Getting Started.

HEWLETT PACKARD

HP Series 200, Model 16

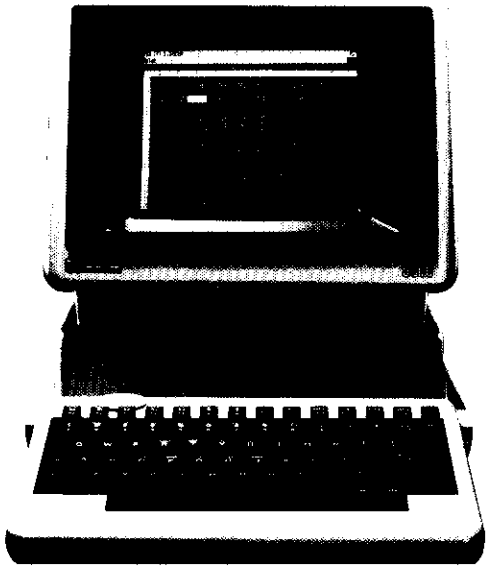
BASIC 2.0, BASIC Extensions 2.0 and Pascal Language Systems

Technical Data, November 1982

Introduction

Hewlett-Packard's Series 200, Model 16 is a compact and powerful, low-cost, personal technical computer. It is based on the Motorola MC68000 microprocessor, a 16-bit "computer on a chip" with 32-bit internal architecture that has become the industry standard for computation-intensive applications.

The Model 16 is designed to serve both the professional engineer and scientist as well as anyone who needs a personal computer with maximum computing power. It merges computer-aided engineering tools with practical business aids to easily turn "extra" work into "computer-aided" work.



Small in size, the Model 16 takes up as little space as the in-basket on your desk. Standing some 11.1 inches (282 mm) high, 12.4 inches (315 mm) wide, the Model 16 features a 9-in. (229 mm) CRT and small, detached ASCII keyboard.

Other Model 16 special features include:

- a unique rotary control knob for easy interaction;
- a large memory, ranging from 128K up to 768K bytes;
- built-in CRT graphics with a resolution of 300 x 400 pixels;
- built-in HP-IB (IEEE 488-1978) and serial interfaces;
- two built-in slots for a plug-in interface card and additional memory.

The Model 16 is available with two language systems – BASIC and Pascal.* Hewlett-Packard BASIC includes enhancements characteristic of more powerful languages like FORTRAN or ALGOL. HP BASIC includes subprograms, multi-dimensional arrays, unified I/O and mass storage, labelled COMMON blocks and external program control. HP BASIC 2.0 Extensions further enhances the already powerful HP BASIC 2.0 Language System. Among the features are advanced I/O capabilities, matrix operations, and mass storage and editing enhancements. HP Pascal gives you the high speed and protection from program modification of a compiled language and more. It offers extensive enhancements not found in other Pascal systems such as: powerful I/O and graphics procedure libraries with full peripheral support; extensive debugging capability; and access to MC68000 assembly language.

In addition, HP has numerous applications software packs available for the Model 16, including computer-aided engineering tools such as AC Circuit Analysis and Digital Filter Design; mathematics modules like Numerical Analysis and a comprehensive Statistical Library; and business aids like Project Management, Graphics Presentations and VisiCalc®.

The Model 16 can be linked to a variety of HP peripherals offering a complete, one-vendor system solution. A choice of new 3½-in. micro-flexible disc drives is available – either a single or dual "micro-floppy" disc unit, each providing 270K bytes of storage per disc. A single 3½-in. micro-floppy also is available with a 4.6-megabyte Winchester hard disc. Other system peripherals include a low-cost color graphics plotter and a dot matrix printer.

The Model 16 comes with two built-in interfaces: HP-IB (Hewlett-Packard's implementation of IEEE Standard 488-1978) and RS-232-C for serial communication with other computers and peripherals. Other interfaces available as plug-in cards include: HP-IB, RS-232-C, GPIO, BCD, Datacomm, and a color video monitor card.

Table of Contents

VisiCalc® is a registered trademark of VisiCorp., Inc.
* Pascal expected availability December, 1982.

BASIC 2.0 Capabilities

The BASIC 2.0 Language System is anything but basic. It adds to the inherent simplicity of BASIC the computational power usually found in FORTRAN, ALGOL and APL. Beyond that, high-performance, I/O-intensive constructs are incorporated, giving this BASIC the highest performance found in interactive instrument control systems on the market today.

BASIC 2.0 puts mini-computer architecture and power on *YOUR* desk, bench or test-bay, not down the hall. With the BASIC Language System, high performance and easy programming are not mutually exclusive – both are provided for your benefit. This increase in your software productivity is applicable to a wide spectrum of task and project demands because this BASIC provides an impressively rich command set at your fingertips. Sophisticated applications can be met with state-of-the-art hardware power as your resource. At the same time simple tasks can be solved in minutes, not hours.

BASIC 2.0 increases your software *PRODUCTIVITY* by providing:

- A computer system at your fingertips. You don't wait to "log-on" – just sit down and start developing solutions.
- Interactive editing. The editor is always there. A rotary control knob lets you quickly scroll through your program.
- Statement syntaxing at program entry. You don't have to run your program to find syntax errors; BASIC 2.0 proofreads your code.
- Extensive debug and trace tools. You can single-step your program one line at a time to check program logic flow and variable assignments. Tracing tools allow a line-by-line log of line numbers and variable changes to be printed while your software is running.
- Structured programming constructs that organize your code. Complex algorithms are easier to develop, document and maintain.
- Independent subprograms. You can break your software down into smaller modules for easier development and coding. Subprograms can be stored and loaded separately so you can develop disc-resident subprogram utility libraries.
- Dynamic variable allocation. This increases the flexibility of your subprogram libraries so you can efficiently manage memory space as you go.
- Labeled COM, to increase the flexibility of your subprograms. You can have several independent "common" variable blocks, allowing easy subprogram variable usage and subprogram communication.
- A unified device and mass storage I/O system. You can easily redirect I/O to and from devices and files. In essence, your I/O routines can access a mass-storage file in the same way as an external device providing an excellent test harness for your I/O dependent code.

BASIC 2.0 increases your software *PERFORMANCE* by providing:

- 16M bytes of address space including memory-mapped I/O with the Motorola MC68000. As applications grow, you won't outgrow your memory space or system performance. High-volume data acquisition can be done directly to memory.
- Intrinsic I/O drivers. BASIC 2.0 is optimized for I/O so you don't have to write or "link in" separate I/O drivers – they're already there.
- Optimized transactional I/O. I/O path set-up times are reduced to the bare minimum. Short data transfers are just that – short. I/O paths can be turned around quickly, allowing you to go from input to output extremely fast. BASIC 2.0 gives your I/O-dependent software unparalleled "agility."
- The richest HP-IB I/O command set in the industry. Instrument control over HP-IB is quick and to-the-point. More than 20 commands are dedicated to HP-IB I/O to give you performance and flexibility in letting the bus work for you.
- 15 levels of prioritized software interrupt. You can optimize I/O operations to closely match peripheral device speeds without tying down the system. With 15 levels of interrupts, a multitude of external events can be serviced with varying levels of priority set by you. Even the keyboard rotary control knob can be used as a system interrupt source in addition to the soft keys.

These are just a few reasons why the BASIC 2.0 Language System can increase your software performance and productivity. BASIC 2.0 gives you high performance *AND* easy software development, taking you from problem definition to solution quickly and efficiently. You generate results, not just software.

BASIC 2.0 Operating Characteristics

Language System Memory Requirements

ROM-based
1 ROM board AND 21K bytes RAM.

RAM-based
277K bytes RAM.

Range

Real Precision:
$-1.797693134862315E+308$ to $-2.225073858507202E-308$
0
$2.225073858507202E-308$ to $1.797693134862315E+308$

Integer Precision:
-32768 to $+32767$

Math Hierarchy

Highest priority: ( )
User defined and built in functions
↑
*, /, MOD, DIV
+, – (unary and binary + and –)
=, <>, <, >, <=, >=
NOT
AND
Lowest priority: OR, EXOR

Built-In Functions

Mathematical and trigonometric functions and operations are included in the following with typical execution times in milli-seconds.

Absolute (ABS)	0.041	integer 0.033
Integer quotent (DIV)	0.449	integer 0.075
Digit round (DROUND)	1.498	
e^X (EXP)	3.485	
Integer (INT)	0.090	
Ln (LOG)	3.706	
Log (LGT)	3.800	
Modulus (MOD)	0.801	integer 0.074
Random number (RND)	0.419	
Sign (SGN)	0.055	
Square root (SQR)	1.743	
Sine (SIN)	3.621	
Cosine (COS)	4.200	
Tangent (TAN)	4.200	
Arcsine (ASN)	5.009	
Arccosine (ACS)	5.004	
Arctangent (ATN)	3.398	
↑	7.155	
/	0.432	
*	0.302	integer 0.080
+	0.137	integer 0.069
−	0.161	integer 0.069
Logical Operators		
AND		
OR		
EXOR		
NOT		
Relational Operators		
=	Equal	
<	Less than	
>	Greater than	
<=	Less than or equal to	
>=	Greater than or equal to	
<>	Not equal to	
String Operator		
&	Concatenation	

BASIC 2.0 Keyword Summary

General Functions

ABS – returns the absolute value of its argument.
ACS – returns the principal value of the arccosine of its argument.
AND – returns the logical conjunction of its arguments.
ASN – returns the principal value of the arcsine of its argument.
ATN – returns the principal value of the arctangent of its argument.
BINAND – returns the value of a bit-by-bit AND of its arguments.
BINCMP – returns the value of the 1's complement of its argument.
BINEOR – returns the value of a bit-by-bit exclusive OR of its arguments.
BINIOR – returns the value of a bit-by-bit inclusive OR of its arguments.
BIT – returns an INTEGER representation of the contents of one bit of its argument.
COS – returns the cosine of its argument.
DIV – returns the integer portion of the quotient of the dividend divided by the divisor.
DROUND – returns the value of a numeric expression, rounded to the specified number of significant digits.
ERRL – returns a value of 1 if the most recent execution error occurred during the specified line.
ERRN – returns the number of the most recent program execution error.
EXOR – returns the exclusive disjunction of its arguments.
EXP – raises the base of the natural logarithm (Naperian e, =2.718281828459045..) to the power of the argument.
INT – returns the greatest integer which is less than or equal to the evaluated expression.
KNOBX – returns the net number of pulses generated by the rotary control knob since the last interrupt.
LGT – returns the logarithm (base 10) of its argument.
LOG – returns the natural logarithm (base e) of its argument.
MOD – returns the remainder after performing division of its arguments.
NOT – returns the logical complement of its argument.
NPAR – returns the number of parameters passed in the call to the currently executing subprogram or multi-line function.
OR – returns the logical disjunction of its arguments.
PI – returns an approximate REAL value for pi.
RND – returns a pseudo-random number which is greater than zero and less than one.
ROTATE – returns an INTEGER value representing the value obtained by creating a bit-string version of its argument and rotating the argument the number of bit positions specified.
SGN – returns an INTEGER value of one if the given expression is positive, zero if it equals zero, and −1 if it is negative.
SHIFT – returns an INTEGER value representing the value obtained when its argument is converted to a 16-bit string pattern and shifted the number of positions specified.
SIN – returns the sine of its argument.
SQR – returns the square root of its argument.
TAB – moves the print position to the specified column on the current printing device.
TABXY – moves the print position to the column and line specified on the internal CRT.
TAN – returns the tangent of its argument.
TIMEDATE – returns the current value of the real-time clock.

String Functions

CHR$ – converts a numeric value into a character byte.
LEN – returns an integer representing the current number of characters in a string expression.
NUM – converts the first character of the string expression to its equivalent decimal value ASCII character code.
POS – determines the position of a substring within a string.
VAL – converts a string expression into a numeric value.
VAL$ – returns a string which represents the value of its argument.

General Statements

ALLOCATE – dynamically reserves memory space for string variables, string arrays and numeric arrays during execution.
ALPHA OFF/ON – enables or disables the CRT alphanumeric area for viewing.
BEEP – outputs an audible tone with programmable frequency and duration.

CALL – transfers program execution to the specified subroutine subprogram and specifies values for the pass parameters.
CASE, CASE ELSE – structured sequence used with SELECT for selecting alternative actions depending on the result of a conditional test (see SELECT).
COM – dimensions and reserves space for simple and array variables in a special "common" memory area so more than one program segment can access the variables.
CONT – resumes program execution without pre-run initialization (see PAUSE).
DATA – allows numeric values or string literals to be defined and assigned with the READ statement.
DEALLOCATE – reclaims the memory space reserved for use by the ALLOCATE statement.
DEF FN – defines a multi-line function which returns either a single REAL value or a string value to a calling program segment.
DEG – sets degree mode for results and arguments of trigonometric functions and graphics rotational commands.
DEL – deletes all program lines within the range specified by the beginning and ending line identifiers.
DELSUB – deletes one or more subprograms or multiple line functions from memory.
DIM – dimensions and reserves memory for REAL numeric arrays, simple strings and string arrays.
DISABLE – disables all active ON (event) statements except ON ERROR, ON END and ON TIMEOUT.
DISP – causes the values of the print list to be sent to the display line on the CRT.
DISP USING – causes the values of the print list to be sent to the display line on the CRT according to the format specified by the image specifier.
DUMP ALPHA – transfers the contents of the CRT screen to the device currently specified by DUMP DEVICE IS.
EDIT – accesses the editor to enter a new program or for modification of program lines.
ELSE – is part of the IF...THEN...ELSE... END IF construct, provides an alternative action to be performed.
ENABLE – re-enables all ON (event) statements which were suspended by DISABLE.
END – marks the end of a main program segment.
END IF – marks the end of an IF...THEN...ELSE...END IF construct.
END LOOP – marks end of LOOP construct.
END SELECT – marks end of SELECT contruct.
END WHILE – marks end of WHILE construct.
EXIT – provides exit from a structured LOOP (see LOOP).
FN – is used to call a user-defined, multiple-line function subprogram.
FNEND – is the last statement in a multiple-line function subprogram.
FOR...NEXT – defines a loop which is repeated until the loop counter exceeds its final value.
GOSUB – transfers program control to the subroutine that begins at the specified line.
GOTO – transfers program execution to the specified line.
IF...THEN, IF...THEN...ELSE...END IF – provides conditional branching or execution of one or more statements when the specified condition is true.
INTEGER – dimensions and reserves memory for integer variables and arrays.
LET – assigns a value to a simple numeric variable or assigns a set of characters to a simple string variable.
LIST – causes the entire program, or lines specified within a range list to be output to the current PRINTER IS device.
LIST # – causes the entire program, or lines specified within a range list to be output to the specified printing device.
LOOP...EXIT – repeats statements in a structured loop as long as the EXIT expression is FALSE.
OFF/ON ERROR – disables/enables an event-controlled branch to occur whenever a trappable error occurs.
OFF/ON KEY – disables/enables an event-controlled branch to occur when an Special Function Key is pressed.
OFF/ON KNOB – disables/enables an event-controlled branch to occur every specified number of seconds if the rotary control knob has generated pulses since the last interrupt.
ON event GOTO/GOSUB/CALL/RECOVER – is used with ON ERROR, ON END, ON KEY, ON INTR, ON KNOB and ON TIMEOUT to cause a branch in program execution based on the specified GOTO, GOSUB, CALL or RECOVER statement. With RECOVER, the branch will occur regardless of the current program environment.
ON expression GOSUB – transfers program execution to one of several subroutines depending on the value of the expression.
ON expression GOTO – transfers program execution to one of several line identifiers depending on the value of the expression.
OPTION BASE – specifies the lower bound of subscript values for all array dimensions when the lower bound is not explicitly stated in the array declaration.
OPTIONAL – is used in formal parameter lists to declare which parameters are optional when passing to a subprogram or FN function.
PAUSE – halts program execution without altering the data or state information so program operation can be continued.
PRINT – prints the specified items in the print list to the current system printer.
PRINT USING – outputs the items in the list according to the format specified by the image specifier.
PRINTALL IS – assigns a destination printing device for output which is normally sent to the display line of the CRT.
PRINTER IS – specifies the current printing device for any PRINT, PRINT USING, CAT and LIST statements.
RANDOMIZE – is used to modify the seed used by the computer's random number generator.
RAD – sets radian mode for results and arguments of trigonometric functions and graphics rotational commands.
READ – assigns values to program variables in conjunction with the DATA statement.
REAL – reserves storage for full precision, floating point variables and arrays.
REM or ! – allows comments to be inserted into your program.
REN – renumbers the program line numbers.
REPEAT...UNTIL – this loop construct repeats the statements in a structured loop until the expression following UNTIL is true.
RESTORE – repositions the data pointer for a program segment.
RESUME INTERACTIVE – enables the EXECUTE, PAUSE, STOP, STEP, CLR I/O, and RESET keys after a SUSPEND INTERACTIVE statement.
RETURN – marks the end of a subroutine.
RETURN expression – the last executed statement of a multi-line function.
RUN – enables program execution at the specified line identifier.
SCRATCH – erases all or selected portions of memory, depending on the secondary keyword.

SELECT – this structured statement allows the execution of several different actions depending upon the result of a conditional test when used with CASE or CASE ELSE. SELECT must end with END SELECT.
SET TIME – sets the time-of-day given by the real-time clock.
SET TIMEDATE – sets the time-of-day and the date given by the real-time-clock.
STOP – terminates execution of the program.
SUB – is the first statement in a subroutine subprogram and specifies the formal parameter list for the subprogram.
SUBEND – is the last statement of a subroutine subprogram and transfers execution back to the calling program segment.
SUBEXIT – can be used within the body of a subroutine subprogram to transfer execution back to the calling program segment.
SUSPEND INTERACTIVE – disables EXECUTE, PAUSE, STOP, STEP, CLR I/O and RESET keys.
TRACE ALL – allows the tracing of program flow and variable assignments while a program is running.
TRACE OFF – disables all tracing activity enabled by TRACE ALL or TRACE PAUSE.
TRACE PAUSE – causes program execution to pause before executing the specified line.
UNTIL – used with REPEAT for looping (see REPEAT).
WAIT – nondestructively suspends program execution for an approximate amount of time in seconds.
WHILE – repeats a structured loop as long as its expression is true.

Mass Storage Statements

ASSIGN – links an @name to a device or file, providing a unified I/O structure to a program.
ASSIGN @name TO * – cancels an @name assignment and its attribute.
CAT, CAT TO – list the contents of a mass storage medium's directory.
COPY – provides the capability to copy single files or to BACKUP entire media.
CREATE ASCII – creates an ASCII type file on the mass storage device.
CREATE BDAT – creates files which hold binary data types.
GET – reads the specified ASCII file into the computer's program memory.
INITIALIZE – sets up directory for a new medium. Data previously on a disc is destroyed.
LOAD – retrieves and places in memory any program file (PROG) which was previously stored with the STORE statement.
LOAD BIN – retrieves and puts into memory a binary program file (BIN) which was previously stored with the STORE BIN statement.
LOADSUB ALL FROM – loads all of the BASIC subprograms in a program file (PROG) into the computer's memory.
MASS STORAGE IS – specifies the current system mass storage device. (See MSI below.)
MSI – can be used to abbreviate MASS STORAGE IS.
OFF END – deactivates the end-of-statement branch previously activated by an ON END statement.
ON END – enables an event-controlled branch to occur when an end-of-file condition occurs for a mass storage device.
PROTECT – establishes (or changes) the protect code used on non-ASCII files.
PURGE – deletes a file's entry from the mass storage medium's directory.
RE-SAVE – copies all, or part, of the program currently in the computer's memory into an ASCII file as source code strings.
RE-STORE – copies a BASIC program and all binary programs in the computer's memory onto a mass storage medium in internal format.
RE-STORE BIN – writes all binary programs currently in the computer to the specified file.
RENAME – changes the file's name on the mass storage medium's directory.
SAVE – creates an ASCII file and stores all, or part, of the BASIC program in the computer's memory into it as source code strings.
STORE – creates a PROG file and stores the entire BASIC program and all binary programs in computer memory into it in internal format.
STORE BIN – creates a BIN file and stores all of the binary programs currently in the computer's memory into it.

I/O Functions

KBD$ – returns the contents of the keyboard buffer.
PPOLL – returns a byte representing the 8 status-bit messages of those devices on the HP-IB capable of responding to a parallel poll.
READIO – allows the reading of either bytes or words from any interface register.
SPOLL – returns an integer whose low order byte contains the serial poll response from the addressed device.

I/O Statements

ABORT – resets the interface functions for an HP-IB interface.
ASSIGN – links an @name to a device or file, providing a unified I/O structure to a program.
ASSIGN @name TO * – cancels an @name assignment and its attribute.
CLEAR – allows the active controller to put HP-IB devices into a defined device-dependent state.
CMD – is used with the SEND statement to send numeric or string expressions over HP-IB with ATN true.
CONTROL – sends control information to an interface or to the internal table associated with an @name.
DATA – is used with the SEND statement to send numeric or string expressions over HP-IB with ATN false if the computer is the active controller and is addressed to talk.
DISABLE INTR – sends a word to the interrupt-enable register of the specified interface, disabling all interrupts from that interface.
ENABLE INTR – enables the specified interface to generate an interrupt which can cause end-of-statement branches. Also allows the setup of Powerfail interrupts.
ENTER, ENTER @name – is used to read data from a device, file or string and assign the values read to the variables in the list.
ENTER USING – is used to read data from a device, file or string and assign the values read to the variables in the list according to the specified IMAGE.
FORMAT OFF/ON – specifies whether data is to be interpreted as ASCII (ON) or Series 200 internal format (OFF).
IMAGE – is referenced by the USING clause of the PRINT, OUTPUT, DISP, LABEL and ENTER statements to provide formats for I/O operations.
INPUT – is used to assign keyboard input to program variables.
LINPUT – assigns characters entered from the keyboard to a string variable or substring.

LISTEN – is used with the SEND statement to specify one or more primary addresses.
LOCAL – returns all specified devices to their local state.
LOCAL LOCKOUT – sends the LLO (local lockout) message, preventing an operator from returning the device to local control by its front panel.
MTA – is used with the SEND statement to send the HP-IB interface card's talk address.
MLA – is used with the SEND statement to send the HP-IB interface card's listen address.
OFF/ON INTR – disables/enables an event-controlled branch to occur when an interface card (enabled by ENABLE INTR) requests an interrupt. Priority may also be specified. With the Powerfail option this statement allows branching to an interrupt service routine if power should fail.
OFF/ON KBD – disables/enables interrupt branching when a key is pressed.
OFF/ON TIMEOUT – disables/enables an event-controlled branch to occur when an I/O timeout occurs on the specified interface.
OUTPUT, OUTPUT @name – copies data from the variables in the output list to the specified destination.
OUTPUT USING – copies data from the variables in the output list to the specified destination according to the specified IMAGE.
PPOLL CONFIGURE – programs the logical sense and data bus line on which the specified device responds to a parallel poll.
PPOLL UNCONFIGURE – disables the parallel poll response of the specified devices.
REMOTE – places HP-IB devices having remote/local capabilities into the remote state of operation.
SEC – is used with the SEND statement to send secondary commands and addresses over HP-IB.
SEND – sends control information and data to an HP-IB interface.
STATUS – provides the status value from an interface register, or the internal table associated with an @name, into the specified numeric variables.
TALK – is used with the SEND statement to define which devices talk on the HP-IB.
TRIGGER – initiates device-dependent action from either a selected device or all devices addressed to listen on the HP-IB.
UNL – is used with the SEND command to send the bus unlisten message (UNL) on the HP-IB.
UNT – is used with the SEND command to send the bus untalk message (UNT) on the HP-IB.
USING – used by the PRINT, OUTPUT, DISP, LABEL and ENTER statements, provides formats for I/O operations.
WRITEIO – writes either bytes or words to any interface register.

Graphics Functions

RATIO – returns a value equal to the ratio of the physical dimensions of the graphic device's hard clip limits.

Graphics Statements

AXES – draws a pair of axes with optional equally spaced tick marks.
CLIP – redefines the soft clip area.
CLIP ON/OFF – specifies whether the current clipping area is the soft clip area (ON) or the hard clip area (OFF).
CSIZE – sets the size and aspect ratio for labelled characters.
DRAW – draws a line from the pen's current position to the specified X,Y coordinate position using the current line type and pen number.
DUMP GRAPHICS – copies the contents of graphics memory onto the device currently specified by DUMP DEVICE IS.
DUMP DEVICE IS [EXPANDED] – specifies which device receives the data when DUMP GRAPHICS or DUMP ALPHA is executed. EXPANDED allows a 2 for 1 expansion along each axis, and rotates the resulting image 90°.
FRAME – draws a frame around the current clipping area using the current pen number and line type.
GCLEAR – clears the plotter's background.
GLOAD – allows you to load the contents of an integer array into graphics R/W memory for display.
GINIT – resets all global graphics parameters to their power-on values.
GRAPHICS ON/OFF – turns the CRT graphics raster on/off.
GRID – draws a full grid pattern.
GSTORE – allows the copying of graphics R/W memory contents in coded form into an integer array.
IDRAW – draws to a position specified as an increment to the current position of the pen.
IMOVE – moves to a position specified as an increment to the current position of the pen.
IPLOT – incremental plot is similar to IDRAW and IMOVE. The pen control is determined by a value following the X and Y coordinates.
LABEL – directs text and the contents of variables to the current plotting device.
LABEL USING – directs text and the contents of variables to the current plotting device according to the specified IMAGE.
LDIR – determines the angle at which labeling statements draw the characters.
LINE TYPE – selects a line type and repeat length for lines, labels, frames, axes and grids.
LOCATE – syntaxes to VIEWPORT.
LORG – specifies the position of the characters being labelled relative to the current pen position.
MOVE – updates the current position of the pen to the specified X,Y coordinate position.
PEN – selects the pen used by the plotter. PEN –1 erases lines on the CRT.
PENUP – lifts the physical pen from the plotting surface.
PIVOT – specifies a rotation of axes which is applied to all lines drawn by DRAW and IDRAW statements.
PLOT – moves or draws to the specified X,Y coordinate. Pen control is specified following the X,Y coordinates.
PLOTTER IS – selects a plotter to receive the plotting statements.
RPLOT – relative plot moves or draws to the X,Y position relative to the last absolute pen position. Pen control follows the X,Y values.
SCALE – syntaxes to WINDOW.
SHOW – isotropically defines the plotting units mapped on the VIEWPORT area.
VIEWPORT – specifies an area on which a scale specified by the WINDOW, and SHOW statements, are mapped.
WINDOW – specifies the minimum and maximum values for the plotting area specified by VIEWPORT.

BASIC Extensions 2.0 Capabilities

BASIC Extensions 2.0 includes over 190 keyword additions or extensions to enhance the already powerful BASIC 2.0 Language System to make Series 200 BASIC the most powerful desktop language in the computer industry. The command set for BASIC Extensions 2.0 was designed to contribute to the features already available in other HP desktop language sets. These features include advanced I/O such as DMA, fast handshake and interrupt buffer transfers, matrix operations and Mass storage enhancements.

BASIC Extensions 2.0 provides two binary programs: Advanced Programming and the Color Video Output. RAM or ROM-based BASIC 2.0 must be resident in the computer prior to loading these binaries. The binary programs can be loaded separately or stacked together to provide optimum memory utilization. This means that if you are only using the matrix statements, you only need the Advanced Programming binary loaded.

Features of BASIC Extensions 2.0

The features provided by the BASIC Extensions can be divided into these catagories:

- Entry and Editing Enhancements
- Debugging Extensions
- Matrix Operations
- String Utilities
- Timer Routines and Event Controls
- IO Enhancements including DMA and Fast Handshake
- Buffered IO Capabilities
- Formatting Enhancements
- Mass Storage Enhancements
- Callable Pascal or Assembly Language Subroutines
- BCD (98623A) Interface Support
- Color Video Graphics Extensions for 98627A Interface

Extensions Memory Requirements

Advanced Programming	170K bytes RAM
Color Video Output	21K bytes RAM

Advanced Programming Binary Keyword Summary

General Functions

CRT – returns the select code for the internal CRT.
ERRDS – Error Device Status – returns the address of the I/O resource involved in the most recent error.
ERRL – Error Line – extended for TRANSFER.
ERRN – Error Number – returns the number of most recent error; extended for TRANSFER.
FRACT – returns the fractional part of the specified value.
KBD – returns select code of the built-in keyboard.
MAX – returns the largest value in a list of values.
MIN – returns the smallest value in a list of values.
PROUND – the power-of-ten rounding function returns a value rounded to the power of ten specified.
PRT – returns the value 701, the most common select code for a peripheral printer.
SC – returns the select code of an @ name.

String Functions

DATE – accepts a date in the form of DD MMM [–]YYYY and converts it to the number of seconds between that date and the Julian date: 24 Nov –4713.
DATE$ – computes date in DD MMM [–]YYYY format from Julian date.
DVAL – returns the whole number value of the string expression in the radix specified.
DVAL$ – returns the ASCII string containing the specified whole number converted to the radix specified.
ERRM$ – Error Message – returns the text of the error message of the most recent error.
IVAL – returns the integer value of the string expression from –32768 to 32767 in the radix specified.
IVAL$ – returns the ASCII string of the integer value converted to the specified radix.
LWC$ – Lower Case – converts alpha characters to their lower case equivalents in the current lexical order.
REV$ – returns the specified string in reverse order.
RPT$ – returns a string containing a string repeated the specified number of times.
SYSTEM$ – returns system status and configuration information.
TIME – converts a string of the form HH:MM:SS to seconds between midnight and the specified time.
TIME$ – converts the Julian time to a string of the form HH:MM:SS.
TRIM$ – returns a string with any leading or trailing blanks removed.
UPC$ – Upper Case – converts all lower case alpha characters in a string to the corresponding upper case characters in the current lexical order.

Commands

CHANGE – searches the program for the specified string and replaces it with the specified replacement string.
COPYLINES – copies lines of a program from one location to be inserted at another location in a program.
EDIT KEY – allows "edit-key" mode to be entered for defining the special function keys as typing aids.
FIND – searches a program for the specified string.
INDENT – indents your program to reflect the structure based on the statements used.
MOVELINES – moves lines from one location in a program to another.
REN – Renumber – extended to selectively renumber portions of a program.
SCRATCH – erases the program in memory; extended to provide for immediate termination of TRANSFER operations.
SCRATCH A – erases all variables, programs, scratchable binaries and common; extended to provide for immediate termination of TRANSFER operations.
SCRATCH C – erases common area; extended to provide for immediate termination of TRANSFER operations.
SCRATCH KEY – deletes special function key definitions.
XREF – the cross reference command lists the variables, I/O path names line labels and other identifiers used in a program.

General Statements

CALL – extended to access Pascal or Assembly generated subprograms (see CSUB also).
COM – extended to allow for BUFFER specification.
CSUB – the compiled sub statement is used to identify a compiled Pascal or Assembly language subprogram.
DELSUB – the delete subprogram statement is extended to allow for deletion of compiled Pascal or Assembly language subprograms.

DIM – extended to allow for BUFFER specification.
INTEGER – extended to allow for BUFFER specification.
LIST KEY – list the definition of the special function keys.

Event Programming Statements

OFF CYCLE – disables ON CYCLE event branching.
OFF DELAY – disables ON DELAY event branching.
OFF EOR @ – cancels ON EOR @ event branching.
OFF EOT @ – cancels ON EOT @ event branching.
OFF SIGNAL – disables ON SIGNAL software interrupts (see SIGNAL).
OFF TIME – disables event branches set by ON TIME.
ON CYCLE – sets up and enables a periodic event branch each time the specified number of seconds elapses.
ON DELAY – sets up and enables an event branch to occur the specified number of seconds after ON DELAY is executed.
ON EOR @ – sets up and enables an event branch whenever a record terminator is encountered in a TRANSFER operation.
ON EOT @ – sets up and enables an event branch when the last byte is encountered in a TRANSFER operation.
ON SIGNAL – sets up and enables an event branch when a SIGNAL statement is executed.
ON TIME – sets up and enables an event branch when the time specified matches the time in the computers real time clock.
SIGNAL – generates a software interrupt; used with ON SIGNAL.
SYSTEM PRIORITY – sets system priority for event interrupts.

I/O Statements

ABORTIO @ – causes early termination of a transfer to an I/O path.
ASSIGN @ – extended for assigning an I/O path name to a buffer. Also extended to support the following attributes: BYTE or WORD, CONVERT IN or OUT, EOL or EOL OFF, FORMAT ON or OFF, PARITY OFF or ODD or EVEN or ZERO or ONE, WIDTH or WIDTH OFF and RETURN.
BREAK – causes Break sequence to be sent on the RS-232 or Datacomm interface.
BUFFER – used in the DIM, COM, INTEGER, REAL and ASSIGN @ statements to define the buffer area to hold data for the TRANSFER statement.
BYTE – defines byte (8-bit) type transfer for an interface as opposed to word type transfer (used with ASSIGN @).
CONT – used in the TRANSFER statement to indicate that the transfer is to continue until an end-of-transfer condition is encountered.
CONTROL – extended for new CRT register and control operations on a buffer.
CONVERT IN or OUT – used in the ASSIGN @ statement to define conversion tables for ENTER (IN) or OUTPUT (OUT). The conversion table can be set up to convert by INDEX into a table or by PAIRS of characters in the table: the original character and its replacement.
ENTER – extended to allow entering data from a buffer.
EOL – defines End-Of-Line sequence to be sent with PRINT and OUTPUT.
EOL OFF – Resets EOL to Carriage Return/Line Feed.
EOR – used in the TRANSFER statement to specify the End-of-Record delimiter.
IMAGE – extended to include several new format specifiers.
LEXICAL ORDER IS – defines the collating order used by all string relational operators and the UPC$ and LWC$ functions. Six language tables are preset: ASCII, STANDARD, FRENCH, GERMAN, SPANISH, and SWEDISH.
OUTPUT – extended for output to buffers.
PARITY – used with ASSIGN @ to specify parity type.
PASS CONTROL – used to pass active control on the HP-IB to another controller on the bus.
PPOLL RESPONSE – stores the specified value in the parallel poll response register of the HP-IB interface.
PRINT – extended to support WIDTH and EOL sequence.
PRINTALL IS – extended to support WIDTH and EOL sequence.
PRINTER IS – extended to support WIDTH and EOL sequence.
REQUEST – used to request service from the active controller on HP-IB.
RESET – resets an interface, file or buffer.
RETURN – when used with the ASSIGN @ statement, returns a value indicating the outcome of an assign statement.
STATUS – extended to return status information of buffers and an additional CRT register.
TRANSFER – provides for unformatted DMA, interrupt and fast handshake buffer transfers.
WAIT – used in the TRANSFER statement to cause the transfer to execute in serial mode.
WAIT FOR EOR @ – suspends overlapped program execution until a record boundary is reached in a transfer operation.
WAIT FOR EOT @ – suspends overlapped program execution until the completion of a transfer operation.
WIDTH – used in the ASSIGN @, PRINTER IS, and PRINTALL IS to specify the printer width.
WIDTH OFF – used in the ASSIGN @, PRINTER IS, and PRINTALL IS statements to set the printer width to infinite.
WORD – used with ASSIGN @ to define word (16-bit) type of interface transfer.

Mass Storage Statements

CAT – extended to catalog the binary programs and subprograms needed by a PROG type file and the binaries in a BIN type file. New control options have also been added (see COUNT, NO HEADER, SELECT and SKIP).
CAT TO – extended to catalog a mass storage directory to a string array.
CHECKREAD OFF – disables verification of data written to a mass storage file.
CHECKREAD ON – enables verification of data written to a mass storage file.
COUNT – used in the CAT statement to return the number of items listed. Also used in the TRANSFER statement to terminate a transfer when the specified number of bytes have been transferred.
DELIM – used with the TRANSFER statement to assign a delimiter for terminating an inbound transfer.
END – used with the TRANSFER statement to specify an interface dependent message for terminating an inbound transfer (such as EOI for HP-IB).
LOAD KEY – loads the definitions of the special function keys from the specified mass storage file.
LOADSUB FROM – provides a library feature by loading all undefined subroutines or functions from a mass storage file. Can also be used to load a single subprogram by specifying the subprogram name.
MASS STORAGE IS – extended to include support for the HP913X.
NO HEADER – used with the CAT statement to disable header information in a catalog listing.
RECORDS – used in the TRANSFER statement to specify the number of records to be transferred.

RE-STORE KEY – stores special function keys into a previously used file name.
SELECT – used with the CAT statement to select entries that begin with the specified characters.
SKIP – used with the CAT statement to skip the specified number of items before listing catalog entries.
STORE KEY – stores the special function key definitions onto the specified file.

Matrix Functions

BASE – returns the lower bound for the specified dimension of an array.
DET – returns the determinant of the specified matrix or the last matrix inverted.
DOT – returns the inner (dot) product of the two specified vectors.
RANK – returns the number of dimensions in an array.
SIZE – returns the number of elements in the specified dimension of an array.
SUM – returns the sum of all elements in an array.

Matrix Statements

CSUM – see MAT CSUM.
IDN – see MAT IDN.
MAT +, –, *, /, <, <=, =, <>, >=, > – perform the indicated scalar math on a numeric array.
MAT +, –, ., /, <, <=, =, <>, >=, > – perform the specified array math, element by element, on the specified numeric arrays (. is for element by element multiply.)
MAT * – matrix multiplication.
MAT ... = – array assignment or initialization.
MAT REORDER – used to rearrange the data in an array according to an associated pointer vector.
MAT SORT – used to reorder an array or set up a reorder vector according to a specified key in either ascending or descending order.
MAT CSUM – computes a column sum on the specified matrix.
MAT IDN – sets a square matrix to the identity matrix: the main diagonal becomes all 1's, all other elements become 0's.
MAT INV – produces the inverse of the specified matrix.
MAT RSUM – computes the row sum on the specified matrix.
MAT TRN – computes the transpose of the specified matrix.
REDIM – Redimension – changes the subscript bounds of a list of arrays.

Advanced Program Binary Interface Capabilities

Incremental to the BASIC 2.0 interface capabilities, the Advanced Programming Binary provides support for the BCD Interface (98623A). The optional DMA Controller Card (98620A) is also supported and provides high-speed DMA transfers with the GPIO and HP-IB interfaces. These are capabilities supported by the Advanced Programming Binary and are not available in BASIC 2.0.

Color Video Interface Binary Keyword Summary

Graphics Statements

DIGITIZE – determines the X and Y coordinates of the current graphics input device's locator when the digitize button (or stylus) is pressed.
GRAPHICS INPUT IS – defines the device for graphics input.
PLOTTER IS – extended to allow graphics output to an external color display.
READ LOCATOR – determines the X and Y coordinates of the current graphics input device's locator without requiring the digitize button (or stylus) to be pressed.
SET ECHO – sets the current graphics output device's locator position to the specified position.
TRACK ... IS ON/OFF – enables the graphics input device's locator to be tracked by the current graphics output device's cursor (or pen) position.

Color Video Binary Interface Capabilities

Incremental to the BASIC 2.0 interface capabilities, the Color Video Binary provides support for the external Color Video Interface (98627A).

Pascal 2.0* Capabilities

The Pascal 2.0 Language System gives you all the advantages of a compiled language and more. It offers you high speed and protection from program modification, plus extensive enhancements not found in other Pascal systems.

The system brings to the Model 16:

- Powerful and versatile data and programming structures
- An I/O Procedure Library providing sophisticated device I/O
- A Graphics Library with full peripheral support
- A Librarian allowing the user to create custom procedure libraries
- A comprehensive Editor to speed up program development
- Extensive debugging capability for software troubleshooting
- Motorola MC68000 Assembly Language
- A high-performance compiler which produces ready-to-run MC68000 object code.

These powerful features can now be delivered directly to your desktop or test bay packaged in the state-of-the-art hardware of the Model 16.

Minicomputer Tools at Your Fingertips

This system brings you the sophisticated program development tools of a minicomputer. A menu-driven Command Interpreter is used to load the various program development subsystems (Editor, File Manager, Compiler,etc.) from mass storage or to make the subsystems part of RAM-resident libraries for almost instantaneous access.

Standard I/O Procedures and More

I/O Procedures of HP Standard Pascal (the same Pascal used on the HP 1000 and HP 3000) simplify program development by providing support for a full range of printers and mass storage devices. An enhanced I/O Procedure Library is supplied as an integral part of the system. This library allows you to solve complex interfacing tasks by providing such sophisticated capabilities as DMA transfers, interrupt I/O and HP-IB bus control. All of this is made available through the familiar method of high-level Pascal procedure calls.

Graphics for Data Display or Monitoring

A Graphics Procedure Library provides 26 powerful commands for two dimensional, line oriented graphics. The library is a functional subset of the DGL graphics offered on the HP 1000 Series computers so that programs using that subset will generally transport with little or no trouble. Graphics performance on the internal CRT ranges up to 2500 ¼-in. vectors per second. And, of course a wide range of peripherals are supported for everything from color video display to graphics plotters.

Fine-tuned Program Performance

A Motorola MC68000 Assembler lets you write speed-critical or frequently executed algorithms directly in Assembly language. These routines can then be called from Pascal programs.

The Efficiency of a Modular System

Before installing programs for an application, unneeded parts of the system can be easily removed. For instance, the File Manager can be removed from the system to reduce storage requirements and to prevent tampering with files.

THE Pascal Language

The structure of programs and data in Pascal is strictly defined. Yet within that structure the programmer is provided with great flexibility that allows complex problems to be solved in a simple, self-documenting fashion.

The benefits of Pascal include:

- Flexible program design – extremely flexible data types allow the language to be adapted to the problem, rather than the reverse.
- Structured program design – a large program can be easily broken down into its component parts to yield a simpler and more efficient solution.
- Modular program development – frequently-used procedures and data types can be compiled into module libraries and then selectively imported into programs and procedures.
- Simplified program maintenance – Pascal gives you a "handle" on the costs of software maintenance. The final program is readable and understandable, so it's easier to maintain and adapt to the changing needs of the application.

The Programming Environment

The HP Standard Pascal Language System consists of a Command Interpreter and six separate subsystems – editor, file manager, compiler, assembler, debugger and linker/librarian.

The Command Interpreter

The main job of the command interpreter is to load and run the other subsystems. The features of the Command Interpreter are:

- Menu driven – functions are executed by pressing a single key.
- Automatic loading and linking – programs can be compiled and executed by pressing one or two keys; compiling, loading, and linking is done automatically.
- Permanently loaded libraries – such libraries as the editor or file manager subsystems can be made memory-resident for almost instantaneous access.
- Memory resident volumes – allows you to reserve part of your computer's memory for use as an ultra-fast mass storage medium.

The Editor

A versatile screen-based editor allows you to create, change and store programs, documents and virtually any kind of text imaginable. The editor has a built-in prompt line which makes it easy to learn how to use while its single keystroke commands save you unnecessary typing. Features of the editor include:

- Insertion and deletion of text
- Use of the "knob" for high-speed scrolling
- User-defined "markers" that allow sections of text to be accessed directly
- String search and replace capability
- Moving and duplicating sections of text within the editor
- Copying sections of text into the editor from external files
- Optional auto indentation for text or program structuring
- Adjustable margins

* Pascal expected availability December, 1982.

The File Manager

The file manager provides a means of managing the volumes and files used by the Pascal Language System. A volume may consist of a series of files with a directories or it may be a device such a printer or CRT. The capabilities of the file manager include:

- Creating file directories
- Listing file directories
- Creating, removing and changing the names of files
- Copying a file or an entire volume of files for backup
- Removing an entire volume of files
- Listing files to an external device
- Packing volumes to regain unused mass storage space

The file manager capabilities support three directory formats, including Logical Interchange Format (LIF). LIF text files can be transported between the Pascal Language System and other Hewlett-Packard peripherals and computers which support LIF ASCII format, including Model 16 BASIC 2.0.

The Compiler

The Pascal Language System compiler compiles HP Standard Pascal source code stored on mass storage files into MC68000 machine code. A list of the reserved words, standard procedures and functions, library procedures and functions, and compiler options is included in this publication.

The advantages of the compiler include:

- Performance – source code is compiled directly into executable MC68000 code; there is no intermediate interpreter to degrade execution speed.
- Language power – compiles HP Standard Pascal, the version of ISO (International Standards Organization) Standard Pascal that is also used on the HP 1000 and HP 3000 computers. The compiler can be directed to accept four language extensions which are a significant aid in program development (see SYSPROG Compiler Directives).
- Compatibility – existing software investment is protected.

The compiler can be directed to accept only ISO Standard Pascal and to accept most UCSD Pascal extensions. The HP Standard extensions to ISO Standard Pascal and the UCSD extensions are indicated in the list of Reserved Words and Standard Procedures. A list of the UCSD Pascal extensions is also included.

- Flexibility – a comprehensive set of compiler directives is available to direct such details of the compilation as compiler listings, error checking and debugging.

The Assembler

The Pascal Language System includes a Motorola MC68000 Assembler. The Assembler gives the user access to the powerful MC68000 instruction set. Object code is generated from source files written via the editor. Assembled code can then be executed independently, or can be called as external procedures from Pascal programs. In certain cases speed can be increased for time-critical or freqently-executed sections of code by writing them in Assembly language.

The Debugger

The Pascal Language System Debugger is a powerful debugging facility which can be used to debug Pascal and Assembly language programs. The Debugger maintains an independent display screen so that the user display can be toggled on or off during debugging.

The capabilities of the Debugger include:

- Stepping through programs one line at a time
- Setting program break points
- Display and modification of register contents
- Formatted display of memory contents
- Byte, word and long word modification of memory contents
- Display and traversal of the Pascal procedure stack
- Error trapping
- Register trace capability

The Linker/Librarian

The linker/librarian serves to create and maintain libraries. The libraries hold commonly-used modules of code which can be automatically accessed by programs calling for them.

The benefits provided by the linker/librarian include:

- Leverage – frequently-used routines and data types can be written once and then shared by many different programs.
- Modular program development – a group of programmers can independently work on sections of a program and then integrate them using the library facility.

The linker/librarian also provides a disassembler that can be used to list an assembly source version of the MC68000 code generated by a Pascal compilation or a MC68000 assembly.

Unlike many compiled systems, the linker/librarian need not be used before executing programs since linking is taken care of automatically at run time.

Range

Real Precision:

$-1.797693134862315E+308$ to $-2.225073858507202E-308$

0

$2.225073858507202E-308$ to $1.797693134862315E+308$

Integer Precision:

-2^{31} to $+2^{31}-1$ (32-bit integer)

Math Hierarchy

Highest priority	( )
	NOT, UNARY+, UNARY–
	*, /, DIV, MOD, AND, +, – OR
Lowest priority	=, <>, >, <=, >=, <, IN

Execution Times

All times for the following are in milliseconds:

	Time	
Operation	**Real**	**Integer**
For Loop		.009
Assignment	.010	.006
Addition	.106	.009
Subtraction	.131	.010
Multiplication	.285	
Division	.414	
Exponentiation	3.470	
Sine	3.600	
Square Root	1.720	
Log	3.690	
Procedure Call/Return	.030*	

* Procedure Call/Return includes no parameter passing. Time can be reduced to .011 msec by use of Sovflcheck, $range, and $stackcheck compiler directives.

The Pascal 2.0 Language System

Reserved Words

The following list describes the words that are reserved by the Pascal 2.0 Language System. Those words not reserved by the UCSD or ISO language definitions have been indicated by "* (not UCSD)" or "+ (not ISO)".

And – specifies the logical conjunction of its boolean operands
Array – a structured data type (see Data Types).
Begin – used to delimit the start of a compound statement.
Case – provides for execution of one of several statements depending upon the value of a selector variable.
Const – precedes the constant definition block in a program, procedure or module.
Div – provides division with truncation on integer operands.
End – used to delimit the start of a compound statement.
Export – precedes a list of procedure and data descriptors in a module to be made available to other programs (see Import, Implement) (*, +).
File – a structured data type (see Data Types).
For..Do – defines a loop which is repeated until the loop counter exceeds its final value.
Function – defines a function block which returns a value of the type assigned.
Goto – provides an unconditional branch to a specified label.
If..Then, If..Then..Else – provides for conditional execution of one of more statements or procedure calls.
Implement – precedes the program block of a module. (see Export,Import) (*, +).
Import – precedes a list of module identifiers to be imported into the program block (see Export,Implement) (*, +).
In – tests for inclusion of an element in a set.
Label – precedes the label declaration block in a program, module, or procedure.
Mod – yields the remainder after division on integer operands.
Module – identifies the heading of a module and specifies the module identifier (*, +).
Nil – a value associated with a pointer not assigned to an object.
Not – denotes the logical negation of its boolean operand.
Or – specifies the logical disjunction of its boolean operands.
Packed – used with Array to define a structured data type (see Data Types).
Procedure – identifies the heading of a procedure and specifies the identifier and formal parameter list for the procedure.
Program – identifies the heading of a program and specifies the identifier and a list of entities through which the program communicates with its environment.
Record – a structured data type (see Data Types).
Repeat..Until – defines a loop which is repeated until the specified condition comes true. Always executed at least once.
Set – a structured data type (see Data Types).
Type – precedes the variable declaration block in a program, module, or procedure.
Var – precedes the variable declaration block in a program, module, or procedure.
While..Do – defines a loop which is repeated while the specified condition remains true. The loop may terminate before the first execution.
With..Do – provides for accessing the fields of a record as variables.

Standard Procedures and Functions

The following list describes the procedures and functions of the standard Pascal 2.0 Language System. Those functions and procedures not included as part of the standard UCSD or ISO language definitions have been indicated by "* (not UCSD)" or "+ (not ISO)". Differences in syntax are footnoted.

File Handling

Reset(f) -	procedure	file f is opened read only and positioned at the first component.
Rewrite(f) -	procedure	file f is opened write only and positioned at the first component.
Close(f) -	procedure	file f is closed from read and write states (+).
Put(f) -	procedure	writes the value of the buffer variable f↑ to the current component of f and advances to the next component.
Get(f) -	procedure	advances the current position.
Open(f) -	procedure	file f is opened for direct access (read/ write) and positioned at the first component (*, +).
Seek(f,k) -	procedure	direct access file f is positioned at component k (+).
Position(f) -	function	returns the index of the current component in file f (*, +).
Maxpos(f) -	function	returns the index of the last possible component of file f (*, +).
Eof(f) -	function	returns boolean true if the file f is not readable, otherwise returns boolean false and advances the file pointer to the next file component.
Eoln(f) -	function	returns boolean true if an end-of-line is reached in textfile f, otherwise returns boolean false

Dynamic Allocation

New(p) -	procedure	allocates a new variable and assigns p as a pointer to it.
Dispose(p) -	procedure	deallocates the space associated with the variable pointed to by pointer p (*).
Mark(p) -	procedure	marks a space in the heap from which space is allocated with subsequent calls to New (+).
Release(p) -	procedure	releases the entire space marked by the previous call to mark with pointer p (+).

Arithmetic and Numeric

Abs -	function	returns the absolute value of the argument.
Sqr -	function	returns the squared value of the argument.
Sqrt -	function	returns the square root of the argument.
Sin -	function	returns the sine of the radian argument.
Cos -	function	returns the cosine of the radian argument.

Arctan -	function	returns the arctangent of the argument in radians.
Exp -	function	raises the base of the natural logarithm to the power of the argument.
Ln -	function	returns the natural logarithm of the argument.
Odd -	function	returns boolean true if the argument if odd, false otherwise.
Trunc -	function	returns the truncated integral part of its real or longreal argument.
Round -	function	returns the value of its real argument rounded to an integer.
Pack(a,i,z) -	procedure	moves the components of ARRAY a into PACKED ARRAY z starting at an offset i into z (*).
Unpack(z,a,i) -	procedure	moves the components of PACKED ARRAY z into ARRAY a starting at an offset i into a (*).
Hex -	function	returns the decimal integer value of its hexidecimal string argument (*, +).
Octal -	function	returns the decimal integer value of its octal string argument (*, +).
Binary -	function	returns the decimal integer value of its binary string argument (*, +)

Ordinal

Chr -	function	returns the character whose ordinal value is given by the argument.
Succ -	function	returns the value whose ordinal value succeeds that of the argument.
Pred -	function	returns the value whose ordinal value preceeds that of the argument

String

+ -	operator	used infix for concatenation (+,1).
str(s,b,l) -	function	returns a substring of length l beginning at character b from string argument s (+,2).
strlen(s) -	function	returns the current length of its string argument (+,3).
strmax(s) -	function	returns the maximum allowable length of the string argument (*, +).
setstrlen(s,l)	procedure	sets the current length of the string s to length l (*, +)
strmove (n,s1,p1,s2,p1) -	procedure	moves n characters from string s1 beginning at character p1 to string s2 beginning at character p2 (*, +)
strwrite (s,p1,p2,l) -	procedure	formats variable list l to string s beginning at character p1 and ending at character p2 using textfile conventions (+,4).
strread (s,p1,p2,l) -	procedure	extracts formatted values from string s beginning at character p1 and ending at character p2 and puts it in variable list l using textfile conventions (*, +).
strappend (s1,s2) -	procedure	appends string s1 onto s2 (*, +).
strinsert (s1,s2,n) -	procedure	inserts s1 into s2 starting at character n (+,5).
strdelete(s,b,n) -	procedure	removes n characters from string s beginning at character b (+,6).
strltrim(s) -	function	removes leading blanks from string s (*, +).
strrtrim(s) -	function	removes trailing blanks from string s (*, +).
strpos(s1,s2) -	function	returns the index of the characters s2 where the first occurrence of s1 begins (+,7).
strrpt(s1,n) -	function	returns a string of n concatenated copies of string s1 (*, +).

1. Also supported by UCSD CONCAT function.
2. Also supported by UCSD COPY function; UCSD uses STR for different purpose.
3. Also supported by UCSD LENGTH function.
4. limited version supported by UCSD STR function.
5. Also supported by UCSD INSERT function.
6. Also supported by UCSD DELETE function.
7. Also supported by UCSD POS function.

Input and Output

Read(f,v) -	procedure	performs a read operation of file f storing the result in variable v.
Readln(f,v) -	procedure	equivalent to read, but skips to the beginning of the next line after completion.
Write(f,p) -	procedure	performs a write operation to the file f of the write parameter p.
Writeln(f,p) -	procedure	equivalent to write, but appends a line marker to the file f.
Readdir (f,k,v) -	procedure	moves direct access file f to component k and then does the equivalent of a Read.
Writedir (f,k,p) -	procedure	moves direct access file f to component k and then does the equivalent of a Write.
Page(f) -	procedure	causes a top-of-form to be output when the textfile f is printed.
Prompt(f) -	procedure	writes list without a carriage return.

Data Types

Standard Types

Array – structured type having a fixed number of components, all of the same type.

Boolean – simple type with values "true" and "false" with true > false.

Char – simple type with values defined by the 8-bit ASCII character set.

File – structured type specifying a data structure consisting of a sequence of components all of the same type, but not fixed in number.

Integer – simple type with whole number values ranging from -2^{31} to $-2^{31}-1$.

Longreal – simple type with value range same as Real.

Packed Array – an Array with components stored in compacted form.

Real – simple type of real numbers with precision of 64-bit binary floating point.

Record – a structured type having a fixed number of components, possibly of different types. Variant records are allowed.
Set – a structured type defining a range of values that can be operated on by the set operators.
String – a structured type representing a packed array of type Char with dynamically varying length (not part of ISO).
Text – a structured type having components of type File of Char, but structured into lines separated by line markers.

Pointer Type
A type pointing to the components of a dynamic variable.

Subrange Type
A type defined as a subrange of another ordinal type.

Enumerated Type
A simple type defined by a user-defined list of values.

Input/Output Procedure Library

The I/O procedure library provides enhanced I/O capabilities for those complex tasks where Pascal's standard I/O procedures are not adequate. It contains a group of procedures which can be used with the following interface cards:

- 98622A GPIO Interface Card
- 98624A HP-IB Interface Card and internal HP-IB
- 98626A Serial Interface Card and internal Serial Interface
- 98628A Datacomm Interface Card

An optional DMA Controller (98620A) is available to support high-speed DMA transfers with the GPIO and HP-IB interfaces.

A group of general procedures can be used independent of the interface card. In addition, there are two groups of procedures designed to take advantage of the unique capabilities of the HP-IB and serial interface cards.

This library is not part of the standard Pascal language. It is provided as an external procedure library which is included as part of the Pascal 2.0 Language System.

General Procedures

This group of procedures can be used independent of the type of interface card.

Abort_transfer – terminates a transfer on the specified buffer.
Buffer_data – a function returning the number of characters available in the buffer.
Buffer_reset – resets the buffer pointers of the specified buffer.
Buffer_space – a function returning the available space in the specified buffer.
Iobuffer – creates a buffer area of the specified number of bytes.
Iocontrol – outputs a value to the specified interface card control register.
Ioinitialize – resets the I/O system and its drivers.
Ioread_byte – a function returning a byte from the specified interface card register.
Ioread_word – a function returning a word from the specified interface card register.
Ioreset – resets the specified interface to its power-up state.
Iostatus – a function returning the specified interface status register.
Iowrite_byte – writes a byte to the specified interface register.
Iowrite_word – writes a word to the specified interface register.
Off_eot – cancels branch specified by On_eot.
On_eot – causes branch to user specified procedure upon completion of buffer transfer.
Readbuffer – reads a single byte from the specified buffer.
Readbuffer_string – reads a specified number of string characters from a buffer into a string.
Readchar – reads as single byte from the specified interface.
Readnumber – performs a free field numeric entry from the specified interface.
Readnumberln – performs a free field numeric entry from the specified interface terminated by a line feed.
Readstring – reads characters into the specified string.
Readstring_until – reads characters into the string until the specified termination character is received.
Readuntil – reads characters until the specified match character is received.
Readword – reads a single, 16-bit numeric value from the specified interface.
Set_timeout – sets the minimum time the computer will wait for an I/O operation to the specified interface to complete.
Skipfor – reads, but does not store, the specified number of characters.
Transfer – transfers the specified number of bytes into or out of a buffer using fast handshake or DMA.
Transfer_end – transfers data into or out of a buffer until an EOI is generated.
Transfer_until – transfers data into a buffer until the specified terminator occurs.
Transfer_word – transfers a specified number of 16-bit values into or out of a buffer.
Writebuffer – writes a single byte into a buffer.
Writebuffer_string – writes the specified string into a buffer.
Writechar – writes a single byte of data to the specified interface.
Writenumber – performs a free field output with no terminator.
Writenumberln – performs a free field output terminated by carriage return/line feed.
Writestring – writes a string to the specified interface with no terminator.
Writestringln – writes the string followed by a CR/LF.
Writeword – writes a single, 16-bit value to the specified interface.

HP-IB Procedures

This group of procedures is used for accessing the unique capabilities of the 98624A HP-IB interface card and internal HP-IB port.

Abort_hpib – sends the abort message to all devices.
Active_controller – a function indicating whether the specified interface is in active controller state.
Addr_to_listen – send interface talk address, device listen address.
Addr_to_talk – send interface listen address, device listen address.
Clear – sends a device clear or selected device clear message.
Clear_hpib – clears the specified HP-IB control line.
End_set – function indicating whether EOI was sent on last byte read.
Hpib_line – a function returning the current state of the specified HP-IB control line.
Listen – sends the specified listen address over the bus.
Listener – a function indicating whether the specified interface is addressed to listen.

Local – puts the indicated device(s) in local state.
Local_lockout – sends the local lockout message to the bus.
Locked_out – a function indicating whether the specified interface is in local lockout state.
My_address – a function returning the HP-IB address of the specified interface.
Pass_control – passes control from the specified interface to another device on the bus.
Ppoll – performs a parallel poll of the bus.
Ppoll_configure – configures the specified device on the bus for a particular parallel poll response.
Ppoll_unconfigure – causes the specified device(s) on the bus to disable the parallel poll response.
Remote – puts the specified bus device(s) in remote programming state.
Requested – a function indicating whether service is being requested.
Secondary – sends a secondary command byte to the bus.
Send_command – sends a single specified command byte to the bus (ATN true).
Request_service – used to set up the serial poll response byte and to request service from the active controller.
Set_hpib – sets the specified HP-IB control line.
Spoll – performs a serial poll on the specified device.
System_controller – a function indicating whether the specified interface is system controller.
Talk – sends the specified listen address over the bus.
Talker – a function indicating whether the specified interface is addressed to talk.
Trigger – sends the trigger command to the specified device(s) on the bus.
Unlisten – sends an unlisten command over the bus.
Untalk – sends an untalk command over the bus.

Serial Procedures

This group of procedures is used for accessing the unique capabilites of the 98626A/98628A interface cards.

Clear_serial – clears the specified serial link.
Send_Break – sends a break sequence over the specified serial link.
Serial_line – a function returning the state of the specified serial line.
Set_baud_rate – sets the baud rate of the specified serial interface.
Set_char_length – sets the character length of the specified serial interface.
Set_parity – sets the parity mode of the specified serial interface.
Set_serial – sets the specified serial interface modem line.
Set_stop_bits – sets the number of stop bits of the specified serial interface Set_stop_bits – sets the number of stop bits of the specified serial interface.

Graphics Procedure Library

Await_locator – Waits for and reads from locator device.
Clear_display – Clears the graphics display.
Display_init – Enables the graphics display.
Display_term – Disables the graphics display.
Graphics_init – Initializes the graphics system.
Graphics_term – Terminates the graphics system.
Gtext – Outputs graphical text to the graphics display.
Input_esc – Invokes a device dependend inquiry from the graphics display.
Inquire_ws – Returns device-dependent information from graphics system.
Int_line – Draws a line to the integer coordinate specified.
Int_move – Moves to the integer coordinate specified.
Line – Draw a line to the specified real coordinate.
Locator_init – Enables the locator device for input.
Locator_term – Disables the locator device.
Move – Moves to the specified real coordinate.
Output_esc – Performs a device dependent function on the graphics display.
Sample_locator – Samples the locator device.
Set_aspect – Redefines the aspect ratio of the virtual coordinate system.
Set_char_size – Sets the character size for graphical text.
Set_color – Set the color for graphics operations.
Set_display_lim – Defines the limits of the graphics display (in mm).
Set_echo_pos – Defines the locator echo position on the graphics display.
Set_line_style – Sets the line style for lines and text.
Set_locator_lim – Redefines the locator limits of the graphics locator.
Set_text_rot – Specifies the direction for graphical text.
Set_viewport – Maps the graphics user units to a specified location on the graphics display. (see Set_window)
Set_window – Defines the boundaries of the graphics display in user units.

Compiler Directives

ALIAS – specifies an external name for a procedure
ANSI – causes error message to be issued for any non-ANSI Standard feature of HP Standard Pascal.
CALLABS – chooses between 32-bit absolute and 16-bit program counter relative jumps for forward and external procedure calls.
CODE – causes executable code to be emitted.
CODE_OFFSETS – causes line number/program counter pairs to be printed for each executable statement listed.
COPYRIGHT – places a string in the object file indicating the program copyright ownership.
DEBUG – causes text to be compiled with debug capabilities.
DEF – specifies the number of logical records allowed for external definitions.
IF..END – allows for condition compilation of the delimited region of text.
HEAP_DISPOSE – allows disposed objects to be reused.
INCLUDE – specifies a text file to be included at the indicated position in the program.
IOCHECK – causes error checks to be emitted following calls on system I/O routines.
LINENUM – causes the integer parameter which follows to become the current line number; useful in listing and debugging.
LINES – specifies number of lines per listing page.
LIST – causes the source to be listed.
OVFLCHECK – causes overflow checks to be emitted for all in-line arithmetic operations.
PAGE – causes listing to resume at top of next page.
PAGEWIDTH – specifies number of characters per printer line.

PARTIAL_EVAL – suppress evaluation of the right operand of AND (OR) operator when the left operand is FALSE (TRUE).
RANGE – causes run time checks to be emitted for array and case indexing, subrange assignment and pointer dereferencing.
REF – specifies the number of logical records allowed for external references.
SAVE_CONST – causes compile time storage for structure constants to be retained for the scope of the constant's name.
SEARCH – specifies the name and order of files to be searched to satisfy IMPORT references.
STACKCHECK – causes stack overflow checks to be emitted at procedure entry.
SYSPROG – enables the compiler for the following four system programming extensions, such as:
TRY..RECOVER – statement for error trapping.
SIZEOF – function returning the size of a data type or variable, ADDR: function returning the address of a variable.
ANYVAR – disables type compatibility checking for procedure variables.

Note: These features are not part of HP Standard Pascal.

TABLES – causes symbol table information to be printed after the listing of each procedure.
UCSD – makes UCSD Pascal standard procedures and function available (see UCSD COMPATIBILITY).

UCSD Compatibility

Pascal 2.0 provides for compatibility with most of the standard procedures and functions of UCSD Pascal. The main objective is to allow UCSD Pascal programs to be moved easily to the Model 16 to protect existing software. Future versions of HP Standard Pascal will probably not provide UCSD compatability. While compatability with UCSD Pascal is not perfect , most programs will transport with minimal effort. Here is a list of UCSD features and the corresponding level of support on the Model 16:

UCSD Feature	Model 16 Support
Default string length 80	unsupported; no default length
Maximum string length 255	supported
Arbitrary string type as actual parameter	procedure must specify STRING
Strings	
Setting length of string	supported
LENGTH of string function	supported
POS string position function	supported
CONCAT string function	supported
COPY substring function	supported
DELETE substring function	supported
INSERT substring function	supported
SCAN string/character array procedure	supported
MOVELEFT byte-oriented data moving	supported
FILLCHAR byte stream fill	supported
I/O	
Untyped files	supported
UNITREAD direct I/O	supported
UNITWRITE direct I/O	supported
UNITBUSY I/O test	supported
UNITCLEAR I/O flush	supported
BLOCKREAD direct I/O	supported
BLOCKWRITE direct I/O	supported
IORESULT variable	supported
SEEK random access positioning	supported, but with index base
CLOSE file options LOCK,NORMAL,PURGE,-CRUNCH	option must be a string
UNITWAIT I/O idle	supported
INTERACTIVE text files	specifier not allowed, but equivalent behavior provided by HP TEXT files
Standard units PRINTER, CONSOLE, SYSTERM, GRAPHIC	supported
Data Structures	
SIZEOF variable or type	supported
HALT program termination	supported
GOTOXY cursor positioning	supported
MEMAVAIL heapspace interrogation function	supported, but returns bytes, not words
Program heading w/o files listed	supported
EXTERNAL procedures/functions	supported
SETs with up to 4k elements	support limited to 255 elements
16-bit integer	normal integer 32 bits; may declare subrange –32768 to 32767
Long BCD integer to 36 digits	unsupported; have 9.5 digit integer
STR long integer to string conversion	HP Pascal has more general STRWRITE
32-bit real numbers	Model 16 uses 64-bit reals
LOG function	only LN supported
TIME functions	returns different format
PWROFTEN function	unsupported
Multiword comparison of arrays, records	unsupported
Nested comments	unsupported
Packing rules	totally different; machine dependent
CASE statement for illegal selector	must add OTHERWISE clause
EXIT statement	unsupported; can be simulated by SYSPROC TRY/RECOVER
SEGMENT procedures	unsupported
Separate compilation UNITs	HP Pascal MODULES provide a similar facility; some translation is required.

Compiler Options

F (byteflip)	unsupported (irrelevent)
C (copyright)	supported as COPYRIGHT
D (debugging)	supported as DEBUG
G (goto's allowed)	unsupported (always allowed)
I (iochecks)	supported as IOCHECK
Intermixed declarations in INCLUDE	supported
L (listing control)	supported as LIST
P (page eject)	supported as PAGE
Q (quiet screen)	unsupported (irrelevent)
R (range checks)	supported as RANGE
S (swapping compiler segments)	unsupported (irrelevent)
U (user mode compilation)	unsupported (irrelevent)

Interface Capabilities

In addition to the built-in HP-IB and Serial Interfaces, there is a choice of several external interface cards for use with the BASIC and Pascal Language Systems. Their support is shown in the following table:

	BASIC 2.0	Pascal 2.0
GPIO (98622A)	Yes	Yes
BCD (98623A)	Requires Advanced Programming Binary	No
HP-IB (98624A)	Yes	Yes
Serial (98626A)	Yes	Yes
Color Video (98627A)	Requires Color Video Interface Binary	Yes
Data Communications (98628A)	Yes	Yes

These interface cards do not include the cables which must be ordered separately or under option. An optional DMA Controller Card (98620A) is available to support high-speed DMA transfers with the GPIO and HP-IB interfaces. It is supported in Pascal 2.0 and requires the Advanced Programming Binary in BASIC 2.0.

GPIO Interface

The 98622A GPIO Interface provides 16 bits of latched input and output data for bidirectional transfer of information. Extended control and status lines are available for applications that require more than one signal from the computer. Several handshake modes are also available to permit interfacing to a variety of equipment.

Data Input/Output

There are 16 output data lines and 16 input data lines. The output lines provide high current/voltage drivers, using open-collector buffers. The input data lines are terminated by a resistive divider of 3K Ohms to +5 V and 6.2K Ohms to ground accepting standard TTL signal levels.

Electrical Characteristics for Data Output Lines

	Min.	Max.	Units
Output Low Voltage @ 16 mA		0.4	V
Output Low Voltage @ 40 mA		0.7	V
Output High Voltage (open collector)		30.0	V
Output Low Current		40.0	mA
Output High Current @ Output High Voltage		0.25	mA

Electrical Characteristics for Data Input Lines

	Min.	Max.	Units
Input Low Voltage		0.8	V
Input High Voltage	2.0		
Input Current @ Input Low Voltage = 0.4V		−0.8	mA
Input Current @ Input High Voltage = 2.7V		40	μA

Control Lines

Ten lines provide control information between the peripheral and the 98622A GPIO Interface. The outgoing lines are electrically equivalent to the open-collector data output lines. The incoming lines have the following characteristics:

Electrical Characteristics for Control Input Lines

	Min.	Max.	Units
Input Low Voltage		0.6	V
Input High Voltage	1.9		V
Hysteresis	0.4		V
Input Low Current @ Input Low Voltage = 0.4V		−0.4	mA
Input High Current @ Input High Voltage = 2.7V		20	μA

The control lines and their meanings are:

PCTL Peripheral Control – indicates that the Model 16 computer is ready for input data or that data is ready for output; PCTL is reset by a ready-to-busy transition on PFLG or by an interface reset.

PFLG Peripheral Flag – indicates to the Model 16 computer that the peripheral has completed the data transfer; also used to request peripheral interrupt when enabled.

PSTS Peripheral Status (optional) – indicates to the computer the readiness of the peripheral; PSTS is sampled by the computer whenever communication with the peripheral is requested.

STI0, STI1 Extended Status (optional) – driven by the peripheral and may be used for any purpose; examined by reading the 98622A peripheral status register.

CTL0, CTL1 Extended Control (optional) – driven by the computer and may be used for any suitable purpose by the user; asserted by writing to the 98622A peripheral control register.

I/O Direction – indicates to the peripheral the direction of the current data transfer

PRESET Peripheral Reset – used to initialize a peripheral when the Model 16 computer is turned on, when the RESET key or CLEAR I/O key are pressed or when the 98622A peripheral reset register is written to.

EIR External Interrupt Request – used to generate an interrupt request based on some external event; the current state can be examined by reading the 98622A peripheral status register.

Switch Configuration

The following switches can be configured on the interface card.

Select Code – the factory select code setting for the 98622A card is 12; the 98622A card can have a select code setting from 8 to 31.

Interrupt Level – the factory interrupt priority level setting for the 98622A card is 3; the 98622A card can have an interrupt level setting from 3 to 6.

Output Data Line Sense – a 1-bit switch allows the output data lines to use either positive-true or negative-true logic.

Input Data Line Sense – a 1-bit switch allows the input data lines to use either positive-true or negative-true logic.

PFLG Line Sense – a 1-bit switch allows the peripheral flag line to use either positive-true or negative-true logic.

PCTL Line Sense – a 1-bit switch allows the peripheral control line to use either positive-true or negative-true logic.

PSTS Line Sense – a 1-bit switch allows the peripheral status line to use either positive-true or negative-true logic.

Handshake Mode – a 1-bit switch allows selection of full or pulsed handshake mode.

Data In Clock Source – a 6-bit switch allows selection of when input data is to be clocked into the 98622A input latches. The upper and lower input bytes can have seperate clock sources chosen from PFLG ready to busy transition, or busy to ready; or when the computer reads the input latch.

DMA Capability

(Requires the Advanced Programming Binary for BASIC.)

The 98622A is capable of carrying out DMA transfers with the computer via the optional two-channel 98620A DMA Controller Card. The following DMA capabilities are supported by the 98622A:

- Word or Byte Mode DMA
- Regular or Burst DMA transfer

Additional BASIC 2.0 Capabilities

Interrupt Capability

The 98622A is capable of generating interrupts to the computer under the following conditions:

- PCTL clear
- PCTL clear & PFLG ready
- EIR asserted

Transfer Rates

The maximum data rates for the 98622A GPIO Interface with the BASIC 2.0 Language System are as follows:

	Input	Output
Handshake	63K bytes/sec.	65K bytes/sec.

With the Advanced Programming Binary:

Interrupt, burst	65K transfers/sec	75K transfers/sec
Fast handshake	115K transfers/sec	115K transfers/sec
DMA, regular	540K transfers/sec	480K transfers/sec
DMA, burst	770K transfers/sec	670K transfers/sec

BCD Interface

(Not supported in Pascal. Requires Advanced Programming Binary for BASIC.)

The 98623A BCD Interface connects the Model 16 computer with bit-parallel, digit-parallel, binary-coded decimal devices for data input. Up to eight significant BCD digits, two sign bits (mantissa and exponent), exponent digit, function code digit, and an overload bit can be read. Input format is selectable, allowing two independent instruments to be read from one 98623A Interface Card. Data can also be accepted as five input bytes of pure binary information. Eight data output lines are also provided for use as general purpose control and/or data output.

Transfer Rates

The maximum data rates for the 98623A BCD Interface with the Advanced Programming Binary are as follows:

	Input
Handshake (formatted)	5.2K bytes/sec (325 readings/sec)
Handshake (unformatted)	34K bytes/sec (6.9K readings/sec)

Data Input/Output

The 98623A BCD interface provides 43 data input lines (eight BCD digits, mantissa sign, exponent sign, exponent digit, and an overload bit) for BCD data entry or five bytes of bit-parallel data entry. Eight data output lines are also provided for general purpose data output or control. The data input and output lines have low-power Schottky TTL receivers and drivers.

Electrical Characteristics for Output Data Lines

	Min.	Max.	Units
Output Low Voltage @ 12 mA		0.4	V
Output Low Voltage @ 24 mA		0.5	V
Output High Voltage	3.4		V
Output Low Current		24	mA
Output High Current		−15	mA

Electrical Characteristics for Input Data Lines

	Min.	Max.	Units
Input Low Voltage		0.8	V
Input High Voltage	2.0		V
Input Low Current @ Input Low Voltage = 0.4V		−0.4	mA
Input High Current @ Input High Voltage = 2.7V		20	μA

Control Lines

Five control lines provide control information between the peripheral(s) and the 98263A BCD Interface. The incoming and outgoing control lines use open collector receivers and drivers with the following electrical characteristics:

Electrical Characteristics for Control Input Lines

	Min.	Max.	Units
Input Low Voltage		0.5	V
Input High Voltage	1.9		V
Hysteresis	0.4		V
Input Low Current @ Input Low Voltage = 0.4V		−0.4	mA
Input High Current @ Input High Voltage = 2.7V		20	μA

Electrical Characteristics for Control Output Lines

	Min.	Max.	Units
Output Low Voltage @ 16 mA		0.4	V
Output Low Voltage @ 40 mA		0.7	V
Output High Voltage (open collector)		30	V
Output Low Current		40	mA
Output High Current @ Output High Voltage		0.25	mA

The control lines and their meanings are:

CTLA, CTLB Peripheral Control A and B – indicates that the Series 200 computer is requesting input data or that data is ready for output; CTLA(B) can be reset by a ready-to-busy or busy-to-ready transition on FLGA(B) or by an interface reset.

FLGA, FLGB Peripheral Flag A and B – indicates to the computer that the peripheral has completed the data transfer; also used to request peripheral interrupt when enabled.

PRESET Peripheral Reset – used to initialize a peripheral when the computer is turned on, when the RESET key or CLEAR I/O key are pressed or when writing to the 98623A reset register.

Data Formats

Two BCD data input formats are supported by the 98623A which are switch selectable on the interface card. This switch status can then be interrogated by the language system to insure that the incoming data is being formatted correctly.

Standard – Up to 8-BCD-digit signed mantissa, 1-BCD-digit signed exponent, 1-digit function code and overload indication.

Optional – Up to 4-BCD-digit signed mantissa from one device. Up to 5-BCD-digit signed mantissa with positive exponent from a second device.

Data Codes – 8421 binary-coded decimal weighting with codes 0-9 representing digits 0-9 and other codes as follows:

1010	(LF)	line feed
1011	(+)	plus
1100	(,)	comma
1101	(–)	minus
1110	(E)	exponent
1111	(.)	decimal point

Additional Input Information

Exponent, Function Code: 8421 binary-coded decimal weighting (codes 0 – 9 only), Mantissa sign, Exponent sign, Overload: 1 binary bit.

Interrupt Capability

The 98263A BCD Interface is capable of generating interrupts to the computer under the following condition:

FLGA ready & FLGB ready

Switch Configuration

The following switches can be configured on the interface card:

Select Code – the factory select code setting for the 98623A card is 11; the 98623A card can have a select code setting from 8 – 31.

Interrupt Level – the factory interupt priority level setting for the 98623A card is 3; the 98623A card can have an interrupt level setting from 3 to 6.

Input Data Line Sense – a 1-bit switch allows the input data lines to use either positive-true or negative-true logic.

CTLA(B) Line Sense – a 1-bit (ea.) switch allows the peripheral control line to use either positive-true or negative-true logic.

DFLGA(B) Line Sense – a 1-bit (ea.) switch allows the peripheral flag line to use either positive-true or negative-true logic.

CTLA(B)-2 – a 4-bit (ea.) switch allows selection of full or pulsed mode handshake.

Option Format – a 1-bit switch selects standard (one device) or optional (two devices) data format.

SIGN1(2) Line Sense – a 1-bit (ea.) switch allows the mantissa and/or exponent sign lines to use either positive-true or negative-true logic.

OVLD Line Sense – a 1-bit switch allows the overload line to use either positive-true or negative-true logic.

HP-IB Interface

In addition to the standard built-in HP-IB interface, there is an optional external 98624A HP-IB Interface Card. Both interfaces implement the IEEE 488-1980 Standard Digital Interface for Programmable Instrumentation. Both interfaces can communicate with as many as 14 HP-IB compatible insturments, connected with a maximum of 20 metres (65.6 ft.) of cable.

Data Input/Output

Eight bi-directional data lines provide data input/output.

Control Lines

DAV
NRFD — provide handshake
NDAC

Interface Management

IFC
ATN
SRQ — provide control of the interface system
REN
EOI

Interface Functions

The chart below specifies the level of implementation in terms of IEEE 488-1978 mnemonics. The Device Trigger, Device Clear and Remote/Local state responses are achieved by programming the Model 16 computer for end-of-line interrupts on those conditions.

Source Handshake	SH1
Acceptor Handshake	AH1
Talker	T6
Listener	L4
Service Request	SR1
Remote/Local	RL1
Parallel Poll	PP1
Device Clear	DC1
Device Trigger	DT1
Controller	
System control	C1
IFC & Take charge	C2
REN	C3
Respond SRQ	C4
Miscellaneous control	C5
Drivers	E2

Switch Configuration

The following switches can be configured on the interface card:

Select Code – the factory select code setting for the 98624A card is 8 (select code seven for internal HP-IB); the 98624A card can have a select code setting from 8 to 31.

Interrupt Level – the factory interrupt priority level setting for the 98624A card is 3 (internal HP-IB level is 3); the 98624A card can have an interrupt level setting from 3 to 6.

Interface Bus Address – 5-bit talker/listener address. The factory-set bus address for the 98624A is 21 decimal (21 for internal HP-IB; if the computer is not system controller then its default bus address will be 20); the 98624A card can have a bus address setting from 0 to 30.

System Controller – 1-bit switch allows the 98624A interface to act as a system controller or non-system controller. The factory setting is system controller. Internal HP-IB has a jumper or switch.

DMA Capability

(Requires the Advanced Programming Binary for BASIC.)

The internal and 98624A HP-IB interfaces are capable of carrying out DMA transfers with the Model 16 via the optional two-channel 98260A DMA Controller Card. The following DMA capabilities are supported:

- Byte Mode DMA
- Regular DMA transfer (no burst DMA)

Additional BASIC 2.0 Capabilities

Interrupt Capability

The internal and 98624A HP-IB interface are capable of generating interrupts under the following conditions:

- Controller addressed
- Talker addressed
- Listener addressed
- Service Request (SRQ) detected
- Parallel Poll configuration change
- EOI received
- Serial Poll active
- Remote/Local configuration change
- MY Address mode change
- Group Execute Trigger received
- Source handshake error
- Unrecognized universal command
- Unrecognized addressed command
- Secondary command received
- Device Clear received
- Interface Clear detected

Transfer Rates

The maximum data transfer rates for the 98624A external HP-IB interface and the internal HP-IB interface with the BASIC Language System are as follows:

	Input	Output
Handshake	42K bytes/sec.	60K bytes/sec.
With the Advanced Programming Binary:		
Interrupt, burst	40K bytes/sec	55K bytes/sec
Fast handshake	110K bytes/sec	80K bytes/sec
DMA, regular	340K bytes/sec	270K bytes/sec

Serial Interface

In addition to the built-in Serial Interface, there is an optional external 98626A Serial Interface Card. Both interfaces provide bit-serial communication between the Model 16 computer and asynchronous EIA RS-232-C (CCITT V.28/V.24) devices. Data rates range from 50 to 19200 baud (bits/sec). A variety of cabling options allow for current loop, modem and terminal connections. Model 16 Terminal Emulator Software takes advantage of this card for communication to other computers.

Transfer Rates

The maximum data rates for the Serial Interface are as follows:

	Input	Output
Handshake	19 200 baud	19 200 baud

Data Rates and Formats

All signals present at the Serial Interface's connector conform electrically to EIA RS-232-C and CCITT V.28/V.24 specifications. Data formats include 5,6,7 or 8 bits/character and 1, 1.5 or 2 stop bits. Odd, even or no parity is selectable and fixing the parity bit to 0 or 1 is also selectable.

Standard switch selectable data rates available are:

50	75	110	134.5
150	200	300	600
1200	1800	2400	3600
4800	7200	9600	19200

Under program control a data rate can be selected from a set of baud rates ranging from 50 to 19,200 bits/sec (baud).

Switch Configuration

The following switches can be configured on the interface card:

Select Code – the factory select code setting for the 98626A card and the internal Serial Interface is 9; the 98626A card can have a select code setting from 8 to 31.

Interrupt Level – the factory interrupt priority level setting for the 98626A and the internal Serial Interface Card is 3; the 98626A card can have an interrupt level setting from 3 to 6.

Parity – a 3-bit switch to enable or disable, parity, even or odd parity, or fixed '1' or fixed '0' parity bit.

Character Length – a 2-bit switch selects between 5, 6, 7 or 8 bits per character length.

Stop Bits – a 1-bit switch selects between 1 stop bit per character or 1.5 stop bits per character if the character length is 5 bits per character. If the number of bits per character is 6, 7 or 8 then the stop bits switch selects between 1 stop bit per character or 2 stop bits per character.

Modem Line Disconnect – (not available on internal Serial Interface) a 4-bit switch allowing the Ring Indicator, Data Set Ready, Clear To Send and/or Carrier Detect lines to be disconnected and tied high.

Baud Rate Select – allows power up/reset selection for the baud rate. Refer above to the baud rates available for switch selection.

Additional BASIC 2.0 Capabilities

Interrupt Capability

The Serial Interface is capable of generating interrupts to the computer. The interface can be programmed to interrupt on the following conditions:

- Receiver buffer full
- Transmitter buffer empty
- Receiver buffer overrun error
- Received character parity error
- Received character framing error
- Received break indication
- Carrier detect line change
- Clear-to-send line change
- Data-set-ready line change
- Ring indicator change from on to off

Interrupt Buffer Transfers

Interrupt buffer transfers are supported by the Advanced Programming Binary for the Serial Interface.

Color Video Interface

(Requires Color Video Interface Binary in BASIC.)

The 98627A Color Video Interface provides the interconnection to an external color monitor. This interface connects to a high-performance, high-resolution color monitor via three outputs – Red, Green/sync and Blue (RGB). The capabilities provided by this interface make it appear as a "soft plotter".

This interface, when connected to an external monitor, does *not* replace the internal CRT for all applications (e.g., program editing). It provides an enhanced soft graphics output capability.

Colors:	Eight – magenta, blue, cyan, green, yellow, red, white, black.
Resolution:	Selectable to allow a wide variety of monitors to be used in worldwide applications.
U.S. preferred:	(No flicker) – 512 x 390 at 60Hz non-interlaced vertical scan rate.
European preferred:	(No flicker) – 512 x 390 at 50Hz non-interlaced vertical scan rate.
High resolution:	(Some flicker) – 512 x 512 at 46.5Hz non-interlaced vertical scan rate.
Cables:	Four 5-ft. (1.52m) cables supplied with BNC termination.
Cable type:	75 ohm, coaxial RG-59/U – similar to Belden 9259. Note: Longer cables may not meet environmental requirements.
Tested monitors:	Barco CDCT 3/51 – 20 in. with cabinet. Conrac 7211, C19 – 19 in. with cabinet. Mitsubishi – C39-19, available through distributors. These monitors are electrically compatible with the 98627A interface; however, they may not, in all cases, pass applicable government regulations for emitted RFI.

Data Communications Interface

The 98628A Data Communications Interface provides both protocol management and electrical levels for asynchronous serial communications. This card also supports the Distributed System Network/Data Link (DSN/DL) protocol for communications to an HP 1000 series minicomputer. A terminal emulation program, which takes advantage of this card for communication to other computers, is also available.

Data Rates and Formats

All signals present at the 98628A interface card's connector conform electrically to EIA RS-232-C and CCITT V.28/V.24 specifications. Data formats include 5, 6, 7 or 8 bits/character and 1, 1.5 or 2 stop bits. Odd, even or no parity is selectable and fixing the parity bit to 0 or 1 is also selectable.

Standard data rates available with internal clocking:

50	75	110	134.5
150	200	300	600
1200	1800	2400	3600
4800	7200	9600	19200

Switch Configuration

The following switches can be configured on the interface cards:

ASYNC/DATA LINK

Select Code – the factory select code setting for the 98628 card is 20; valid select codes are 8 – 31.

Interrupt Level – the factory interrupt priority level setting for the 98628 card is 3; valid interrupt level settings are 3 – 6.

Async/Data Link – selects between Async or Data Link personality.

The settings listed below are not all switch selectable; however, all values are selectable through the CONTROL statement. Values selected through the CONTROL statement override the switch settings.

ASYNC

These settings are active when the ASYNC/DATA LINK switch is set to its ASYNC position.

Parity – Bits/Char – a two-bit switch which selects between the following Parity – Bits/Char combinations: None – 8, None – 7, Odd – 7, Even – 7.

Hardware Handshake – a 2-bit switch which selects Handshake Off, Non-modem connection; Full Duplex, Modem connection; Half Duplex, Modem connection; and Handshake On, Non-modem connection.

Baud Rate – Stop Bit – a 3-bit switch which selects between the following combinations of baud rates/stop bit settings: 110 – 2, 150 – 2, 300 – 1, 600 – 1, 1200 – 1, 2400 – 1, 4800 – 1, 9600 – 1.

DATA LINK

These settings are active when the ASYNC/DATA LINK switch is set to its DATA LINK position.

DID – a 3-bit switch which selects the following value for the 98628's device address: @, A, B, C, D, E, F or G.

Baud Rate – a 2-bit switch which selects the following baud rates: 300, 1200, 9600 or 19200.

Hardware Handshake – a 2-bit switch which selects between Handshake Off, Non-modem connection; Full Duplex, Modem connection; Half Duplex, Modem connection; and Handshake On, Non-modem connection.

Electrical Specifications

Card power consumption: +5V at 715mA typical
+12V at 37mA typical
−12V at 60mA typical

Pod power consumption (supplied by computer):	+5V typical	+12V typical	−12V typical
Current loop interface (13266A)	200mA	90mA	80mA
300 baud modem (13265A)	100mA	45mA	45mA
Data link adapter (13264A)	30mA	160mA	23mA

Electrical interface capabilities:
RS-232-C, V.24/V.28
RS-449
RS-423, V.10

Additional BASIC 2.0 Capabilities

Interrupt Capability

The 98628A Serial Interface Card is capable of generating interrupts to the computer. The interface can be programmed to interrupt on the following conditions:

ASYNC
- Data or control block available
- Prompt received
- Framing and/or parity error
- Modem line change (DSR, DCD, CTS, RI)
- No activity timeout
- Lost carrier or connection timeout
- End-of-line received
- Break received

DATA LINK
- Data block available
- Space available for new transmission block
- Receive or transmit error
- Modem line change (DSR, DCD, DTS, RI)
- No activity timeout
- Lost carrier or connection timeout

Interrupt Buffer Transfers

Interrupt buffer transfers are supported by the Advanced Programming Binary for the 98628A Data Communications Interface.

DMA Controller Card

(Requires the Advanced Programming Binary for BASIC.)

The 98620A DMA Controller Card enhances the Model 16's interfacing capability by providing two DMA channels for I/O data transfers. This high-speed I/O capability works with the 98622A GPIO, 98624A HP-IB and internal HP-IB interfaces. Although the 98620A can accomodate DMA transfer rates up to memory cycle rates (approx. 1.2M transfers/sec.) lower DMA rates can be expected since actual rates are dependent on a number of factors. The typical transfer rate for the 98622A GPIO Interface is approximately 750K transfers per second, and for the 98624A HP-IB Interface approximately 330K transfers per second.

Model 16 Technical Specifications

These specifications apply to the Model 16 hardware as supported by the BASIC 2.0 Language System, BASIC Extensions 2.0, and the Pascal 2.0 Language System.

Processor*

Type:	Motorola MC68000
Clock frequency:	8 MHz
Internal architecture:	32-bit data and address registers
Address range:	16M bytes
Data bus:	16-bit asynchronous
Instruction types:	56
Major data types:	5
Addressing modes:	14
Interrupt levels:	One non-maskable and 6 maskable

* DMA available with the 98620A DMA controller (requires the Advanced Programming Binary in BASIC).

Rotary Control Knob*

Pulse resolution:	120 pulses per revolution (nominal)
Pulse count range:	−128 to 127 net pulses since last interrupt
Pulse count sign	
Positive:	Net clockwise
Negative:	Net counterclockwise
Interrupt generation period:	.01 sec to 2.55 sec

* Not programmable in Pascal.

Clock and Timers*

Real-time clock	
Resolution:	10 msecs
Accuracy:	50 ppm (4.3 sec./day)
Power-on reset:	Midnight, January 1
Timers	
Match interrupt:	Match on time of day, 0.00 to 84600.00 seconds
Delay interrupt:	10 msecs to 1.94 days
Cycled interrupt:	10 msecs to 1.94 days

* Not programmable in Pascal; timers supported by the Advanced Programming Binary in BASIC.

Beeper*

Range (nominal):	81.375 Hz to 5208 Hz
Resolution:	81.375 Hz nominal
Duration:	.01 sec to 2.55 sec

* Not programmable in Pascal.

Environmental Range*

Operating temperature:	0°C to 55°C
Humidity:	5 to 95% R.H. non-condensing
Maximum wet-bulb temperature:	40°C
Storage environment:	−40°C to 75°C
Maximum altitude:	4572m (15 000 ft.)
EMI:	Conducted and radiated interference meets VDE 0730, CISPR publication 11, and FCC class B standards.
Line transient spike immunity (1 nsec. rise, 800 nsec. duration):	1KV
Additional regulatory compliance:	UL, CSA, IEC, SEV, FEI

* System environmental range also depends on peripherals chosen.

CRT Display

Size:	229 mm (9 in.) diagonal
Alphanumeric capacity	
On screen:	25 lines x 80 characters
Total scrolling:	39 lines x 80 characters, 3 120 characters
Character height:	1.2 mm wide x 2.8 mm high (.05 in. x .11 in.) capital letters
Display enhancements:	None
Graphics capability	
Resolution:	400 dots horizontal x 300 dots vertical
Density:	25 dots/cm. (63 dots/in.)
Raster size:	160 mm x 120 mm (6.3 in. x 4.72 in.)
Display buffering:	Dedicated 3K byte alpha buffer, 16K byte graphics buffer (can be displayed simultaneously)
Soft-key labeling:	Up to 10 user-definable soft-key labels, 14 characters per label
Character set:	256 characters
Character font:	7 x 8 dot character matrix in a 10 x 12 character cell
Intensity:	Adjustable up to 30 ft-lamberts
Refresh rate:	60 Hz standard with 50 Hz user selectable
Implosion protection:	Tension band
Tube phosphor:	P4
Cursor:	Blinking underline

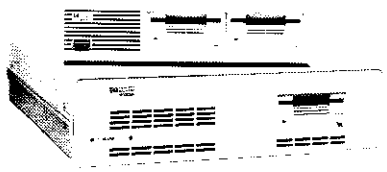
Power Requirements

Source consumption (A max. @ ~V, switch selectable):	1.5A @ 90 – 125 V 0.8 A @ 198 – 250 V
Line frequency:	48 – 66 Hz
Power	
Watts max.:	80 W
Btu/hr.:	280

Physical

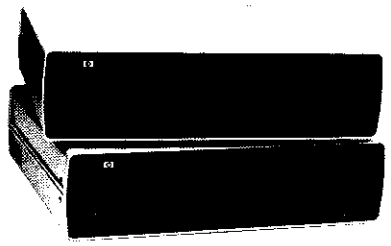
Height:	282 mm (11.10 in.)
Width:	315 mm (12.40 in.)
Depth:	488 mm (19.21 in.)
Cube:	.043 m^3 (1.53 $ft.^3$)
Net weight:	8.9 kg (19.5 lb.)
Shipping weight:	11.3 kg (25.0 lb.)

Model 16 Peripherals



HP 9121D/HP 9133A

HP 82901M/HP 82902M



HP 9134A/HP 9135A

Mass Storage

HP provides a family of low-cost, random access disc drives, designed to match the Model 16's application needs. Choose from entry-level 3½-in. and 5¼-in. flexible disc drives, and 5¼-in. Winchester disc drives for high speed and large capacity.

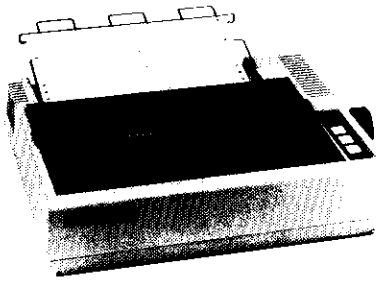
Unit Specifications

Power Requirements	9121S/D	82901M 82902M	9134A	9133A	9135A
Source (selected by rear panel switch)	86 – 127 Vac (115 Vac Line) 200 – 254 Vac (230 Vac Line)	90 – 127 Vac (115 Vac Line) 200 – 250 Vac (230 Vac Line)	100 Vac ±10% 120 Vac ±10% 220 Vac ±10% 240 Vac ±10%	100 Vac ±10% 120 Vac ±10% 220 Vac ±10% 240 Vac ±10%	100 Vac ±10% 120 Vac ±10% 220 Vac ±10% 240 Vac ±10%
Line Frequency (±2%)	48 – 66 Hz	48 – 66 Hz	48 – 66 Hz	48 – 66 Hz	48 – 66 Hz
Power Consumption (max.)	67W (9121D)	80W (82901M) 55W (82902M)	140W	140W	140W
Environmental Range					
Operating Temperature	5° to 45°C (41° to 113°F)	10° to 40°C (50° to 104°F)	10° to 40°C (50° to 104°F)	10° to 40°C (50° to 104°F)	10° to 40°C (50° to 104°F)
Storage Temperature	−40° to 60°C (−40° to 140°F)	−40° to 60°C (−40° to 140°F)	−40° to 60°C (−40° to 140°F)	−40° to 60°C (−40° to 140°F)	−40° to 60°C (−40° to 140°F)
Operating Humidity (non-condensing) 25.5 max. wet bulb temperature	20% to 80%	20% to 80%	8% to 80%	20% to 80%	20% to 80%
Storage Humidity (non-condensing)	5% to 95%	5% to 95%	5% to 95%	5% to 95%	5% to 95%
Operating Altitude	0 to 4572m 0 to 15000 ft.	−304 to 4572m −1000 to 15000	0 to 4572m 0 to 15000 ft.	0 to 4572m 0 to 15000 ft.	0 to 4572m 0 to 15000 ft.
Storage Altitude	−304 to 15240m −1000 to 50000 ft.	−304 to 15240m −1000 to 50000 ft.	−304 to 15240m −1000 to 50000 ft.	−304 to 15240m −1000 to 50000 ft.	−304 to 15240m −1000 to 50000 ft.
Size/Weight					
Height	76 mm (2.99 in.)	109 mm (4.31 in.)	130 mm (5.1 in.)	130 mm (5.1 in.)	130 mm (5.1 in.)
Width			424 mm (16.7 in.)	424 mm (16.7 in.)	424 mm (16.7 in.)
Dual	325 mm (12.8 in.)	425 mm (16.75 in.)			
Single	325 mm (12.8 in.)	279 mm (11 in.)			
Depth	285 mm (11.2 in.)	374 mm (14.74 in.)	475 mm (18.7 in.)	475 mm (18.7 in.)	475 mm (18.7 in.)
Net Weight			13.4 kg (29.5 lbs.)	14.5 kg (32 lbs.)	15.5 kg (34.1 lbs.)
Dual	4.5 kg (10 lbs.)	9.07 kg (20 lbs.)			
Single	3.8 kg (8.4 lbs.)	7.71 kg (17 lbs.)			
Shipping Weight			16.8 kg (37 lbs.)	17.6 kg (38.9 lbs.)	18.6 kg (41 lbs.)
Dual	5.9 kg (13 lbs.)	12.7 kg (28 lbs.)			
Single	4.7 kg (10.4 lbs.)	9.98 kg (22 lbs.)			

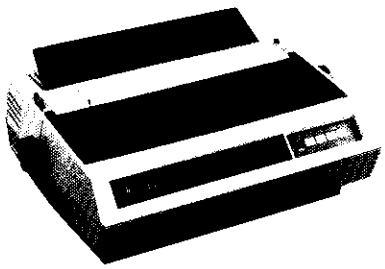
Drive Specifications

Technical	3½-in. Flexible Discs 9121D, 9121S, 9133A	5¼-in. Flexible Discs 82901M, 82902M, 9135A	5¼-in. Winchesters 9133A, 9134A, 9135A
Formatted Capacity	540K (dual)	540K (dual)	4.6M
Bytes per Unit	270K (single)	270K (single)	4.8M (Opt. 010)
Bytes per Sector	256	256	256
Sectors per Track	16	16	30 (31 for Opt. 010)
Tracks per Surface	66	33	150 (152 for Opt. 010)
Recording Surfaces/Disc	1	2	2 (4 surfaces per unit)
Tracks per Inch	135/in.	48/in.	255/in.
Recording Format	Double density	Double density	—
Max. Sustained Transfer Rate*	17.8 kb/sec	6.8 kb/sec	50 kb/sec
Average Access Time	415 msec (on), 1 415 msec (off)	187 msec (on), 435 msec (off)	168 msec†
Rotational Speed	600 rpm	300 rpm	3600 rpm
Media Part Number (10 pack)	HP 92191A	HP 92190A	—

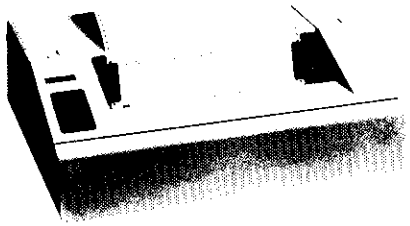
* Mainframe dependent. † 60 msec within one volume.



HP 82905B



HP 2602A



HP 2631B

Printers

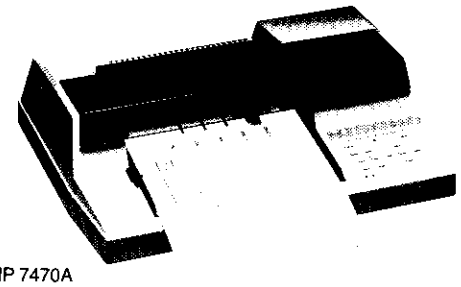
For high-quality, hard-copy output of your applications, you can choose from a selection of HP printers: the HP 82905B low-cost impact printer; the HP 2670 Series thermal printers (including the HP 2673A intelligent graphics printer); and the HP 2602A daisy wheel printer.

Printer Specifications

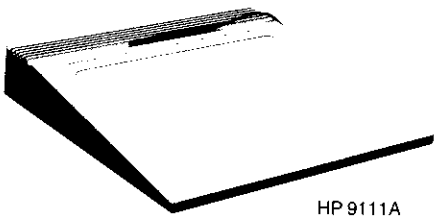
	HP 82905B*	HP 2602A	HP 2670 Series
Printing Technique	Dot matrix impact	Daisy wheel	Dot matrix thermal
Print Speed	80 characters/sec bidirectional; logic seeking in text mode	25 peak + 20 CPS (typical) using average English Shannon text bidirectional	120 characters/sec bidirectional; logic seeking in text mode
Character Structure	Text mode: 9 x 9 dot matrix Graphics mode: 72 x 60 or 72 x 120 dots/in.	Full-formed character	9 x 15 dot matrix
Print Pitch (CPI) Line Length (characters) Normal Normal Expanded Compressed Compressed Expanded Normal Emphasized	**Print Pitch** / **Line Length** 10.00 / 80 5.00 / 40 16.50 / 132 8.25 / 66 10.00 / 80	Print pitch: 10 or 12 CPI Print line: 33.58 cm (13.2 in.) 132/158 columns in 10 or 12 pitch, respectively Proportional spacing	**Print Pitch** / **Line Length** 10.00 / 80 5.00 / 40(2673A) 16.20 / 132
Character Set	96 U.S. ASCII Roman character set	98 character print wheels	128 U.S. ASCII Line drawing Roman Extension (international characters, 8-bit mode) ISO (international characters, 7-bit mode) (HP 2673A only)
Forms Handling	Forms tractors Programmable page length Automatic perforation skip Variable vertical line spacing: 1/6-in. standard; programmable to various line densities	Multicopy controls: 1 to 5 part forms Paper thickness: 0.061 cm (0.024 in.) max. Forms width: 38.74 cm (15.2 in.)	Form feed button Margin control Auto-page mode (HP 2673A) Tabs (HP 2673A) Non-volatile memory stores configuration information entered via the control panel (HP 2673A)
Forms Specifications	Paper width range: 10.2 cm (4 in.) to 25.4 cm (10 in.) Paper thickness: 0.3 mm (0.01 in.) max. Multipart forms: original plus 2 copies		Thermal paper width: 21.6 cm (8.5 in.) Paper options include fan-folded, page perforated; roll; or roll page perforated

* 82905B is not recommended for use in Pascal Program development.

	HP 82905B	HP 2602A	HP 2670 Series
Other Printing Features	Characters per line: 40, 66, 80, 132 Line feed rate: 5 lines/sec		Underlining character enhancement Framing character enhancement (HP 2673A) Triple-pass mode (HP 2673A)
Operating Requirements Source	100 Vac (Opt. 001) 120 Vac (Opt. 002) 220 Vac (Opt. 003) 240 Vac (Opt. 004)	(+10%, −15%): 120, 220, 240 Vac	(+5%, −10%): 100, 120, 220, 240 Vac, switch selectable
Frequency	50 − 60 Hz	49.5 − 61 Hz	47.5 − 66 Hz
Power Consumption	100 Watts max.	90 Watts typical printing	15 Watts max. non-printing and 50 Watts max. printing (HP 2671A), HP 2671G) 30 Watts max. non-printing and 75 Watts max. printing (HP 2673A)
Operating Temperature	5° to 35°C (41° to 95°F)	7° to 41°C (45° to 105°F)	0° to 55°C (32° to 131°F); thermal paper limited to 40° (104°F)
Humidity (non-condensing)	10% to 90%	10% to 80%	20% to 95%
Size Height Width Depth	 10.7 cm (4.2 in.) 37.4 cm (14.7 in.) 30.5 cm (12.0 in.)	 23.49 cm (9.25 in.) 61.59 cm (24.25 in.) 45.88 cm (17.75 in.)	 10.5 cm (4.1 in.) 42.8 cm (16.9 in.) 42.4 cm (16.7 in.)
Weight Net Shipping	 5.5 kg (12 lbs.) 8.2 kg (18 lbs.)	 23 kg (50 lbs.)	 2671A, 2671G: 6.9 kg (16 lbs.) 2673A: 8.3 kg (19 lbs.) 2671A, 2671G: 12.7 kg (28 lbs.) 2673A: 14.1 kg (31 lbs.)
Electromagnetic Compatibility	FCC Class B certified peripherals	FCC Class B certified peripherals	FCC Class B certified peripherals
Interface Required	HP-IB	HP-IB	HP-IB
Other			2671G, 2673A raster graphics Type: unidirectional raster graphics copy; 90 dots/in. horizontal and vertical resolution; 720 dots across a raster row Output format: precise dot-for-dot copies from HP raster devices; offsets, windowing, and auto-centering (2673A)



HP 7470A



HP 9111A

Graphics

The HP 7470A Graphics Plotter is a low-cost plotter that gives you professional-looking, hard-copy graphics. You can easily turn information into attractive, colorful visuals for use in decision making and communication. The HP 9111A Graphics Tablet and Model 16 are ideal for creating graphics, entering graphics data directly from source documents, and entering commands from a customized menu.

HP 7470A Specifications

Plotting Area	
Y-axis	190 mm (7.5 in.)
X-axis	273 mm (10.7 in.) metric setting 258 mm (10.2 in.) English setting
Media sizes	8½ x 11-in. (ANSI A) 210 x 297 mm (ISO A4)
Resolution	
Smallest addressable step size	0.025 mm (0.001 in.)
Repeatability	
With a given pen	0.1 mm (0.004 in.)
From pen to pen	0.2 mm (0.008 in.)
Pen Velocity	
Pen down	Max.: 38.1 cm/sec (15 in./sec) Programmable: 1 to 38 cm/sec in 1 cm/sec increments
Pen up	50.8 cm/sec (20 in./sec)
Acceleration	Approximately 2 G's
Interface Required	HP-IB (IEEE 488-1978)
Operating Requirements	
Source	100, 120, 220, 240 Vac, −10%, +5%
Frequency	48 − 66 Hz
Power consumption	25 Watts max.
Size and Weight	
Height	12.7 cm (5 in.)
Width	43.2 cm (17 in.)
Depth	34.3 cm (13.5 in.)
Weight	
Net	6.1 kg (13.5 lbs.)
Shipping	12.6 kg (28 lbs.)

HP 9111A Specifications

Resolution	0.100 mm (0.004 in.)
Accuracy	±0.600 cm (0.0236 in.) at 20°C for each measuring point; change of 0.004 mm for each °C deviation from 20°C
Repeatability	± resolution unit
Data rate	Programmable from 1 to 60 coordinate pairs/sec actual rate ± .2 Hz from programmed rate
Data format	ASCII or binary X,Y coordinate data
Active digitizing area	218.5 x 300.8 mm (8.6 x 11.8 in.); can be extended to include the area occupied by the 16 soft keys
Document material	Single sheet, electrically nonconductive, homogenous, less than 0.5 mm thick
Interface	HP-IB (IEEE 488-1978)
Operating Requirements	
Voltage options	100, 120, 220, 240 Vac
Frequency	50/60 Hz
Consumption	25 Watts max.
Temperature	0° to 55°C (32° to 131°F)
Relative humidity	5% to 90% at 40°C, noncondensing
Size and Weight	
Height	85 mm (3.35 in.)
Width	440 mm (17.3 in.)
Depth	440 mm (17.3 in.)
Weight	
Net	5.8 kg (12.8 lbs.)
Shipping	10.8 kg (23.8 lbs.)

Configuring a Model 16

Use the information below to easily configure the right Model 16 system for your application needs.

The Model 16 comes in two versions to suit your price/performance requirements: the Model 16S bundled system and the Model 16A minimum configuration. Each has built-in RAM on its processor board, built-in HP-IB and Serial Interfaces, and two slots for further expansion. Differences are:

	Model 16A	Model 16S
Processor board RAM	128K bytes	256K bytes
Additional 256K RAM boards	0	1
Total RAM included	128K bytes	512K bytes
Boot ROM	Boot-up from 3½-in. or 5¼-in. flexible disc drives only	Boot-up from all supported mass storage devices
Graphics	Standard, but can be deleted	Standard
Ram-based BASIC 2.0	Ordered separately	Included
Built-in HP-IB Interface (with 2-meter cable)	Yes	Yes
Built-in Serial Interface (cable not included)	Yes	Yes
Expansion slots built-in	0	2
Expansion slots available	2	1*

* One of the Model 16S' expansion slots is used for the additional 256K byte RAM board that comes standard.

Built-In Interfaces

To make a useable system, a mass storage device must be connected to the Model 16. One 2-meter HP-IB interconnect cable (10833B) is shipped with the Model 16 for this purpose. Every additional device (e.g., printer, plotter, etc.) which is to be connected to the built-in HP-IB interface will also need an interconnect cable (10833A/B/C/D).

The built-in Serial Interface is not shipped with a cable, as there are two different cables available. Order separately the 5061-4215 cable with the DTE connector (male) or the 5061-4216 cable with the DCE connector (female).

Expansion Slots

There are four uses for the Model 16's expansion slots: RAM memory, ROM languages, a DMA controller or an I/O card. RAM, ROM and DMA each use one expansion slot. An I/O card uses one slot, but covers the adjacent slot above it. The adjacent slot may still be used for RAM, ROM or DMA. Therefore, I/O cards may only be used in the bottom slot, while RAM, ROM and DMA may be used in either slot. (Only one DMA controller card is allowed per machine, however.) This is shown graphically on the next page.

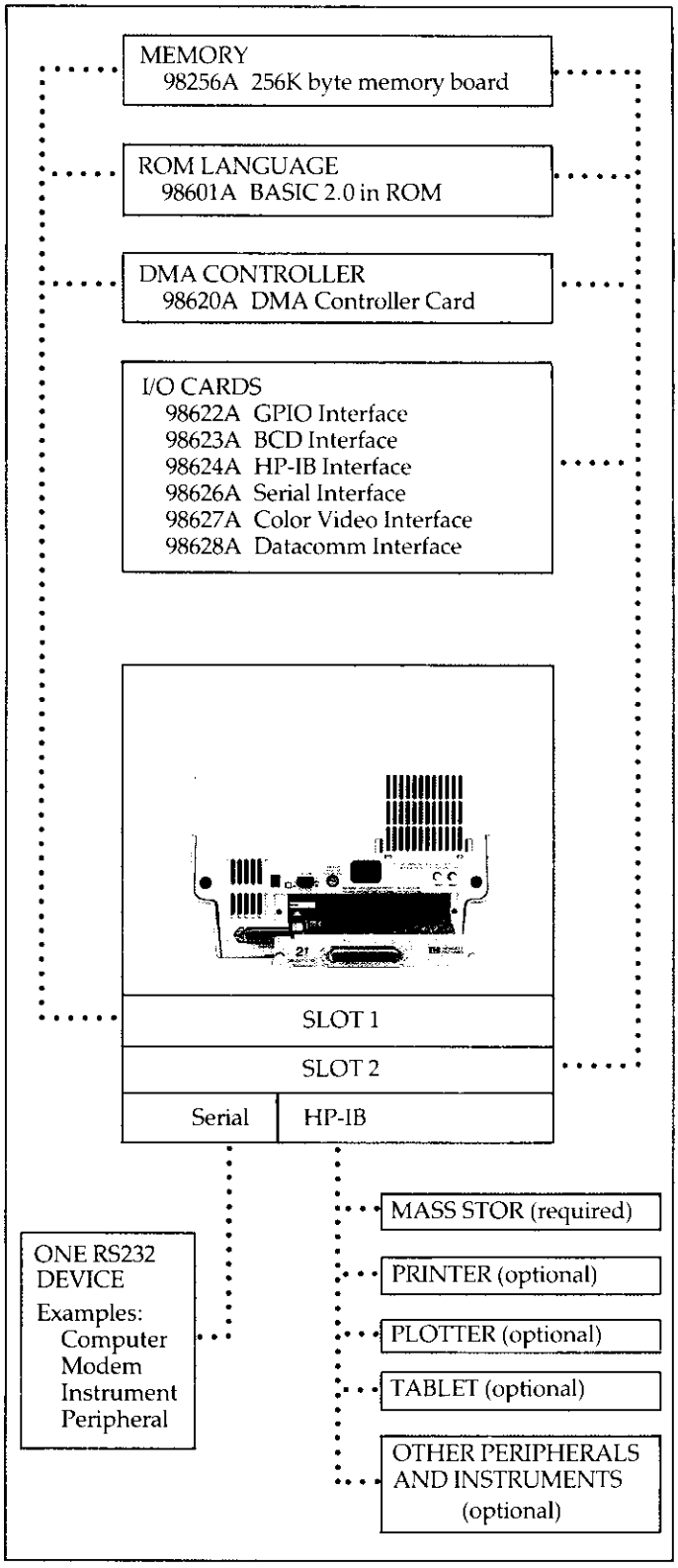
Memory Requirements

Both RAM- and ROM-based language systems use some RAM. RAM-based systems load entirely into RAM, while ROM-based systems "steal" some RAM for a workspace area. In the case of BASIC, the system memory requirements must be subtracted from the total memory in the configuration to find the maximum program size. These requirements are summarized below:

Language System	Memory Required (Approx.)
RAM-based BASIC 2.0 (98611A)	277K bytes
ROM-based BASIC 2.0 (98601A)	21K bytes
RAM-based BASIC Extensions 2.0 (98612A)*	
Advanced Programming Binary	170K bytes
Color Video Interface Binary	21K bytes

* BASIC Extensions 2.0 requires that ROM- or RAM-based BASIC 2.0 be present.

Pascal 2.0 (98615A) is a RAM-based system composed of a kernel and several modules that load on top of it. The different modules have different memory requirements, and the kernel itself is user-modifiable. To have enough room for loading the modules and creating a program, a minimum recommended Pascal program development environment is 512K bytes. In a run-only environment, however, the minimum memory needs may be reduced to as low as approximately 130K bytes. This will vary depending on the particular requirements of the application.


MEMORY
98256A 256K byte memory board
ROM LANGUAGE
98601A BASIC 2.0 in ROM
DMA CONTROLLER
98620A DMA Controller Card
I/O CARDS
98622A GPIO Interface
98623A BCD Interface
98624A HP-IB Interface
98626A Serial Interface
98627A Color Video Interface
98628A Datacomm Interface
SLOT 1
SLOT 2
Serial
HP-IB
ONE RS232 DEVICE
Examples:
Computer
Modem
Instrument
Peripheral
MASS STOR (required)
PRINTER (optional)
PLOTTER (optional)
TABLET (optional)
OTHER PERIPHERALS AND INSTRUMENTS (optional)

For More Information

For additional information and a personal demonstration of Hewlett-Packard's high-powered 16-bit personal technical computer, HP Series 200, Model 16, and other HP personal computing systems, see your nearest HP dealer. In the United States, to locate the HP dealer nearest you, call TOLL FREE 800-547-3400. (In Oregon, Alaska, and Hawaii, call 503-758-1010.) TTY users with hearing or speech impairments, please dial 503-758-5566.

For other worldwide HP locations, contact:

United States
Hewlett-Packard Company
Personal Computer Division
1010 NE Circle Blvd.
Corvallis, OR 97330

Australia
Hewlett-Packard
Australia (Pty.) Ltd.
31-41 Joseph Street
Blackburn, Victoria 3130

Canada
Hewlett-Packard (Canada) Ltd.
6877 Goreway Drive
Mississauga, Ontario
L4V1M8

Europe, North Africa, Middle East
Hewlett-Packard SA
7, rue du Bois du Lan
P.O. Box
CH 1217 Meyrin 2
Geneva, Switzerland

Hong Kong
Hewlett-Packard Hong Kong, Ltd.
5th and 6th Floors
Sun Hung Kai Centre
30 Harbour Road
Hong Kong

Japan
Yokogawa-Hewlett-Packard, Ltd.
29-21, Takaido-Higashi
3-chome
Suginami-ku, Tokyo 168

New Zealand
Hewlett-Packard (N.Z.) Ltd.
P.O. Box 9443
Kilbirnie
Wellington 3

Singapore
Hewlett-Packard
Singapore (Pty.) Ltd.
Alexandra Post Office
P.O. Box 58, Singapore
9115

South Africa
Hewlett-Packard SA (Pty.) Ltd.
Private Bag, Wendywood
Sandton, Transvaal, 2144
South Africa

Other Countries
Hewlett-Packard
Intercontinental
3495 Deer Creek Road
Palo Alto, CA 94304
U.S.A.

5953-4629

HEWLETT PACKARD

Printed in U.S.A. 11/82 (50M)