

HP 9000 BASIC Language System

hp HEWLETT
PACKARD

A Technical Supplement* for the
HP 9000 Model 20 Computers

* Data Subject to change.



HP Computer Museum
www.hpmuseum.net

For research and education purposes only.

Introduction

Note: This BASIC Language System is single-user only and will operate only on the HP 9000 Model 20.

The HP 9000 BASIC Language System features new software technology that can provide a unique programming environment for the engineer who needs the performance of a compiler but prefers the friendly interaction of an interpretive language. The key to providing such a system is a run-time compiler that adds the functionality of an interpreter to the high-speed performance of a compiler.

HP BASIC is conversational and easy-to-use and it incorporates standard ANSI BASIC statements. But it also extends far beyond ANSI BASIC to give you the powerful features normally associated with such languages as FORTRAN, PL/1 and Pascal.

HP 9000 BASIC Language System Key Features

- Single-user, Multi-programming
- Event-driven Task and Program Scheduling
- Interactive Editor
- Engineering Extensions

The HP 9000 BASIC Language System provides complex features in an easy-to-use form. It offers multi-tasking, high-performance graphics and extended I/O which are common to minicomputer operating systems. HP 9000 BASIC has made these features accessible with clear, logical commands which are as easy to use as the rest of the BASIC language. But with the HP 9000, you can access these commands from a program or directly from the keyboard without writing a program. The result is the ability to isolate and accelerate testing of complex machine functions which are traditionally difficult to access and require extra technical proficiency to use.

The BASIC Language System is characterized by its friendly program development tools. The system features a full screen editor operated by special keys on the keyboard. It also offers a collection of commands that allow the programmer to:

- trace the flow of program control or values of variables,
- pause programs and modify variables or instructions from the keyboard or
- step through programs line-by-line.

Functionality

The tools and extensions of the HP 9000 BASIC system make it ideal for the engineer who has problems that change frequently. The conversant, descriptive syntax of the BASIC language plus the HP 9000 debug and development tools let users get programs written and running correctly in a shorter period of time than with traditional systems. Consequently, the HP 9000 BASIC system is an excellent tool for automating the engineering laboratory.

Multi-programming

HP 9000 BASIC is a memory-based, multi-programming system. The number of programs it can support concurrently depends on the amount of physical memory available (typically, 20 programs on a 2 Mbyte system). Each program runs in its own "partition" which is a virtual machine. With simple keyboard commands, you can attach the system keyboard and CRT to any partition.

HP 9000 BASIC supports two kinds of priorities: CPU allocation and program task control. The first one allocates the CPU to various programs according to 5 priority levels. If programs are of equal priority, CPU time is allocated by time-slicing. EVENT semaphores are available to allow synchronization between programs. A separate capability can suspend execution of programs while waiting for I/O. Information can be shared via the file system.

There is also a special provision that allows a program to have a higher priority than inputs from the keyboard. This provides immediate response to hardware interrupts.

Task Scheduling and Interrupt Handling

The second kind of priority supported on the BASIC system controls the execution of tasks within a program. Control of task execution can be achieved from the keyboard, from another program or by an external event. Here are some examples:

- a peripheral interrupt,
- a signal from another program,
- depressing a special function key,
- depressing any key on the keyboard,
- a time of day,
- expiration of a timer,
- detection of an end-of-record, end-of-file or end-of-transfer from a mass storage device or
- an error condition.

Compilation and Memory Management

On many other computer systems, the user must choose between interpreted or compiled systems. Through the innovation of Hewlett-Packard's new BASIC Run-Time Compiler, the HP 9000 can maintain the highly interactive program development environment of an interpreter and still deliver the performance of a compiled language.

When the user runs a program in source form, line-by-line compilation of the code begins. It is important to note that as the code is compiled, it is executed and both the source and object code remain in memory. This is significant for two reasons. First, because the object code is stored, it only needs to be compiled once. For example, if a program iterates a loop 1000 times, the first



time through the loop it is compiled and executed. The last 999 times through, only the object code is executed. This results in significant performance gains over an interpreted system. Second, because the BASIC source code remains in memory, all the BASIC debug and development capabilities (ie. trace, pause, edit, step and examine/alter variables) are available to the programmer. The result is a programming environment with the development tools of an interpreter and the performance of a compiled language.

After program development is complete, the programmer can choose to store only the compiled object code on mass storage. He can then load only the compiled code and execute it. This uses significantly less memory during execution.

The BASIC Language System offers excellent flexibility in managing the HP 9000 memory. A memory partition can be as small as 48 Kbytes or as large as the physical memory permits. (Note: the BASIC system does not support Virtual Memory). With this system, the programmer can rely on the operating system to dynamically allocate memory among programs or specify a memory allocation for his program's exclusive use during program execution.

Screens

Included in the HP 9000 System is the ability to "section" the CRT into rectangular "screens". Each screen can be attached to various system functions such as PRINT, INPUT, error messages, etc. Overlapping screens are also allowed.

Screens may be either "public" or "private". Private screens are only visible when the CRT is attached to the defining partition. Public screens are always displayed.

File Systems

HP 9000 BASIC offers a flexible mass storage system. It supports three different directory formats: a hierarchical file format called Structured Directory Format (SDF), Logical Interchange Format (LIF) and 9845 Desktop Computer Format.

The SDF format is BASIC's default system and is best suited for managing large, mass storage media. SDF allows multiple directories to organize the file system in a hierarchical format. In addition, SDF permits passwords to be assigned to each file for READ, WRITE and MANAGER permissions. This improves file security and reduces the administrative problems associated with users sharing a large mass storage device.

The LIF format is a standard for transporting ASCII source and data among HP computer products on removeable mass storage media. LIF is also a convenient means of transporting information between the HP 9826/36 Desktop Computers and various models of the

HP 1000 and HP 3000 computer families. Please consult your local Hewlett-Packard Sales Office for more specific details on data transfer between HP computers.

The HP 9000 supports HP 9835/45 DATA files. Conversion between BCD and binary numeric representations occurs automatically. This simplifies the task of writing programs designed to share data between the HP 9000 and HP 9835/45.

Software Transportability

HP 9000 BASIC was designed to provide a growth path from the HP 9845 and 9826/36 Desktop Computers. Programs from the HP 9826/36 can be transported via LIF ASCII files. Programs from HP 9835/45 can be transported via 9845 DATA files. Users will enjoy an excellent level of compatibility with the arithmetic and control features of the HP Enhanced BASIC Language. Although there are some minor differences in the functionality of language extensions, these differences can be translated. In the case of the 9845, where language differences are more apparent, a utility (HP Part No. 09020-10000) is provided to assist with the transporting of code.

Language Characteristics

Note: For a complete listing of commands, statements and operators, see Appendix A.

Variable Types

INTEGER	16-bit integer representation
DOUBLE	32-bit integer representation
REAL	64-bit IEEE binary floating point format
SHORT	32-bit IEEE binary floating point format
STRING	ASCII character strings (32-bit length field)

Arrays

Utilize any variable type
512 Kbytes maximum array allocation (except blank common and buffer variables)
Up to 6 dimensions

Structured Programming Features

LOOP; EXIT IF; END LOOP; FOR; NEXT;
IF; THEN; ELSE; END IF
REPEAT; UNTIL; SELECT;
CASE; CASE ELSE; END SELECT;
WHILE; END WHILE
FORCE DECLARE ON/OFF – disable/enable implicit variable declarations

Local Language Support

Roman Extension and Katakana character sets
Extended characters in character strings, line identifiers, file names and variable names
Local keyboards and lexical string sorting algorithms for French, German, Katakana, Spanish, and Swedish

BASIC System Extensions

3D Graphics (HP 97052A)



Note: No graphics capability (2D or 3D) is included with the BASIC Language System. To access any graphics features, the 3D Graphics System (97052A) must be installed.

The BASIC 3D Graphics System provides high performance graphics for the engineer. The key to this performance is how quickly the memory-based BASIC language can execute graphic statements on the Model 20's intelligent color and monochromatic displays. With BASIC graphics, you can choose between two dimensional or three dimensional transformation algorithms to easily convert the coordinates of a programmer's application into a two dimensional projection.

The HP 9000 BASIC 3D Graphic System feature set includes a comprehensive set of clipping and viewing functions. Additional features are:

- Ability to use data arrays for storing and plotting graphics.
- Parallel or perspective projections.
- Full clipping capability.
- All Model 20 displays support simultaneous and independent handling of graphics and alphanumerics to improve user interaction.
- Besides the standard text font, HP 9000 BASIC lets the programmer design his own text fonts. A typical font is output at a rate of 550 char./sec. This font can then be viewed and rotated like any other 3D object.
- The HP 9000 supports a full array of graphic input and output devices and one set of unified statements to handle these devices. This feature provides easy I/O redirection and plotting to multiple devices simultaneously.

The ability to store objects in data arrays distinguishes the HP 9000 BASIC system as a tool for object-oriented graphics. This tool gives the programmer the ability to create objects quickly and easily.

Scale, rotate and translate modeling transformations are provided to allow "instancing" and nested definitions of graphics objects. Once the object has been created, BASIC lets you pivot it, view it from any angle, or even look at it from the inside out. The IMAGE Data Base Management's ability to store data arrays combines nicely with the graphics system to build graphics data bases.

With the BASIC 3D Graphics package you can expect the following performance:

	9020B High-performance Monochromatic	9020C High-performance Color
2 D DRAW	3100 v/sec.	3100 v/sec.
3 D DRAW	2250 v/sec.	2250 v/sec.
1/4" octagon area fills/sec.	63	40
Circles/sec. (60 sides, unfilled)	59	59
2 D Matrix plot No pen control Pen control	5600 v/sec. 4150 v/sec.	5600 v/sec. 4150 v/sec.
3 D Matrix plot No pen control Pen control	3300 v/sec. 2680 v/sec.	3300 v/sec. 2680 v/sec.
2 D Absolute Matrix plot 1/2" vectors 1" vectors	9400-9600 v/sec. 7200-7400 v/sec.	9400-9600 v/sec. 7200-7400 v/sec.

Note: Rates assume 2 in. vectors or less.

IMAGE/QUERY Data Base Management (HP 97053A)

For data base management, HP 9000 BASIC supports the same IMAGE/QUERY Data Base Management package that has become popular with the HP 3000, HP 1000, HP 250 and HP 9845 computer families. An IMAGE data base consists of two levels of data objects and uses a path-oriented or chained approach to data retrieval. The first level is comprised of master data sets which define and store a search or key item to access data in the master data set or a detail data set. Detailed data sets are the second level of objects and are used strictly to store data.

A user may define the structure of a data base using either a schema processor or the DEFINE section of QUERY.

QUERY is a BASIC software package that puts a friendly, interactive interface between the user and the data base. It is suitable for the computer non-sophisticate or a user who wants to access a data base without having to write a special program to do so.

Both the HP-UX Operating System and BASIC Language System support IMAGE/9000. You should note, however, that QUERY/9000 is currently available only with the BASIC system.

Specifications

Data Bases

Data sets/data base:	63
Max. number of volumes:	20
Max. data base size:	Limited by media

Data Sets

Number of entries/data set:	$2^{31} - 1$ constrained only by physical volume. Note: Data sets cannot span volumes.
-----------------------------	--

Detailed data sets/master key:	31
Keys/detail data set:	31

Data Items

Items/data base:	1023
Items/data entry:	1023

Item Types

I2	2 byte integer	R8	8 byte floating point number
I4	4 byte integer	Xnn	nn byte character string
R4	4 byte floating point number	C2	Code (enumerated type)

IMAGE Features

IMAGE offers statements to create and maintain data bases as well as access a data base for information update and retrieval.

IMAGE's DBOPEN command lets the programmer choose between an exclusive or shared mode of operation. Exclusive mode permits the user to request exclusive access to the whole data base. Shared mode permits the user to gain access to a list of specified data sets through a locking mechanism. Shared mode is useful when you want multiple programs in separate partitions to access a common data base concurrently.

IMAGE is able to limit user access to the data base and data sets by the use of passwords. There are 13 passwords available for assignment to each data base. Each password can be assigned to multiple data sets and users may gain access to the data sets if he knows the password associated with that set. Once a user has gained access to a data set, he has access to all data items within that set.

To retrieve data, IMAGE supports four classes of access methods:

Serial – A sequential progression forward or backward through a data set.

Direct – Immediate access to any user specified record number.

Calculated – IMAGE supports a hashing algorithm that allows a record number to be calculated directly from the key item value, rather than through search.

Chained – A progression that uses linked lists to step forward or backward through a data set.

An enhancement to the chained access method is the "sorted chain." This feature sorts chains as the data base is updated with either the standard ASCII or any of the local language lexical sort algorithms supported by the BASIC language.

One of IMAGE's key features is the simplicity and power of the programmer interface. The following data base definition and design tools highlight this simple interface.

Codes – This lets the user declare his own item type and specify a set of strings that describe the range of values allowed. For example, an item called "color" could have the range red, blue or green.

Ranges – You can define a minimum, maximum, and default value for any numeric item. The values are available to your program via a DBINFO request.

Pseudonyms – This feature lets you assign up to four additional names for one data item. The values are available to your program via the DBINFO command.

Utilities – Included with IMAGE is a set of utility commands to help manage the data base. These utility commands can create and initialize data sets, purge data sets, erase all or part of a data set or extend the capacity of a data set. QUERY includes a utility library that helps you copy the data base to a back-up mass storage device, recover a corrupt data base or define a data base. A utility that is of interest to HP 9845 users is the Import/Export utility which will help transport 9845 data bases from the 9845 to an HP 9000.

Data Base Design Kit – Contains step-by-step procedures to help you define a data base.

QUERY Features

QUERY is useful for the following functions:

Defining a data base. QUERY will lead the user step-by-step via a set of prompts through the procedure of defining and creating a data base. QUERY is also a powerful tool for redefining and editing the definition of a data base.

Updating data bases. This is a quick and convenient way to add new entries, or delete or modify existing entries

of a data base. With this tool, range-checking of data is always performed.

Searching for selected information. QUERY has some powerful searching and selecting capabilities that are helpful for retrieving data from a single data set or a combination of multiple data sets. Selection criteria include all the standard numerical operators plus partial string matching (WITH, STARTING WITH, WITH FIRST NAME, WITH LAST NAME, IS, IS NOT, and null values). Multi-layered searches (ie. search for X, within X, search for Y, within X Y, search for Z) are also supported. For searching across data sets for desired data, these tools can be invaluable.

QUERY also supports a facility for showing the data base structure in a graphical form, the ability to create and use data entry forms and an interface to user subprograms.

Extended I/O

To complement Hewlett-Packard's wide line of instrumentation, HP 9000 BASIC offers an extensive range of I/O capabilities. The BASIC language fully supports the HP 9000 hardware's ability to offer DMA and vectored interrupts on every I/O channel. In addition, the BASIC language offers the following:

- One set of I/O statements that can interface to all devices (from printers to voltmeters, files, buffers and data communication ports. By changing one line of source code, you can easily re-direct I/O to any of the mentioned destinations.
- A complete set of control statements for the IEEE-488 Standard Interface (HP-IB).
- High speed transfers between devices and buffers. These buffers are circular for continuous transfers and can support both input and output transfers simultaneously.
- Extensive branching facilities enable a direct branch or a call to a program segment to be initiated by an End Of File, End Of Transfer or any external device interrupt.
- A complete set of image specifiers for formatted I/O. Each operates symmetrically on input and output.

This BASIC I/O capability provides a responsive, flexible solution that is easily programmed. The combination of powerful statements and programmability make the HP 9000 BASIC Language System an ideal tool for laboratory automation.

Asynchronous Terminal Emulator (HP 97056A)

The HP 9000 Asynchronous Terminal Emulator is intended as a general purpose tool for transporting data across a wide variety of mainframes. To run the emulator, you must have the HP 27128A Asynchronous Interface Card. The emulator features the following specifications:

- Line or character mode.
- DC1/DC3 and Enq/Ack handshakes.
- 110 to 19200 baud rate.
- Provides most features of the HP 2621 terminal.
- Easily configured from a menu with special function keys.
- Upload and download ASCII files.
- With BASIC multi-programming, the emulator can run as a background task (e.g., you can download a file from a host computer while other programs run in separate partitions).

BASIC System Support

Memory Requirements

The BASIC Language System requires 512 Kbytes of memory. This includes Advanced Programming, BASIC Compiler, SDF directory format, internal and external hard disc drivers, an HP-IB driver, self test code, operating system code and dynamically allocated memory.

In addition, the following binaries and separate products may be individually loaded:

Options:	# of bytes	Part No.
I/O	59,800	
Mass Storage	15,384	
LIF directory format	8,824	
Memory disc driver	1,504	
9845 directory format	10,376	97050A
Descriptive error messages	9,520	
Flexible Disc driver	12,264	
HP 9885 driver	4,004	
GP-IO driver	2,024	
Serial interface driver	13,676	
HP 7970 tape driver (requires I/O)	5,320	

Separate Products:

3D Graphics	89,840	
CRT driver	17,636	97052A
IMAGE DBM	63,252	97053A

Therefore, to accommodate minimum user applications will require at least 768 Kbytes (3 memory finstrate boards) of memory.

Supported Peripherals

External Mass Storage Devices

9895A	8-inch Flexible Disc Drive
7908/11/12P	Fixed Disc Drives
7933H	404 Mbyte Fixed Disc Drive
82901M	5¼-inch Flexible Disc Drive
7971A	9-track Magnetic Tape Drive

Graphic Input Devices

9111A	Graphics Tablet
-------	-----------------

Graphic Plotters

7470A*	2-pen, A size
9872C/T	8-pen, B size
7580A	8-pen, D size
7585A	8-pen, E size

External Printers

2631B/G	Dot Matrix Printer
---------	--------------------

97098A I/O Expander

BASIC Training

9826/36 BASIC Language Operating & Programming Class (5 days)

HP 9000 BASIC Extensions (1 day) HP Prod. No. 97005A

Taught as a seminar extension to the HP 9826/36

BASIC Language Operating and Programming Class.

(Note: 9826/36 BASIC class or equivalent knowledge is a pre-requisite for this class.)

Ordering Information

Required

97050A	BASIC Language System
--------	-----------------------

Optional:

97052A	3D Graphics
97053A	IMAGE/QUERY DBM
97056A	Asynchronous Terminal Emulator (BASIC)

Distribution media for above software:

Opt. 042	On 5¼" Flexible Disc
----------	----------------------

Manuals

	Part Number
BASIC Language Reference	97050-90005
BASIC Programming Techniques	97050-90000
BASIC Graphics Programming Techniques	97052-90000
BASIC Asynchronous Terminal Emulator Manual	97056-90000
IMAGE Data Base Programming Techniques	97053-90000
QUERY/9000 Users Guide	98053-90001
Data Base Design Kit	97053-90002

Support Services

Software Notification Service (SNS) – Software Status Bulletin plus a subscription to Communicator Magazine.

Manual Update Service (SSS) – SNS, MUS (above) plus software updates and problem reporting.

System Information Service (SIS) – SSS (above) plus

Phone-In Consulting Service (PICS)[†] for two callers.

General Consulting Service – On site Systems Engineer assistance on a daily basis.

Customer Support Service (CSS) – SSS, PICS, assigned account-responsible HP Service Engineer and on-site assistance.

* Requires 9/82 plotter firmware update. Consult local HP sales office for more details.

[†] Pre-requisites for any services with PICS are the appropriate customer courses or equivalent knowledge as determined by the local HP Sales Office.

Appendix A

HP Enhanced BASIC Commands, Statements and Operators

Program Development

!	Begins a comment.
CHANGE ... TO	Replaces one string with another.
COMPILE	Compiles and stores a program, subprogram or multi-line function.
COPYLINES	Duplicates and relocates program lines.
DEF FN... =	Defines single-line function.
DEF FN	Defines function subprogram.
RETURN	Exits function, returning its value.
FNEND	Defines end of function.
DEL	Deletes program lines from memory.
DEL SUB...	Deletes subprogram(s) or multi-line function(s) or the MAIN from memory.
EDIT	Enters program editor at specified line.
EDIT KEY	Edits the definition of an SFK.
END	Marks end of main program. Implied STOP.
FIND	Finds specified string, and enters program editor.
FORCE DECLARE ON/OFF	Forces declaration of all variables.
INDENT	Indents the program lines.
LET	Begins assignment statement.
LIST	Lists program lines.
LIST KEY	Lists SFK definitions.
MOVELINES	Relocates program lines.
OPTION BASE	Defines array defaults.
REM	Begins comment line.
REN	Renumbers program lines.
SCRATCH	Clears program and variables not in common.
SCRATCH A	Clears programs and variables in all partitions.
SCRATCH KEY	Clears SFK definition.
SCRATCH P	Clears program in current partition.
SPACE DEPENDENT ON/OFF	Requires proper spacing in program entry.
SUB	Begins a subprogram.
SUBEXIT	Transfers execution to line after CALL.
SUBEND	Last line of a SUB subprogram. Implied SUBEXIT.

Program Execution & Debug

CONT	Resumes execution of paused program.
DEFAULT ON/OFF	Enables/disables default results on math errors.
ERRDS	Returns I/O error select code.
ERRL	Returns error state of specified line.
ERRMS	Returns error message.
ERRN	Returns error number.
FORCE DECLARE ON/OFF	Requires declaration of all variables.
ON/OFF ERROR	Branch to service routine on error.
PAUSE	Suspends program execution.
PRINTALL IS	Defines where PRT ALL messages are sent.
PRT ALL	Log all messages/results.
RUN	Starts execution at specified line.
STEP	Executes a single program line.
STOP	Terminates a program.
TRACE	Traces the program flow between specified lines.
TRACE ALL	TRACE plus TRACE ALL VARIABLES.
TRACE ALL VARIABLES	Traces all variables.
TRACE OFF	Terminates tracing.

TRACE PAUSE
TRACE VARIABLES
TRACE WAIT
XREF

Sets program break points.
Traces specific variables.
Causes a wait on each trace message.
Cross reference of variable usage.



Storage Allocation Statements

ALLOCATE
ASSIGN @...
COM

Dynamic storage allocation statement.
Opens/closes named I/O path to BUFFER.
Reserves memory for variables, BUFFERS, and I/O path name accessed by multiple contexts.

DATA
DEALLOCATE
DIM
DOUBLE
INTEGER
OPTION BASE
REAL
REDIM
RESET
SCRATCH C
SCRATCH V
SHORT

Defines data for the READ and RESTORE statements.
Releases memory reserved by ALLOCATE.
Reserves memory for local variables, BUFFERS, and I/O path names.
Dimensions and reserves memory for local DOUBLE variables and arrays.
Dimensions and reserves memory for local INTEGER variables and arrays.
Defines default base of array elements.
Dimensions and reserves memory for local REAL variables and arrays.
Redimensions arrays.
Resets BUFFER fill and empty pointers.
Clears all local and COM variables.
Clears all local variables.
Dimensions and reserves memory for local SHORT variables and arrays.

General Math

*
+
-
/
^
ABS
DIV
DROUND
MINREAL
EXP
FLT
FRACT
MAXREAL
INT
LET
LGT
LOG
MAX
MIN
MOD
MODULUS
PROUND
RANDOMIZE
RND
RES
SGN
SQR

Multiplication operator
Addition operator
Subtraction Operator
REAL Division Operator
Exponentiation Operator.
Absolute Value function.
Returns integer portion of a quotient.
Returns numeric argument rounded to a digit count.
Returns the smallest REAL number available.
Returns e to a specified power.
Returns an argument as a REAL number.
Returns the fractional part of an argument.
Returns the largest REAL number available.
Returns greatest integer of less than or equal to argument.
Assigns value of its expression to a variable.
Returns base 10 logarithm of an argument.
Returns base-e (ln) logarithm of an argument.
Returns the largest number in a list.
Returns the smallest number in a list.
Operator, remainder of an integer division.
Operator, remainder of a truncated division operation.
Returns the numeric expression rounded to the specified power of ten.
Modifies the seed specified in the RND function.
Returns a pseudo-random number.
Returns the result of the last numeric computation executed from the keyboard.
Returns the sign of an argument.
Returns the square root of an argument.

Trigonometric Statements and Functions

ACS
ASN
ATN
COS
DEG
GRAD
PI
RAD
SIN
TAN

Returns the arccosine of its argument.
Returns the arcsine of its argument.
Returns the arctangent of its argument.
Returns the cosine of its argument.
Sets trig mode to degrees mode.
Sets trig mode to grads.
Returns the value 3.1415926535897931 .
Sets trig mode to radians.
Returns the sine of its argument.
Returns the tangent of its argument.

Relational Operators

<
<=
<>

Less Than.
Less Than or Equal.
Inequality.

=	Equality.
>	Greater Than.
>=	Greater Than or Equal.

Logical Operators and Functions

AND	Returns logical AND of two arguments.
BINAND	Returns 32-bit logical AND of two arguments.
BINCMPL	Returns 32-bit logical complement of its argument.
BINEOR	Returns 32-bit logical Exclusive OR of two arguments.
BINIOR	Returns 32-bit logical Inclusive OR of two arguments.
BIT	Returns state of a bit.
DROTATE	Returns 32-bit value after binary bit rotation.
DSHIFT	Returns 32-bit value after binary bit shift.
EXOR	Returns logical Exclusive OR of two arguments.
NOT	Returns logical complement of its argument.
OR	Returns logical OR of two arguments.
ROTATE	16-bit version of DROTATE.
SHIFT	16-bit version of DSHIFT.

Matrix/Array Statements and Functions

BASE	Returns the lower bound of a dimension.
DET	Returns the determinant of a matrix.
DOT	Returns the inner product of two vectors.
MAT	Performs a single matrix operation.
MAT CSUM	Returns the sum of the elements in each column.
MAT IDN	Returns the identity matrix.
MAT INV	Returns the inverse of a matrix.
MAT REORDER	Reorders an array.
MAT RSUM	Returns the sum of the elements in each row.
MAT SORT	Sorts an array.
MAT TRN	Returns the transpose of a matrix.
OPTION BASE	Defines the default base for arrays.
RANK	Returns the number of dimensions in an array.
SIZE	Returns the number of elements in a dimension.
SUM	Returns sum of array elements.

String Operators & Functions

&	Concatenates two strings.
BIT	Returns state of bit or result of argument matched to 32-bit mask.
CHR\$	Converts numeric expression to an ASCII character.
DVAL	Returns 32-bit value of binary, octal, decimal or hex string expression.
DVAL\$	Returns binary, octal, decimal or hex string version of 32-bit integer argument.
ERRM\$	Returns error message.
IVAL	16-bit version of DVAL.
IVAL\$	16-bit version of DVAL\$.
KBD\$	Returns keyboard input during ON KBD.
LEN	Returns the length of a string.
LEXICAL ORDER IS	Sets the lexical order.
LWC\$	Converts a string to all lower case letters.
NUM	Returns the numeric value of an ASCII character.
POS	Returns the position of one string within another.
REV\$	Reverses a strings order.
RPT\$	Repeats a string a specific number of times.
TRIM\$	Deletes blanks from the beginning and end of a string.
UPC\$	Converts a string to all upper case letters.
VAL	Returns numeric value of a decimal string.
VAL\$	Returns decimal string version of numeric expression.

Program Control Statements

CALL	Invokes subprogram, passing parameters.
CAUSE...	Simulates an ON... condition.
DISABLE	Disables branches defined by ON... statements.
ENABLE	Enables branches defined by ON... statements.
END	Marks end of main program. Implied STOP.
FN	Invokes DEF FN function.
RETURN <expression>	RETURNS to line after FN, passing a value.
FOR	Defines the beginning of a loop.
TO	Defines loop index termination value.

STEP	Defines loop index increment.
NEXT	Increments loop index.
LOOP	Defines a loop repeated until EXIT IF is true.
EXIT IF	Exits loop if its expression is true.
END LOOP	Defines end of LOOP.
GOSUB	Transfers execution to a subroutine.
RETURN	Transfers execution to line after GOSUB.
GOTO	Transfers execution to a program line.
IF	Specifies a conditional expression.
THEN	Begins lines executed if expression is true.
ELSE	Begins lines executed if expression is false.
END IF	Defines end of IF...THEN sequence.
NPAR	Returns number of parameters passed to subroutine, or multi-line function.
OFF...	Disables an ON... condition.
ON...	Performs CALL/GOSUB/GOTO/RECOVER upon ON... condition.
PAUSE	Suspends program execution.
REPEAT	Defines loop repeated until UNTIL expression true.
UNTIL	End of REPEAT loop.
SELECT	Specifies branch expression.
CASE	Executed if CASE value = expression.
CASE ELSE	Executed if no CASE value matched expression.
END SELECT	Defines end of SELECT statement sequence.
STOP	Terminates program execution.
WAIT	Suspends program until any ON... condition.
WAIT...	Suspends program for specified time.
WAIT FOR...	Suspends program until EOR, EOT, EVENT or INTR.
WHILE	Defines loop executed until expression true.
END WHILE	Defines end of WHILE loop.

ON.../OFF ... Statements

ON <pointer>	Indexed GOTO/GOSUB.
ON/OFF CYCLE	Enables/disables branches on a time interval.
ON/OFF DELAY	Enables/disables branch after time delay.
ON/OFF END	Enables/disables branch on file eor/EOF.
ON/OFF END#	Enables/disables branch on file eor/EOF.
ON/OFF EOR	Enables/disables branch on TRANSFER End Of Record.
ON/OFF EOT	Enables/disables branch on End Of TRANSFER.
ON/OFF ERROR	Enables/disables branch on BASIC error.
ON/OFF EVENT	Enables/disables branch on CAUSE EVENT.
ON/OFF GKEY	Enables/disables branch on Graphics Key stroke.
ON/OFF INTR	Enables/disables branch on interface interrupt.
ON/OFF KBD	Enables/disables branch on keyboard keystroke.
ON/OFF KEY	Enables/disables branch on SFK keystroke.
ON/OFF KEY ... LABEL	ON/OFF KEY, with display label definition.
ON/OFF SIGNAL	Enables/disables branch on SIGNAL.
ON/OFF TIME	Enables/disables branch at specific time.
ON/OFF TIMEDATE	Enables/disables branch at specific time & date.
ON/OFF TIMEOUT	Enables/disables branch on I/O timeout.

Mass Storage Statements

ASSIGN @...	Opens/closes file's I/O path and I/O attributes.
ASSIGN #	Opens/closes file's number.
CAT	Lists the contents of a directory or a PROG file's subprograms.
CAUSE END	Simulates End Of File.
CAUSE EOR	Simulates TRANSFER End Of Record.
CAUSE EOT	Simulates TRANSFER End Of Transfer.
CAUSE TIMEOUT	Simulates a time-out.
CHECKREAD	Causes verification of data as it is written to a volume.
CONTROL	Alters pointers within a file.
COPY	Copies a file to another file.
CREATE ASCII	Creates an ASCII file.
CREATE BCD	Creates a BCD file.
CREATE BDAT	Creates a BDAT file.
CREATE DATA	Creates a DATA file.
CREATE DIR	Creates a directory.
CREATE PROG	Creates a PROG file.
GET	Reads a file and attempts to store the strings in memory as program lines.

INITIALIZE	Initializes a mass storage medium.
IOLOCK	Returns number of LOCKs on a file.
LOAD	Loads a PROG file into memory.
LOAD BIN	Loads a BIN file into memory.
LOAD KEY	Loads an SFK file into the SFK's.
LOADSUB	Loads a subprogram into memory.
LOCK	Denies file access to other partitions.
MASS STORAGE IS	Specifies the default mass storage device or the current directory.
MSI	Short for MASS STORAGE IS.
ON/OFF END/EOR/EOT	Branch on occurrence of file condition.
PRINT #	Writes data into a file.
PRINT LABEL	Gives a volume a name.
PROTECT	Protects a capability on a file.
PURGE	Deletes a file.
READ #	Reads data from a file.
READ LABEL	Reads a volume name into a string variable.
RENAME	Renames and/or moves a file.
RE-SAVE	Saves program in a specified file.
RESET	Sets file pointer to record 1.
RE-STORE	Stores a program in a specified file.
RE-STORE BIN	Stores a binary in a specified file.
RE-STORE KEY	Stores key definitions in a specified file.
SAVE	Saves a program in a DATA file.
SAVE ASCII	Saves a program in an ASCII file.
STATUS	Returns information about a file.
STORE	Stores a program in a PROG file.
STORE BIN	Stores a binary in a BIN file.
STORE KEY	Stores key definitions in a DATA file.
STORE SUB	Stores a subprogram in a PROG file.
TYP	Returns the data type to be accessed next in a file.
UNLOCK	Releases LOCK on a file.
VDA	Short for VOLUME DEVICES ARE.
VOLUME DEVICES ARE	Defines default volumes.
WAIT FOR EOR/EOT	Suspends program until TRANSFER EOR/EOT.

Input/Output Statements & Functions

ABORTIO	Terminates overlapped I/O.
ASSIGN @...	Opens/closes named I/O path to device(s), file or BUFFER. Attributes are: BUFFERED ON/OFF, BUFFERSIZE, BYTE/WORD, CHECKREAD ON/OFF, CONVERT IN/OUT/OFF BY INDEX/PAIRS, DELIM, DRIVER, EOL END-DELAY/OFF, FORMAT ON/OFF, HOLDOFF, MAXBUFFS, OVERLAP/SERIAL, PARITY ZERO/ONE/ODD/EVEN/OFF, STOP/NONSTOP WIDTH-OFF, WRAP ON/OFF, XFERRATE.
BEEP	Sounds computer tone device.
BREAK	Sends serial datacomm Break sequence.
CAUSE EOR	Simulates TRANSFER End Of Record.
CAUSE EOT	Simulates TRANSFER End Of Transfer.
CAUSE INTR	Simulates an interrupt on an interface.
CAUSE TIMEOUT	Simulates a time-out.
CONTROL	Sends control information to an interface or I/O path name.
CRT	Returns select code of internal CRT.
DISABLE INTR	Disables ON INTR and WAIT FOR INTR.
DISP	Transfers data to DISPLAYS SCREEN.
DISP USING	Same as DISP, with user-specified formatting.
DUMP ALPHA	Copies ALPHA contents of CRT to output device.
EDIT...\$	Same as LINPUT, but displays current string value.
ENABLE INTR	Enables ON INTR statements.
ENTER	Transfers data from a source to program variables.
ENTERBIN	Inputs single binary value.
ENTER...USING	Same as ENTER, with user-specified formatting.
ERRDS	Returns device selector where last I/O error occurred.
FIXED	Sets the output mode of numbers to a fixed number of digits.
FLOAT	Sets the output mode of numbers to floating point.
IMAGE	Provides format specifiers for ...USING statements.
INPUT	Transfers keyboard input to program variables.
IOCOUNT	Returns number of bytes transferred.

IOSTAT	Returns value of a single STATUS register.
KBD	Returns select code of keyboard.
KBD\$	Returns keyboard input during ON KBD.
LINPUT	Transfers keyboard input directly to string variable.
OUTPUT	Transfers data from program variables.
OUTPUTBIN	Transfers binary items to interface or device.
OUTPUT...USING	Same as OUTPUT, with user-specified formatting.
OVERLAP	Enables I/O concurrent with program execution.
PRINT	Transfers data to PRINTER IS device.
PRINTALL IS	Defines device for error, TRACE and result messages.
PRINTER IS	Defines device for PRINT, CAT, and LIST.
PRINT USING	Same as PRINT, with user-specified formatting.
PRT	Returns select code of internal printer.
READ	Returns values from a DATA statement.
RESET	Resets an interface to its power-on state.
RESTORE	Resets pointer within DATA statements.
RESUME INTERACTIVE	Enables live keyboard.
SC	Returns select code assigned to an I/O path name.
SCREEN	Returns device selector of specified SCREEN.
SERIAL	Disables OVERLAP.
STANDARD	Sets the output mode of numbers to the default.
STATUS	Returns value of I/O register(s).
SUSPEND INTERACTIVE	Disables live keyboard.
TRANSFER	Transfers data between BUFFER and interface, device, or file.
WAIT FOR EOR/EOT	Suspends program until TRANSFER EOR/EOT.
WAIT FOR INTR	Suspends program until interface interrupt.

HP-IB Statements and Functions

ABORT	Aborts bus I/O. May take active control.
CLEAR	Resets bus devices to defined state.
LOCAL	Sets bus devices to their local state.
LOCAL LOCKOUT	Locks out front panel of bus devices.
PASS CONTROL	Passes active control to another bus device.
PPOLL	Returns parallel poll byte from bus.
PPOLL CONFIGURE	Programs device for PPOLL response.
PPOLL RESPONSE	Enables/disables computer's own PPOLL response.
PPOLL UNCONFIGURE	Disables device PPOLL response.
REMOTE	Sets bus device(s) to their remote state.
REQUEST	Defines computer's SPOLL response byte.
SEND	Sends messages and data to a bus.
SPOLL	Returns serial poll byte from a bus device.
TRIGGER	Sends trigger message to bus or device(s).

System & Partition Statements and Functions

ASSIGN...TO SCREEN	Associates system attributes with a SCREEN.
ATTACH	Associates a partition with the CRT and keyboard.
CAUSE EVENT	Increments an EVENT semaphore, and causes ON EVENT branches to occur.
CREATE EVENT	Creates an EVENT semaphore.
CREATE PARTITION	Creates a partition.
CREATE SCREEN	Creates a SCREEN.
DEL EVENT	Deletes an EVENT semaphore.
DEL PARTITION	Deletes a partition.
DEL SCREEN	Deletes a SCREEN.
EVENT LEVEL	Returns the value of an EVENT semaphore.
MOVE SCREEN	Relocates a SCREEN.
ON/OFF EVENT	Enables/disables branch on CAUSE EVENT.
ON/OFF SIGNAL	Enables/disables branch on SIGNAL.
PARTITION STATUS	Returns partition information and status.
RESET SCREENS	Resets the CRT to the default SCREENs.
SCRATCH P	Clears a partition.
SCREEN	Returns device selector of the specified SCREEN.
SIGNAL	Causes ON SIGNAL branches to occur.
SYSTEM PRIORITY	Sets the priority of the current context.
SYSTEM\$	Returns system configuration and status.
WAIT FOR EVENT	Decrements an EVENT semaphore, pausing the program if semaphore is negative.



Real Time Statements and Functions

DATE	Returns the Julian value of the specified date.
DATE\$	Returns a string representing the specified Julian date.
ON/OFF CYCLE	Executes a branch at specified time intervals.
ON/OFF DELAY	Executes a branch after a specified elapsed time.
ON/OFF TIME	Executes a branch at a specified time.
ON/OFF TIMEDATE	Executes a branch at a specified date and time.
SET TIME	Sets the time in the internal clock.
SET TIMEDATE	Sets the time and date in the internal clock.
TIME	Returns a number representing the specified time.
TIME\$	Returns a string representing the time specified.
TIMEDATE	Returns the time and date of the internal clock.
WAIT...	Suspends the program for a specified elapsed time.

Graphics Output Statements

APLOT	Outputs a plot to the internal CRT using displacements from the previous pen position.
APLOT	Outputs a plot to the internal CRT using specified coordinates.
ARPLOT	Outputs a plot to the internal CRT using displacements from a local origin.
AXES	Outputs axes with optional tick marks.
DRAW	Outputs a line to a specified point.
FRAME	Outputs a frame around the current clip area.
GRID	Outputs a full grid pattern.
GTEXT	Outputs object text.
IDRAW	Outputs a line using displacements from the pen's position.
IMOVE	Moves the pen using displacements from the pen's position.
IPLOT	Outputs a plot using displacements from the previous pen position.
LABEL	Outputs labels on the view surface.
LABEL USING	Outputs labels using an image specifier.
LAXES	Outputs a pair of labelled axes with optional tick marks.
LETTER	Starts interactive text generation mode.
LGRID	Outputs a labelled full pattern grid.
MOVE	Moves the pen to a specified point.
PENUP	Lifts the pen on all plotters.
PLOT	Outputs a plot using specified coordinates.
POLYGON	Outputs a closed, regular polygon.
POLYLINE	Outputs a regular polygon which may be open.
RECTANGLE	Outputs a rectangle.
RPLOT	Outputs a plot using displacements from a local origin.
SYMBOL	Outputs a user-defined symbol.
WHERE	Returns the pen's location and status information regarding the pen.

Graphics Attributes Statements

AREA COLOR	Determines the fill color.
AREA INTENSITY	Determines the fill color.
CSIZE	Determines the height, width, and slant of labels.
FONT...IS	Creates a user-defined font.
FONT IS	Enables a user-defined font.
FONT...IS TERMINATED	Disables a user-defined font.
GTEXT JUST	Determines the justification of gtext.
GTEXT ORIENT	Determines the orientation of gtext.
GTEXT SIZE	Determines the height, width, and slant of gtext.
LDIR	Determines the angle for drawing labels.
LINE TYPE	Determines the plotting line type used.
LORG	Determines the labeling location relative to the pen location.
PDIR	Determines the angle for plots with local origins.
PEN	Determines the pen color used.

Graphics Viewing Statements

ASKGU	Returns the UVW coordinates corresponding to a specified XYZ point.
ASKUU	Returns the XYZ coordinates corresponding to a specified UVW point.
CLIP	Defines and enables the soft clip boundaries.
MSCALE	Causes plotting to be output in metric units.
PIVOT	Rotates output relative to the coordinate system.
PROJECT	Determines the 3D projection used: parallel or perspective.
RATIO	Returns the ratio of the two sides of the logical view surface.
SETGU	Sets the UVW coordinate system as the current coordinate system.

SETUU	Sets the XYZ coordinate system as the current coordinate system.
SHOW	Causes an isotropic mapping of output to the viewport.
TRANSFORM	Defines a transformation of output to be concatenated to the viewing operation.
VIEW DEPTH	Defines the location of the front and back clipping planes.
VIEW DISTANCE	Defines the distance from the view plane to the view reference point.
VIEW NORMAL	Defines the orientation of the view plane relative to the view reference point.
VIEW POINT	Defines the location of the view reference point.
VIEWPORT	Defines the viewport.
VIEW UP	Defines the orientation of the coordinate systems.
WINDOW	Defines a window for output.
Graphics Control Statements	
ALPHA ON/OFF	Turns the alpha display on or off.
DIGITIZE	Returns the locator's coordinates on a graphics input device.
DUMP GRAPHICS	Copies the contents of the graphics display to a printing device.
GCLEAR	Clears the current clip area.
GESCAPE	Makes a request to a device driver.
GINIT	Initializes the graphics subsystem.
GKEY	Returns data which was saved by an ON GKEY button press.
GLOAD	Loads the graphics display using an INTEGER array.
GRAPHICS INPUT IS	Initializes a graphics input device.
GRAPHICS...IS ON/OFF	Enables or disables a graphics input device.
GRAPHICS INPUT...IS	
TERMINATED	Terminates the use of a graphics input device.
GRAPHICS ON/OFF	Turns the graphics display on or off.
GSTORE	Stores the graphics display using an INTEGER array.
LIMIT	Sets the hard clip boundaries on a graphics device.
MEMORY	Sets the color value of a memory plane.
ON/OFF GKEY	Enables or disables an end-of-line branch when a button on an enabled graphics input device is pressed.
PLOTTER IS	Initializes a plotter.
PLOTTER...IS ON/OFF	Enables or disables a plotter.
PLOTTER ... IS TERMINATED	Terminates the use of a plotter.
READ LOCATOR	Returns the current position of the locator on a graphics input device.
SET ECHO	Initializes a locator on a graphics input device.
SET LOCATOR	Sets the position of the locator on a graphics input device.
TRACK IS ON/OFF	Enables or disables a locator on a graphics input device.
Data Base Access Statements	
DBBUFFER	Re-defines I/O buffers.
DBCLOSE	Closes all or part of a data base.
DBCONTROL	Enables/disables performance options.
DBDELETE	Deletes a data entry in a data set.
DBFIND	Locates head of search item chain.
DBGET	Reads data values from a data entry.
DBINFO	Returns information about a data base.
DBLOCK	Locks data base, or one or more data sets.
DBOPEN	Initiates access to a data base.
DBPUT	Adds a data entry to a data set.
DBUNLOCK	Releases all or part of a DBLOCK.
DBUPDATE	Replaces data in an existing data entry.
PACKFMT	Specifies variables & skips for PACK/UNPACK.
PACK USING	Transfers variable data to packing string.
UNPACK USING	Transfers buffer data to program variables.
Data Base Utility Statements	
DBCREATE	Creates previously defined data set(s).
DBERASE	Deletes all entries in one or more data sets.
DBEXTEND	Increases the capacity of a data set.
DBPURGE	Purges data set file(s).
DBSTATUS	Outputs status report about a data base.

