value(s)

ENCODE

    ENCODE (a, t) list

        a – character **scalar, array**, or array element
        t – FORMAT statement number, character
            scalar, or array name
        list – as specified for WRITE statement

END

    END

        Must be last statement of every program.

ENDFILE

ENDFILE f

    f   – file reference

ENTRY

ENTRY name $(b_1, \ldots, b_n)$

    name – symbolic name of **entry** point
    b_i – dummy argument corresponding to actual
           argument in CALL statement

EQUIVALENCE

EQUIVALENCE $(k_1), \ldots, (k_n)$

    k_i – list of form $a_i, \ldots, a_n$, where a_i is
          variable name or array element name

EXTERNAL

EXTERNAL $a_1, \ldots, a_n$

    a_i – subprogram **name**

FORMAT

m FORMAT $(q_1 \ r_1 \ z_1 \ \cdots \ r_n \ z_n \ q_2)$
m FORMAT (V)

    m – statement number
    q_i – series of slashes, or null
    r_i – field descriptor
    z_i – field separator (comma or slash)
    V – indication of I/O under list control

FUNCTION

FUNCTION name $(a_1, \ldots, a_n)$
Type FUNCTION name $(a_1, \ldots, a_n)$

    name – symbolic name of single valued function
    a_i – nonsubscripted variable or dummy
          name of subprogram
    type – REAL, INTEGER, DOUBLE
          PRECISION, COMPLEX, LOGICAL,
          or CHARACTER

GO TO

Assigned

    GO TO i
    GO TO i, $(k_1, \ldots, k_n)$

        i – integer or switch variable
        k_i – statement number

Unconditional

    GO TO k

        k – statement number

Computed

    GO TO $(k_1, \ldots, k_n)$, **e**

        k_i – statement label or switch variable
        e – arithmetic expression

IF

Arithmetic

    IF (e) k_1, k_2, k_3

        e – arithmetic expression
        k_i – stafement number, switch variable, or
           null

Logical

    IF (e) s

        e – logical or relational expression
        s – any executable statement except DO or
          another logical IF

IMPLICIT

IMPLICIT type *s (alpha$_1$, \ldots, alpha$_n$), \ldots

    type – REAL, INTEGER, COMPLEX, DOUBLE
          PRECISION, LOGICAL, or CHARACTER
    *s   – optional length specification of associated
          data type
    alpha$_i$– letter or dash-separated letters specify-
          ing implicit typing for variable names
          in program following

INTEGER

INTEGER $a_1 *s_1 (k_1)/d_1/,\ldots,a_n *s_n (k_n)/d_n/$

- a_i – variable, array, or FUNCTION subprogram name
- $*s_i$ – optional size (bytes)
- k_i – dimension information for allocating storage to arrays
- d_i – initial data value(s)

LOGICAL

LOGICAL $a_1 *s_1 (k_1)/d_1/,\ldots,a_n *s_n (k_n)/d_n/$

- a_i – variable, array, or FUNCTION subprogram name
- $*s_i$ – optional size (bytes)
- k_i – dimension information for allocating storage to arrays
- d_i – initial data value

NAMELIST

NAMELIST$/n_1/k_1 \ldots /n_n/k_n/$

- n_i – NAMELIST name
- k_i – list of variables or array names

PARAMETER

PARAMETER $v_1 = e_1,\ldots,v_n = e_n$

- v_i – PARAMETER symbol
- e_i – expression

PAUSE

PAUSE
PAUSE n

- n – integer, character constant, or variable

PRINT

PRINT, list
PRINT t, list
PRINT t
PRINT x

- list – input variable or array list to be printed
- t – FORMAT statement number, character scalar, or array name
- x – NAMELIST name

PUNCH

PUNCH, list
PUNCH t, list
PUNCH t
PUNCH x

- list – input variable or array list to be punched
- t – FORMAT statement number, character scalar, or array name
- x – NAMELIST name

READ

READ, list
READ t, list
READ t
READ x
READ (f, x)
READ (f, t, END = S1, ERR = S2) list
READ (f, END = S1, ERR = S2) list
READ (f' n, ERR = S2) list
READ (f, x, END = S1, ERR = S2)

- list – input variable or array list to be read
- t – FORMAT statement number, character scalar, or array name
- x – NAMELIST name
- f – file reference
- n – integer constant, expression, or variable specifying sequence number of logical record to be accessed
- S1 – optional statement label to receive control upon encountering end of file
- S2 – optional statement label to receive control upon encountering error condition

REAL

REAL $a_1 *s_1 (k_1)/d_1/,\ldots,a_n *s_n (k_n)/d_n/$

- a_i – variable, array, or FUNCTION subprogram name
- $*s_i$ – size specification
- k_i – dimension information for allocating storage to arrays
- d_i – initial data value

RETURN

RETURN
RETURN i

- i – positive integer constant or variable denoting ith nonstandard return in argument list, reading from left to right of CALL statement invoking this returning subroutine

REWIND

REWIND f

    f — file reference

STOP

STOP
STOP n

    n — integer or character constant or variable

SUBROUTINE

SUBROUTINE name $(a_1, \ldots, a_n)$
SUBROUTINE name

    name — symbolic name of subprogram
    a_i — nonsubscripted variable or array name,
          dummy name of subprogram, or * or $
          indicating nonstandard return

Type

type $*b \; a_1 \; *s_1 \; (k_1)/d_1/, \ldots, a_n \; *s_n \; (k_n)/d_n/$

    type — REAL, INTEGER, DOUBLE PRECISION,
          COMPLEX, LOGICAL, or CHARACTER
    $*b$ — default value for missing s_i
    a_i — variable, array, or FUNCTION sub-
          program name
    $*s_i$ — optional positive integer constant size
          specification
    k_i — dimension information for allocating
          storage to arrays
    d_i — initial data value(s)

WRITE

WRITE (f, t, ERR = S2) list
WRITE (f, ERR = S2) list
WRITE (f 'n, ERR = S2) list
WRITE (f, x, ERR = S2)

    f   — file reference
    t   — FORMAT statement number, character
          scalar, or array name
    list — input variable or array list to be printed
          or punched
    x   — NAMELIST name
    n   — integer constant, variable, or expres-
          sion that specifies sequence number of
          logical record to be accessed
    S2   — optional statement label to receive
          control upon encountering error condi-
          tion

USER INTERFACES

BATCH MODE

Batch Call Card

1	8	16
$	FORTY	options
	or	
$	FORTRAN	options

Options underlined are assumed.

LSTIN — listing	}	of source input
NLSTIN — no listing		
LSTOU — listing	}	pseudo-assembly listing
NLSTOU — no listing		
DECK — binary	}	object program deck
NDECK — no binary		
COMDK — compressed	}	source deck
NCOMDK — not compressed		
MAP — storage map	}	of program labels,
NOMAP — no storage map		variables, constants
XREF — listing	}	of symbol cross-reference table, to-from table, GMAP offset on LSTIN report
NXREF — no listing		
DEBUG — debug	}	symbol table
NDEBUG — no debug		
BCD — character set is BCD	}	for object program
ASCII — character set is ASCII		
FORM — standard statement format	}	for source program
NFORM — free form		
LNO — source records line-numbered		
NLNO — source records not line-numbered		
OPTZ — performance	}	of global optimization procedures
NOPTZ — no performance		
DUMP — slave memory	}	dump on abnormal termination of compiler
NDUMP — program registers, upper SSA, slave program prefix		

JREST — job restart ⎫
NJREST — no job restart ⎬ on system interruption

REST — activity restart ⎫
NREST — no activity restart ⎬ on system interruption

NWARN — do not print any compilation warning message

Batch Deck Setup Example

1	8	16
$	SNUMB	
$	IDENT	
$	OPTION	FORTRAN
$	FORTY	Options or $ FORTRAN Options
	⋮ ⎫	FORTRAN Source Deck(s)
	⋮ ⎬	
	⋮ ⎭	
$	EXECUTE	Options
$	File Cards	
	⋮ ⎫	Data
	⋮ ⎭	
$	ENDJOB	
***EOF		

NOTE: If ASCII option is selected for $ FORTY or
$ FORTRAN card, follow card with $ USE
.GTLIT card.

TIME SHARING MODE

Two time sharing versions exist — batch based and
time sharing based. Former is called by YFORTRAN,
latter by FORTRAN.

Log-On Procedure

HIS SERIES 6000 ON 07/26/74 AT 14.768 CHANNEL
0012

USER ID — (user identification)
PASSWORD
XXXXXXXXXXX — (password typed over mask)
SYSTEM? YFORTRAN or FORTRAN
OLD OR NEW — (user selection)
READY
* (user begins entry)

General Format of Line of Input

nnnnnnnnc statement or continuation; statement . . . ;
statement
or
nnnnnnnnc comment

nnn...n numeric line number

c single-character control character; can be blank,
ampersand, asterisk, number sign, or letter C;
must immediately follow last digit of line number

฿ (blank)	If character position immediately following last digit of line number contains blank, and next nonblank character is not ampersand, then next nonblank character is assumed to begin new FORTRAN statement.
& (ampersand)	If ampersand is first nonblank character following line number, next nonblank character is assumed to be continuation of previous statement in previous line of input.
* (asterisk) or C	If line number is terminated by asterisk or letter C, information following is assumed to be comment.
# (pound sign)	If pound sign immediately follows a line number, character following pound sign is placed in column 1.

Time Sharing System Commands

Command	Applicable At Build Mode
ABC	Yes
ACCESS	Yes
ASCASC	Yes
ASCBCD	Yes
AUTOMATIC a	Yes
BCDASC	Yes
BPRINT	Yes
BPUNCH	Yes
BYE	Yes
CATALOG	Yes
DELETE a	Yes
DONE	Yes
EDIT	Yes
ERASE a	Yes
FDUMP	Yes
GET	Yes
HELP	Yes
HOLD	Yes
JABT	Yes
JOUT	Yes
JSTS	Yes
LENGTH	Yes
LIB a	Yes
LIST	Yes
NEW a	Yes
NEWUSER	Yes
NO PARITY	Yes
OLD a	Yes
PARITY	Yes
PERM a	Yes
PRINT a	Yes
PURGE a	Yes
RECOVER	Yes
#RECOVER	No
RELEASE a	Yes
REMOVE	Yes
RESAVE a	Yes
RESEQUENCE a	Yes
ROLLBACK	Yes
#ROLLBACK	No
RUN a	Yes
SAVE a	Yes
SCAN	Yes
SEND	Yes
STATUS	Yes
SYSTEM a	Yes
TAPE a	Yes

a
Applicable response must be made to return to subsystem-selection level

RUN Command for YFORTRAN or FORTRAN

NOTE: Items relating to YFORTRAN only indicated by Y; items relating to FORTRAN only indicated by F.

RUN[H] - nnn fs = fh; fc (opt) ulib #fe

- nnn
: maximum time of processor time (seconds)

fs
: set of file descriptors for source files in BCD, COMDK, or ASCII and/or descriptors for object file. File descriptor * indicates current file (*SCR).

fh
: single file descriptor of random file into which system loadable file is saved

fc
: single file descriptor of sequential file into which binary (C*) result is placed

F
: if the named file does not exist, quick access permanent file of three llinks is created and expands as necessary up to 20 llinks to hold object deck(s). When C* is specified, a file of 48 llinks is defined and its name placed in AFT.

opt
: options (underlined options are assumed)

BCD	object character set is BCD
ASCII	object character set is ASCII
FORM	source is in fixed format
NFORM	source is in free format
LNO	source is line numbered
NLNO	source is not line numbered
OPTZ	optimize object module
NOPTZ	no optimization of object module
GO	program is executed
NOGO	program is not executed
ULIB	field descriptors exist following end of options field that locate user libraries to be searched for missing routines prior to searching for them in system library
NOLIB	no user libraries searched

Y
: TIME = nnn activity time limits set to nnn seconds, where nnn ≤ 180; if not specified, nnn set to 60

	CORE = nn	memory requirements set to nnK + 6K or 24K, whichever is larger; if not specified, nn set to 16
Y		
F	CORE = nn	nn is additional memory to be added if message "⟨F⟩ PROGRAM EXCEEDS STORE SIZE" appears
Y	URGC = nn	urgency for activity set to nn, where nn ⩽ 40; if not specified, nn set to 40
Y	TEST	test file FORTRANY used for batch activity, allocated as file code *
Y	REMO	temporary files (batch) removed from AFT
Y	NAME = name	name provided for main link of saved H* file

ulib   sequence of file descriptors pointing to random files containing user libraries to be searched

fe   set of file descriptors for files required during execution

RUNL Command

RUNL[H] C* file list = H* file (opt) [ulib file]; link list

C* file list   set of file descriptors for object file

H* file   single file descriptor of random file in which system loadable file is saved; if named file does not exist, file of 216 llinks created

opt   options (underlined options assumed)

     ULIB   file descriptors exist following end of options field that locate user libraries to be searched prior to searching system library

     NOLIB   no user libraries searched

     CORE - nn   batch memory requirements set to nn+6K or 23K, whichever is larger; if not specified, nn set to 16

         time sharing memory requirements set to nnK if < 23K or 23K+nnK if nn > 23

NAME=name	name provided for main link of saved H* file; when not provided, name ////// is used
MAP	if file named PSTR is previously defined, load map of link/overlay is written to that file; otherwise temporary file created by that name and written to
GO	execution entered directly from RUNL command
NOGO	execution not entered

ulib file   optional file descriptor for user "RANLIB"

link list   sequence of link phrases wherein each link phrase is used to specify position at which segmentation is to take place

FILE MEDIA CODES

0 – formatted BCD line images, without slew control for the printer
1 – compressed BCD card images
2 – (uncompressed) BCD card images
3 – formatted BCD line images, with trailing printer slew control information
5 – time sharing ASCII format (pre-Series 6000 Software Release E)
6 – time sharing ASCII standard system format
7 – ASCII print line images, with trailing printer slew control information
8 – TSS information record

FILE DESIGNATORS

05 – standard input file (I*)
06 – standard output file (P*)
41 – standard input file (I*)
42 – standard output file (P*); printer destination
43 – standard output file (P*); punch destination

PRINT CONTROL CHARACTERS

Character	Effect
0	1 blank line prior to print
+	Overprint
1	Slew to top of page before printing
Any other	Space to next line

INPUT/OUTPUT

NOTATION

list	–	input/output list
f	–	file reference or designator
t	–	FORMAT information
x	–	NAMELIST name
a	–	internal storage buffer for packed character data
S1	–	optional statement label transfer for end of file
S2	–	optional statement label transfer for error

LIST DIRECTED FORMATTED INPUT/OUTPUT STATEMENTS

READ t, list
PRINT t, list
PUNCH t, list

READ, list
PRINT, list
PUNCH, list

READ (f, t, END = S1, ERR = S2) list
WRITE (f, t, ERR = S2) list

UNFORMATTED SEQUENTIAL FILE INPUT/OUTPUT STATEMENTS

READ (f, END = S1, ERR = S2) list
WRITE (f, ERR = S2) list

UNFORMATTED RANDOM FILE INPUT/OUTPUT STATEMENTS

READ (f'n, ERR = S2) list
WRITE (f'n, ERR = S2) list

FILE HANDLING STATEMENTS

REWIND f
BACKSPACE f
ENDFILE f

INTERNAL DATA CONVERSION STATEMENTS

ENCODE (a, t) list
DECODE (a, t) list

NAMELIST INPUT/OUTPUT

READ (f, x, END = S1, ERR = S2)
READ (f, x)
READ x

WRITE (f, x, ERR = S2)
PRINT x
PUNCH x

SUPPLIED SUBROUTINE SUBPROGRAMS

Call	Use
ANYERR (unique variable)	Placement of error code.
ATTACH (file, catfile, iprmis, mode, istat, buffer)	Access existing perm file.
CALLS (string, name)	Call time sharing subsystem (for time sharing only)
CNSLIO (console, message, nwords, nreply, nrepws)	Console communications (for batch only)
CONCAT (string, 1st char., replace string, replace 1st char., no of chars.)	Move character substring.
CORFL (array, no. of words, location, 0 or 1)	Move data from/to stored file.
CORSEC (real variable)	Memory allocation x processor time.
CREATE (file, isize, mode, istat)	Create temporary mass storage on terminal file.
DATIM (2-char. variable, real variable)	Get current date and time.
DEFIL (name, links, mode, istat)	Create temporary file.
DETACH (file, istat, buffer)	Deaccess current file.
DUMP (area$_i$, ..., dump format) DUMPA (area$_i$, ..., dump format)	BCD ⎱ Dump designated area of memory in specified format; terminate ASCII ⎰ execution.
DVCHK (integer variable)	Divide check test.
EXIT	Purge buffers and terminate current activity.
FCLOSE (file)	Close file and release buffers.
FILBSP (file, no. of files skipped)	Backspace files on multifile tape.
FILFSP (file, no. of files skipped)	Forwardspace files on multifile tape.

Call	Use
FLGEOF (file, integer variable)	End of file processing.
FLGERR (file, integer variable)	Data error processing.
FLGFRC (file, return)	File and Record Control I/O error recovery.
FMEDIA (file code, media code)	Output transliteration.
FPARAM (code, line length)	Set or reset I/O parameters.
FXALT (alt. error location)	Set alternate error procedure location.
FXALT ($ statement no.)	Alternate error return.
FXDVCK (value$_1$, value$_2$)	Divide and check fault test.
FXEM (code, message, no. of words)	Display of error trace.
FXOPT (code, word settings)	Alter word settings.
LINK (link name)	Restore link and transfer to its entry point.
LLINK (link name)	Restore link and return to next statement in calling subroutine.
MEMSIZ (integer variable)	Memory allocated.
PTIME (real variable)	Processor time.
PDUMP (area$_i$,..., dump format)	BCD } Dump designated area of memory in specified
PDUMPA (area$_i$,..., dump format)	ASCII } format; return.
OVERFL (integer variable)	Exponent register overflow test.
RANSIZ (file, format, size)	Specify record size of random file.
SETBUF (file, 1st buffer, 2nd buffer)	Define buffer for file.
SETFCB (location, file$_1$,...)	Define space for file control block.
SETLGT (location, no. of words)	Define logical file table.
SLITE (0)	Clear sense lights.
SLITE (1 ≤ number ≤ 35)	Turn on sense lights.
SLITET (integer, 1 or 2)	Test and turn off sense lights.
SORT (array, no. of records, record size, 0 ≤ key$_1$ ≤ record size,...)	Sort in ascending order.
SORTD (array, no. of records, record size, 0 ≤ key$_1$ ≤ record size,...)	Sort in descending order.
SSWTCH (integer, 1 or 2)	Test sense switch.
TERMNO (2-char. variable)	Station code.
TERMTM (real variable)	Hours of log-on time.
TRACE	Time sharing debug and trace.
USRCOD (char. variable)	User identification.

SUPPLIED INTRINSIC FUNCTIONS

Intrinsic Function	Definition	No. of Arg.	Call Name	Type of: Arg.	Type of: Funct.
Absolute Value	$|a|$	1	ABS	R	R
			IABS	I	I
			DABS	D	D
			CABS[b]	C	C
Truncation	Sign of a times largest integer ≤ $|a|$	1	AINT	R	R
			INT	R	I
			IDINT	D	I
Remaindering[a]	a_1 (mod a_2)	2	AMOD	R	R
			MOD	I	I
			DMOD[b]	D	D
Choosing Largest Value	Max $(a_1, a_2, ...)$	≥2	AMAX0	I	R
			AMAX1	R	R
			MAX0	I	I
			MAX1	R	I
			DMAX1	D	D
			MAX	I, R, D	I, R, D
Choosing Smallest Value	Min $(a_1, a_2, ...)$	≥2	AMIN0	I	R
			AMIN1	R	R
			MIN0	I	I
			MIN1	R	I
			DMIN1	D	D
			MIN	I, R, D	I, R, D
Float	Conversion from integer to real	1	FLOAT	I	R
Fix	Conversion from real to integer	1	IFIX	R	I
Transfer of Sign	Sign of a_2 times $|a_1|$	2	SIGN	R	R
			ISIGN	I	I
			DSIGN	D	D
Positive Difference	a_1 - Min (a_1, a_2)	2	DIM	R	R
			IDIM	I	I
			DDIM	D	D
Obtain Most Significant Part of Double Precision Argument		1	SNGL	D	R
Obtain Real Part of Complex Argument		1	REAL	C	R
Obtain Imaginary Part of Complex Argument		1	AIMAG	C	R
Express Single Precision Argument in Double Precision Form		1	DBLE	R	D
Express Two Real Arguments in Complex Form	$a_1 + a_2 \sqrt{-1}$	2	CMPLX	R	C
Obtain Conjugate of a Complex Argument		1	CONJG	C	C

Intrinsic Function	Definition	No. of Arg.	Call Name	Type of: Arg.	Funct.
Logical "and"	$a_1 * a_2 * ...$	≥2	AND	R, I or T	T
Logical "or"	$a_1 + a_2 + ...$	≥2	OR	R, I or T	T
Logical "exclusive or"	$a_1 \oplus a_2 \oplus ...$	≥2	XOR	R, I or T	T
Ignore Type		1	BOOL	Any except L	T
Extract Bit Field	Beginning with bit a_1 of word a_3 extract a_2 bits	3	FLD	I, I, any except L	T
Logical One's Complement	-a	1	COMPL	R, I T	T

R - Real	C - Complex
I - Integer	T - Typeless
D - Double	L - Logical

[a] Remaindering (MOD (a_1, a_2)) is defined as $a_1 - [a_1/a_2]*a_2$, where the bracketed expression denotes the integer results of the expression a_1/a_2.

[b] These functions are processed by external library subroutines.

SUPPLIED FUNCTION SUBPROGRAMS, MATHEMATICAL

Function	Definition	No. of Arg.	Generic Name	Type of: Arg.	Function
Exponential	e^a	1	EXP	R	R
		1	DEXP	D	D
		1	CEXP	C	C
Natural Logarithm	$\log_e(a)$	1	ALOG	R	R
		1	DLOG	D	D
		1	CLOG	C	C
Common Logarithm	$\log_{10}(a)$	1	ALOG10	R	R
		1	DLOG10	D	D
Trigonometric Sine	$\sin(a)$	1	SIN	R	R
		1	DSIN	D	D
		1	CSIN	C	C
Trigonometric Cosine	$\cos(a)$	1	COS	R	R
		1	DCOS	D	D
		1	CCOS	C	C
Hyperbolic Tangent	$\tanh(a)$	1	TANH	R	R
Square Root	$(a)^{1/2}$	1	SQRT	R	R
		1	DSQRT	D	D
		1	CSQRT	C	C
Arctangent	$\tan^{-1}(a)$	1	ATAN	R	R
		1	DATAN	D	D
	$\tan^{-1}(a_1/a_2)$	2	ATAN2	R	R
		2	DATAN2	D	D
Arcsine	$\sin^{-1}(a)$	1	ARSIN	R	R
Arccos	$\cos^{-1}(a)$	1	ARCOS	R	R

R - Real
D - Double
C - Complex

SUPPLIED FUNCTION SUBPROGRAMS, NONMATHEMATICAL

Function	Usage	No. of Args.	Type of: Arg.	Type of: Function
Left Shift	ILS (i, j)	2	I	I
Right Shift	IRS (i, j)	2	I	I
Left Rotate	ILR (i, j)	2	I	I
Right Logical	IRL (i, j)	2	I	I
Set Switch Word	ISETSW (i)	1	T	I
Reset Switch Word	IRETSW (i)	1	T	I
Mode	MODE (i)	1	I	I
Compare	KOMPCH (char., integer, char., integer, integer)	5	C, I	I
Random Number Generator	RAND (range)	1	R	R
	RANDT (range)	1	R	R
	FLAT (seed)	1	R	R
	UNIFM2 (seed, mean, width)	3	R	R

I - Integer
T - Typeless
C - Character
R - Real

ERROR CODES

Error Code	Default Procedure Abort/ Continue	Function	Error
0	A	Not used	
1	C	I**J	I=0, J=0
2	C	I**J	I=0, J<0
3	C	DA**J / A**J	DA=0, J=0 / A=0, J=0
4	C	A**J / DA**J	A=0, J<0 / DA=0, J<0
5	C	B**C	B<0, C≠0
6	C	A**B	A=0, B=0
7	C	A**C	A=0, C<0
8	C	e**B	B>88.028
9	C	LOG(A)	A=0
10	C	LOG(B)	B<0
11	C	ARCTAN(A/B)	A=0, B=0
12	C	SIN(A) / COS(A)	$|A| > 2^{27}$
13	C	$\sqrt{B}$	B<0
14	C	CA**K	CA=0, K=0
15	C	CA**J	CA=0, J<0
16	C	DA**DB	DB≠0, DA<0
17	C	DA**DB	DA=0, DB=0
18	C	DA**DB	DA=0, DB<0
19	C	e**DA	DA>88.028
20	C	LOG(DA)	DA=0
21	C	LOG(DA)	DA<0
22	C	$\sqrt{DA}$	DA<0
23	C	SIN(DA) / COS(DA)	$|DA| > 2^{54}$
24	C	ARCTAN (DA/DB)	DA=0, DB=0
25	C	CA/CB	CB=(0, 0)
26	C	e**CA	REAL CA>88.028
27	C	e**CA	$|$IMAG CA$| > 2^{27}$
28	C	LOG(CA)	CA=(0, 0)

Error Code	Default Procedure Abort/Continue	Function	Error
29	C	$\{SIN(CA)\}/\{COS(CA)\}$	$\lvert REAL(CA)\rvert>2^{27}$
30	C	$\{COS(CA)\}/\{SIN(CA)\}$	$\lvert IMAG(CA)\rvert>88.028$
31	C	BCD I/O	Illegal format statement
32	C	BCD I/O	Illegal character in data or bad format
33	A	BCD I/O	Attempt to read output file
34	A	BCD I/O	End-of-file
35	C	Rewind and end file processor	Illegal request
36	C	FFFB	Backspace error
37	A	File opening	File not defined
38	A	File opening	No space for I/O buffers
39	A	Binary I/O	Illegal end-of-file
40	C	Binary I/O	List exceeds logical record length
41	A	Binary I/O	Sysout/fixed length records
42	C	Namelist Input	Illegal heading card
43	C	Namelist Input	Illegal variable name
44	C	Namelist Input	Illegal subscript or array size exceeded
45	C	Namelist Input	Illegal character after right parenthesis
46	C	Namelist Input	Illegal character in data
47	A	Backspace record	File cannot be backspaced
48	C	Namelist Input	Illegal logical constant
49	A	Backspace File	Erroneous end-of-file

Error Code	Default Procedure Abort/Continue	Function	Error
50	C	Backspace file	Block count of zero
51	C	Sense light simulator	Index not $0 \leqslant n \leqslant 35$
52	C	Namelist Input	Illegal Hollerith field
53	C	Sense switch test	Index not $1 \leqslant n \leqslant 6$
54	A	File opening	Attempt to write I*
55	A	FXEM	Namelist input
56	A	File opening	Attempt to read P*
57	C	BCD I/O	Illegal character for L conversion
58	C	Backspace record	
59	C	Namelist Input	Empty Hollerith field
60	C	I**J	I**J>2**35 $\lvert I\rvert>1$, J>35 J IS EVEN I<-1, J>35, J IS ODD
61 62 63 64 65 66	A		Reserved for users
67	C	Fault	Exponent underflow
68	C	Fault	Integer overflow
69	C	Fault	Exponent overflow
70	C	Fault	Integer divide check
71	C	Fault	Floating point divide check
72	C	Random I/O	List exceeds logical record length
73	A	Random I/O	File not standard system format. Zero block count; BSN error; zero record count

Error Code	Default Procedure Abort/ Continue	Function	Error
74	A	Random I/O	No device for file
75	A	Random I/O	Bad record reference
76	A	Random I/O	Record size not specified in FCB. Give via $ FFILE card or call RANSIZ (FC, size)
77	A	Random I/O	Random I/O to linked file illegal
78	A	Random I/O	The record number given in the Random Read or Write statement is outside the file limits.
79	A	Random I/O	List exceeds declared record length
80	A	Random I/O	File is not large enough to contain record
81	C	Format I/O Encode/ Decode	Line exceeds size of receiving field
82	A	Format I/O Encode/ Decode	First non-blank character is not (
83	C	Arcsine	$|Arg| > 1.0$
84	C	Format I/O Encode/ Decode	$|Integer| > 2**35-1$
85	C	I/O	"GFRC" error
86	A	Format I/O Encode/ Decode	Encode/Decode-I/O may not be used recursively
87	C	I/O	Space/core obtained
88-143			Not presently used

NOTATION:
I, J, K are integers
A, B, C are real numbers
DA, DB, DC are double-precision numbers
CA, CB, CC where CA=X, Y are complex numbers

TIME SHARING BASED ERROR MESSAGES

FILE AND RECORD CONTROL TYPE ERRORS

1. GET CODE 5 – File Code

    Record size is zero in record control word.

2. PUT CODE 4 – File Code

    Current logical record larger than buffer.

3. CLOSE CODE 3 – File Code

    File to be closed is not in chain.

4. GET CODE 4 — File Code

    Block serial number error.

5. FILE SPACE EXHAUSTED – File Code

    Attempts to "grow" file denied by Time Sharing System.

6. BACK/FORWARDSPACE ERROR – File Code

    Bad status returned on DRL FILSP

COMPILER ABORT

COMPILER ABORTING

This message is printed at terminal, followed by DRL ABORT. The compiler abort code is stored into slave prefix cell 0.

RUN COMMAND ERROR MESSAGES

<61> LAST RUN COMMAND NOT PROCESSED

"RUN" not first three characters of input.

CONCATENATION IMPOSSIBLE IF RANDOM

RUN "random file;" random file illegal.

LINE NO. INTERVAL ILLEGAL IF NOT ASCII

Line number interval specified for other than type 5 or 6 ASCII.

NOT IN RECOGNIZABLE FORMAT

Input file specified is not legal as compiler or loader input.

MULTIPLE ALTER FILES NOT PERMITTED

Only one alter file (A*) permitted.

SAVE FILE(S) CANNOT BE SPECIFIED

"RUN HSTAR =; save file" is illegal.

ILLEGAL DELIMITER IMMEDIATELY FOLLOWING "="

Delimiter not semicolon, comma, left parenthesis, pound sign, or carriage return.

MUST BE RANDOM TO SAVE H*

RUN fs = fh, where fh is not a random file.

MUST BE LINKED TO SAVE C*

RUN fs = fh; fc, where fc is not a linked file.

ILLEGAL OPTION -- xxxx

Compiler/loader option indicated by xxxx is illegal.

ILLEGAL DELIMITER FOLLOWING RUN OPTION "xxxx"

Delimiter must be comma or right parenthesis.

ILLEGAL NAME = SPECIFICATION

Illegal character in name in NAME = option.

USER LIBRARIES EXPECTED

ULIB option specified but no user libraries specified.

USER LIBRARIES NOT EXPECTED

ULIB option not specified but user libraries designated.

TOO MANY USER LIBRARIES SPECIFIED

Maximum of nine user libraries can be specified.

TOO MANY TTY FILE CODES

Maximum of ten terminal file codes can be specified.

LOGICAL FILE CODE NONNUMERIC OR > 43

FORTRAN file codes can range from 1-43.

TOO MANY FILES REQ'D FOR EXECUTION

Maximum of 20 files can be specified.

TEST FILE HAS NOT BEEN ACCESSED

TEST option specified but appropriate ** test compiler has not been accessed.

066 - SPAWN UNSUCCESSFUL - STATUS n

Unsuccessful status returned from derail TASK, where n is equal to

1 - undefined file
2 - no SNUMB available
3 - duplicate SNUMB
4 - no program number available
5 - activity name undefined
6 - illegal user limit (time, size, etc.)
7 - bad status on *J read or write

CANNOT LOCATE MAIN PROGRAM IN LOAD FILE

Name of main program cannot be found in catalog block of H* file.

<50> WORK FILE - FILE TABLE FULL

Attempt to define temporary work file (B*, R*, J*, etc.) has failed; AFT is full.

<50> WORK FILE - SYSTEM TEMP. LOADED

System refuses to allocate temporary work file through derail DEFIL.

Catalog file string errors - (xxxx = file name):

ILLEGAL DELIMITER IN FIELD FOLLOWING xxxx DESCRIPTION

ILLEGAL CHARACTER IN FIELD FOLLOWING xxxx DESCRIPTION

STRING ELEMENT TOO LONG IN FIELD FOLLOWING xxxx DESCRIPTION

ILLEGAL PERMISSIONS IN FIELD FOLLOWING xxxx DESCRIPTION

ALTNAME ILLEGAL IN FIELD FOLLOWING xxxx DESCRIPTION

FILE DESCRIPTION TOO LONG IN FIELD FOLLOWING xxxx DESCRIPTION

NO DATA IN STRING IN FIELD FOLLOWING xxxx DESCRIPTION

File access errors:

  <50>   FILE xxxx - STATUS nn

  <50>   FILE xxxx - I/O ERROR

  <50>   FILE xxxx - NO PERMISSION

  <50>   FILE xxxx - FILE BUSY

<50>   FILE xxxx – NONEXISTENT FILE

<50>   FILE xxxx – NO FILE SPACE

<50>   FILE xxxx – INVALID PASSWORD

<50>   FILE xxxx – FILE TABLE FULL

<50>   FILE xxxx – SYSTEM LOADED

<50>   FILE xxxx – ILLEGAL CHARACTER

Reading and writing I/O errors:

<51>   FILE xxxx – I/O STATUS nn

<51>   WORK FILE – I/O STATUS nn

where nn is status code returned from derail DIO.

RUNL COMMAND ERROR MESSAGES

FILE NAME MUST BE OBJECT DECK (C*) FILE

File specified not an object deck file. If no C*'s specified left of equals sign, message is:
*SRC MUST BE OBJECT DECK

INCORRECT LINK PHRASE IN RUNL COMMAND

For example: Link(, B) or Link(A,)
Link(A, B,) or Link (B, C)
Link (A, ,) or Link(, B,)
Link ( )

INCORRECT SYNTAX FOR RUNL COMMAND

Generally, an illegal delimiter specified.

H* SAVE FILE NOT SPECIFIED

H* save file must be specified to right of equals sign.

ILLEGAL CHAR(S) IN LINK NAME

Characters must be alphabetic, numeric, and dash.

TOO MANY CHARS IN LINK NAME

More than six characters in link identifier.

028 – READ LINKED FILES ONLY WITH THIS COMMAND

Message appears when "PSTR" load map file is random; it must be linked.

SAVE FILE(S) CANNOT BE SPECIFIED

Message appears when H* save file appears to left of equals sign.

M6 – CALL/RSTR CHECKSUM

Message appears when H* save file is not large enough (in current size) to contain bound link/overlay structure.

ADDRESS OUTSIZE OF FILE LIMITS

Message appears when H* save file is not large enough (in current size) to contain bound link/overlay structure and attempt is made to "RUN" file.

DIAGNOSTIC MESSAGES ISSUED BY TIME SHARING LOADER

Messages are prefixed by either W for warning or F for fatal. Majority of errors diagnosed as warnings because user has the ability to hit the break key at any time. Thus, the decision is left to the user to continue or stop.

XXXXXX UNDEFINED

Symbol (XXXXXX) is an undefined SYMREF. DRL ABORT is substituted for all references.

XXXXXX LOADED PREVIOUSLY

SYMDEF (XXXXXX) previously defined in load table.

INCONSISTENT PREFACE FIELD (Deck) (Card)

One of two conditions occur on card number (card) in deck number (deck). The conditions are: (1) a SYMREF (type 5) appears with a non-zero size field (bits 0-17) in the preface card; or, (2) a LABELED COMMON (type 6) appears with a zero size field (bits 0-17).

LABELED COMMON XXXXXX – SIZE INCONSISTENT

LABELED COMMON (XXXXXX) defined previously with smaller size. Loading continues using original size.

ILLEGAL CHECKSUM (Deck) (Card)

Checksum on card number (Card) of deck (Deck) does not compare when recalculated. Loading continues.

ILLEGAL BINARY CARD (Deck) (Card)

Card number (Card) of deck (Deck) is not either preface (type 4), binary (type 5), or BCD (type 6). Card is ignored. Message may also appear where a preface or binary card appears out of expected order.

COMMON SIZE INCONSISTENT (Deck) (Card)

Blank common already defined. A subsequent deck is encountered having a larger blank common region specified. The deck is ignored and loading continues.

ILLEGAL LOAD ADDRESS (Deck) (Card)

A calculated storage address falls outside loadable store. The deck is ignored but loading continues.

XXXXXX LOADED PREVIOUSLY, LABELED COMMON ILLEGAL

SYMDEF (XXXXXX) already defined. XXXXXX appearing in current preface record is a Labeled Common. Deck is ignored.

IC MODIFICATION NOT POSSIBLE

Field requested IC modification ($code). In this case, no other modifiers are allowed. Bits 30-35 of the constructed instruction are checked and found to be non-zero.

XXXXXX UNDEFINED LINK ID IS YYYYYY

Where XXXXXX is an object symbol (SYMDEF) name and YYYYYY is a link identifier. Meaning XXXXXX is an unresolved SYMREF within the bounds of overlay YYYYYY.

XXXXXX UNDEFINED LINK ID

Link identifier XXXXXX is being used to define an origin point for the next overlay. It has as yet been undefined.

XXXXXX NOT LINK ID

Symbol XXXXXX appearing here as a link identifier has been used and entered into the load table previously as another type symbol.

LINK ID XXXXXX USED PREVIOUSLY

The identifier, XXXXXX, for the upcoming overlay has been previously entered in the load table as a link identifier.

The following diagnostics are preceded by a printout of the record in error and are generally associated with OCTAL correction processing.

NON-OCTAL DIGIT IN LOCATION FIELD

Self-explanatory.

FIELD EXCEEDS 12 DIGITS

Twelve octal digits is maximum allowed in word.

ILLEGAL TERMINATOR

Octal field is modified incorrectly. Check syntax rules in the General Loader manual.

FATAL DIAGNOSTICS

EOF READING BINARY (Deck) (Card)

Unexpected EOF while reading binary, identification of last record read is supplied.

ENTRY NOT FOUND

Primary entry name (...... or first primary SYMDEF) was not found in load tables. Diagnostic may also appear when subroutine .SETU. is not found.

H* TOO SMALL

File specified as save file (H*) not large enough to hold program.

REQUEST FOR MORE STORE TO EXPAND LOAD TABLE - DENIED

A request for 1K to be added at the upper address end of the load table was denied by the system. Loading terminates. Suggest user rerun job.

REQUEST FOR MORE STORE TO EXPAND PROGRAM - DENIED

A request to expand memory size for object program denied by the system. Suggest user rerun job.

ILLEGAL STATUS WHILE READING (File)

Only status accepted other than EOF is ready.

BLOCK SERIAL ERROR READING (File)

Block number in file (File) does not agree with expected number.

LIBRARY SEARCH TABLE EXCEEDED

Table used to collect pointers into random library has been exceeded. Table size is arbitrarily set at 200.

REQUEST FOR MORE STORE TO EXPAND LOAD TABLE - DENIED

Addmen request denied. Probable need for increasing TSS memory size.

COMPILER ABORT CODES

NOTE: The abort code Y1 always is displayed as the reason code for any abort. The panel reveals the specific reason code (see codes in parenthesis of following descriptions) in the upper 18 bits of the Q-register.

Y1 (X1)   Compiler space management module has unsuccessfully attempted to allocate contiguous memory block for internal table. Rerun with DUMP option and $ SYSOUT card for file code *F. Return dump to Honeywell Field Support – PCO.

Y1 (X2)   Compiler has attempted to execute request for additional memory more than 10 consecutive times (initial memory plus maximum of 30K). Increase allocation via $ LIMITS card or via "CORE=" option on TSS RUN.

Y1 (X3)   GCOS has denied compiler request for additional memory for internal tables. Increase allocation via $ LIMITS card or via "CORE=" option on TSS RUN.

Y1 (03)   Expression being handled has tree structure depth greater than 64. Expression must be divided.

Y1 (04)   Rerun with DUMP option and $ SYSOUT card for file code *F. Return dump to Honeywell Field Support – PCO.

Y1 (P4)   Unrecoverable error occurred in code generator; error message will print following source statement causing abort. Rerun with DUMP option and $ SYSOUT card for file code *F. Return dump to Honeywell Field Support – PCO.

SYSTEM ABORT CODES

LK   No $ ENTRY card for this link.

Q1   Logical Unit Table overflow – too many file codes.

Q2   Missing Logical Unit Table – supply $ OPTION FORTRAN card.

Q3   No space for Logical Unit 6 Buffer – increase limits.

Q4   Machine error or unexpected error to FORTRAN compiler.

Q5   FXEM told to take an alternate return but an alternate return name was not supplied.

Q6   Termination of object program execution via FXEM.

ASCII/BCD CHARACTER SET

ASCII CHAR	OCTAL	BCD CHAR	OCTAL	MODEL 33/35 KEY	HOLLERITH CARD
NULL	000	---	---	'CS'P	---
SOH	001	---	---	'C'A	---
STX	002	---	---	'C'B	---
ETX	003	---	---	'C'C (EOM)	---
EOT	004	---	---	'C'D (EOT)	---
ENQ	005	---	---	'C'E (WRU)	---
ACK	006	---	---	'C'F (RU)	---
BEL	007	---	---	'C'G (BELL)	---
BS	010	---	---	'C'H	---
HT	011	---	---	'C'I (TAB)	---
LF	012	---	---	LINE FEED	---
VT	013	---	---	'C'K (VT)	---
FF	014	---	---	'C'L (FORM)	---
CR	015	---	---	RETURN	---
SO	016	---	---	'C'N	---
SI	017	---	---	'C'Ø	---
DLE	020	---	---	'C'P	---
DC1	021	---	---	'C'Q (X-ON)	---
DC2	022	---	---	'C'R (TAPE)	---
DC3	023	---	---	'C'S (X-OFF)	---
DC4	024	---	---	'C'T (TAPE)	---
NAK	025	---	---	'C'U	---
SYN	026	---	---	'C'V	---
ETB	027	---	---	'C'W	---
CAN	030	---	---	'C'X	---
EM	031	---	---	'C'Y	---
SS	032	---	---	'C'Z	---
ESC	033	---	---	'CS'K	---
FS	034	---	---	'CS'L	---
GS	035	---	---	'CS'M	---
RS	036	---	---	'CS'N	---
US	037	---	---	'CS'Ø	---
SP	040	blank	20	SPACE BAR	blank
!	041	!	77	'S'1	0-7-8
"	042	"	76	'S'2	0-6-8
#	043	#	13	'S'3	3-8
$	044	$	53	'S'4	11-3-8
%	045	%	74	'S'5	0-4-8
&	046	&	32	'S'6	12
'	047	'	57	'S'7	11-7-8
(	050	(	35	'S'8	12-5-8
)	051	)	55	'S'9	11-5-8
*	052	*	54	'S':	11-4-8
+	053	+	60	'S';	12-0
,	054	,	73	,	0-3-8
-	055	-	52	-	11
.	056	.	33	.	12-3-8
/	057	/	61	/	0-1
0	060	0	00	0	0
1	061	1	01	1	1
2	062	2	02	2	2
3	063	3	03	3	3
4	064	4	04	4	4
5	065	5	05	5	5
6	066	6	06	6	6
7	067	7	07	7	7
8	070	8	10	8	8
9	071	9	11	9	9
:	072	:	15	:	5-8
;	073	;	56	;	11-6-8
<	074	<	36	'S',	12-6-8
=	075	=	75	'S'-	0-5-8

ASCII CHAR	OCTAL	BCD CHAR	OCTAL	MODEL 33/35 KEY	HOLLERITH CARD
>	076	>	16	'S'.	6-8
?	077	?	17	'S'/	7-8
@	100	@	14	'S'P	4-8
A	101	A	21	A	12-1
B	102	B	22	B	12-2
C	103	C	23	C	12-3
D	104	D	24	D	12-4
E	105	E	25	E	12-5
F	106	F	26	F	12-6
G	107	G	27	G	12-7
H	110	H	30	H	12-8
I	111	I	31	I	12-9
J	112	J	41	J	11-1
K	113	K	42	K	11-2
L	114	L	43	L	11-3
M	115	M	44	M	11-4
N	116	N	45	N	11-5
O	117	Ø	46	Ø	11-6
P	120	P	47	P	11-7
Q	121	Q	50	Q	11-8
R	122	R	51	R	11-9
S	123	S	62	S	0-2
T	124	T	63	T	0-3
U	125	U	64	U	0-4
V	126	V	65	V	0-5
W	127	W	66	W	0-6
X	130	X	67	X	0-7
Y	131	Y	70	Y	0-8
Z	132	Z	71	Z	0-9
[	133	[	12	'S'K	2-8
\	134	\	37	'S'L	12-7-8
]	135	]	34	'S'M	12-4-8
^	136	^	40	'S'N	11-0
_	137	←	72	'S'Ø	0-2-8
	140	---	---	---	---
a	141	---	---	---	---
b	142	---	---	---	---
c	143	---	---	---	---
d	144	---	---	---	---
e	145	---	---	---	---
f	146	---	---	---	---
g	147	---	---	---	---
h	150	---	---	---	---
i	151	---	---	---	---
j	152	---	---	---	---
k	153	---	---	---	---
l	154	---	---	---	---
m	155	---	---	---	---
n	156	---	---	---	---
o	157	---	---	---	---
p	160	---	---	---	---
q	161	---	---	---	---
r	162	---	---	---	---
s	163	---	---	---	---
t	164	---	---	---	---
u	165	---	---	---	---
v	166	---	---	---	---
w	167	---	---	---	---
x	170	---	---	---	---
y	171	---	---	---	---
z	172	---	---	---	---
{	173	---	---	---	---
|	174	---	---	---	---
}	175	---	---	---	---
~	176	---	---	---	---
DEL	177	---	---	RUBOUT	12-7-9

Legend:

    'C' = CTRL key
    'CS' = CTRL and SHIFT keys
    'S' = SHIFT key

Honeywell