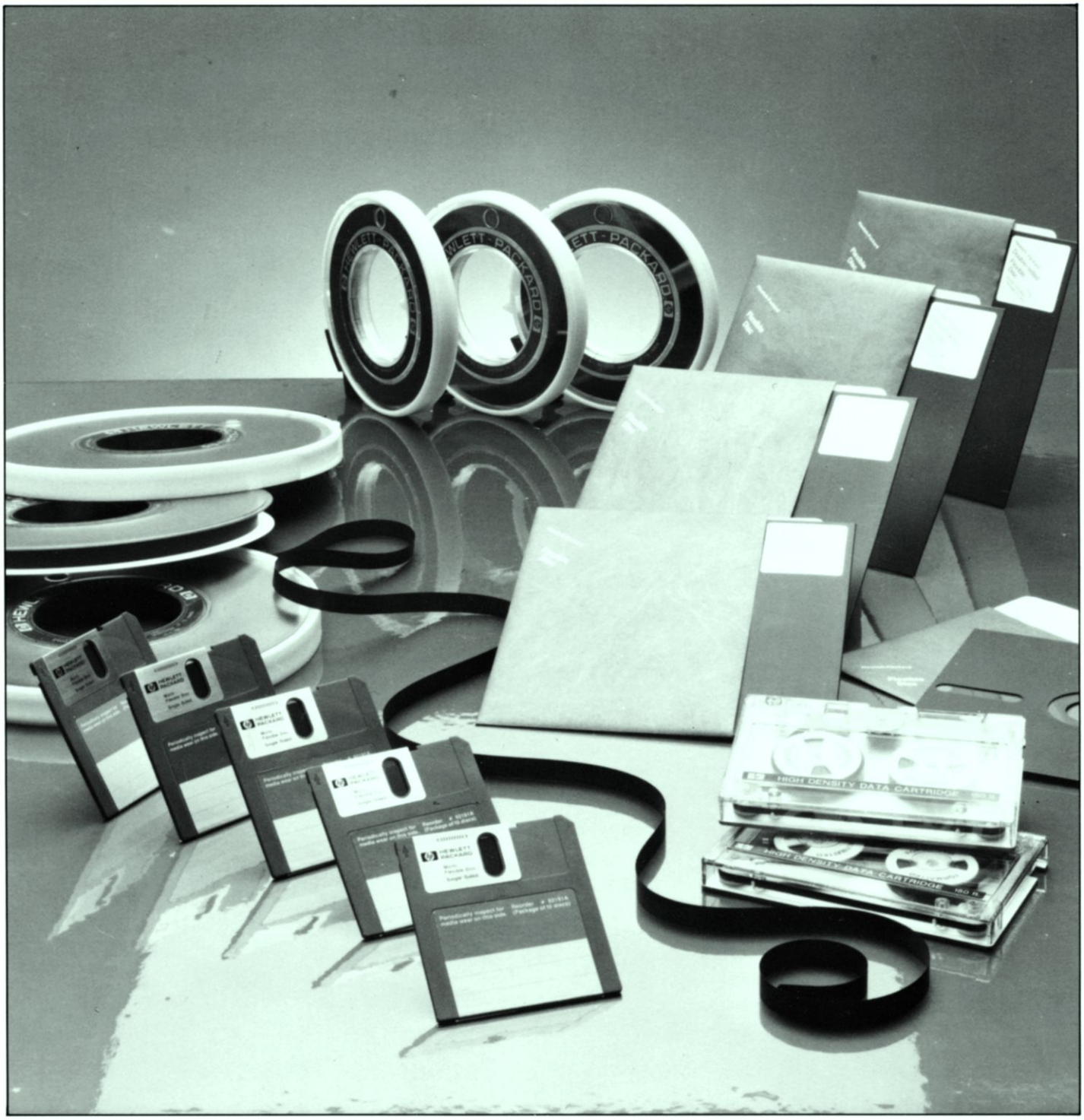
HP 1000 Computer Systems

HEWLETT PACKARD

Software Technical Data, Volume II



Documentation Summary of HP 1000 Data Books

I/O ARCHITECTURE	Distributed Intelligence	Centralized Intelligence
COMPUTER SERIES	A	E/F
SYSTEMS AND COMPUTERS	HP 1000 A-Series Hardware Technical Data	HP 1000 E/F-Series Hardware Technical Data
INTERFACES	HP 1000 A-Series Interfaces Technical Data	HP 1000 E/F-Series Interfaces Technical Data
SOFTWARE	HP 1000 Software Technical Data, Volumes I and II	
COMMUNICATIONS	HP 1000 Computer Systems Communications Products Technical Data book	
PERIPHERALS	HP 1000 Peripherals Selection Guide	
MEASUREMENT AND CONTROL	Control/1000 Technical Information Package	ical Data
	HP 2250 Measurement and Control Systems Technical Data	

NOTE: Data book supplements containing new or revised information are sometimes printed between data book revisions. Ask your Hewlett-Packard representative for the current data book or supplement in your area of interest.

HP 1000 Program Development Software Information Locator

Standard Software Provided with RTE Operating Systems

NOTE: All other software is located (or referenced) in the HP 1000 Computer Systems Technical Data book, Volume I (5953-8710).

HP 1000 FORTRAN 77

HEWLETT PACKARD

For Program Development
in RTE-6/VM or RTE-A

product number 92836A

Hewlett-Packard's HP 1000 FORTRAN 77 is a full implementation of the latest ANSI Standard for FORTRAN. This powerful language processor includes many extensions for compatibility with mainframe implementations of FORTRAN to simplify program migration to the HP 1000.

FORTRAN 77 operates under HP 1000's most advanced real-time executive operating systems, RTE-6/VM and RTE-A, taking full advantage of the extended features available such as large data and program capabilities. In addition, FORTRAN 77 programmers can directly call external routines written in Pascal or Macro/1000 Assembler to maximize flexibility and performance yet minimize program development efforts.

FORTRAN 77 supports both the RTE-A Command Interpreter file system, and code and data separation for transparent support of large programs under RTE-A/VC+.

Features

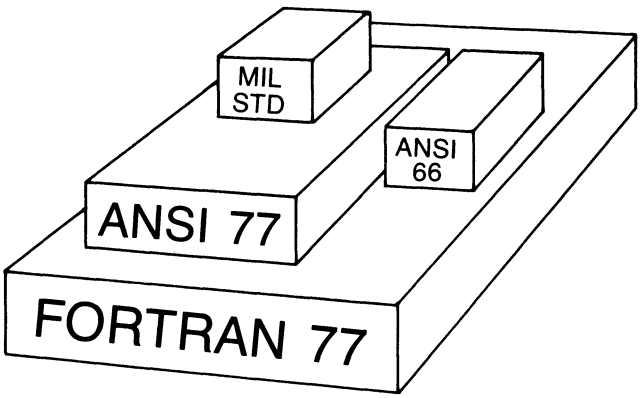
- **Full ANSI 77 FORTRAN (X3.9-1978)**
- **Compatibility with ANSI 66 and HP's FORTRAN 4X**
- **Fast compilation**
- **Supports variables and arrays using up to 128M bytes of virtual storage**
- **Generation of CDS output code for RTE-A/VC+ systems**
- **Full MIL-STD-1753 extensions**
- **Extensions to enhance mainframe compatibility**
- **Long names with up to 16 significant characters**
- **Double complex data type**
- **System programming extensions**
- **Transparent access to remote files in a distributed systems network**
- **Local optimization for efficient generated code**
- **Compatability with other HP 1000 software subsystems such as Graphics/1000-II, DS/1000-IV, and Image/1000**

How FORTRAN 77 provides more powerful programming

Structured Control Flow For Simplified Coding

IF THEN ELSE construct: IF(...) THEN, ELSE, ELSEIF(...) THEN, and ENDIF statements are included for enhanced readability.

Faster, more flexible DO loops: can be specified for zero trip. Trip count is pre-computed, and the index can be integer, real or double.



Alternate returns: allows program flow to return to a statement other than the one following the subroutine statement call (e.g. for error processing).

Alternate entries: allows multiple entry points to a program, increasing program modularity.

STOP and PAUSE: statements display character strings as well as numbers.

Powerful Data Manipulation

CHARACTER data type: replacing Hollerith constants and data in ANSI 66, CHARACTER data offers easy string manipulation.

Expressions valid in most contexts: In most places where a variable or constant is legal, so is an expression.

Constant expressions in declarations: With PARAMETER statements, programs can be 'parameterized' so that array sizes and other constant values need not be hard coded.

Variable dimensions as run-time expressions: Array upper and lower bounds (for parameters) can be computed from other parameters and COMMON variables.

Additional intrinsic functions: included are functions for rounding, various transcendentals and string manipulation.

Generic intrinsic functions: single names may be used for all versions of an intrinsic (e.g. SIN in place of DSIN) for most FORTRAN library functions.

General-Purpose I/O

OPEN, CLOSE, INQUIRE statements: for disc files and devices. INQUIRE can be used to investigate properties of a file.

Direct access I/O: allows any record in a direct-access file to be read or written quickly by record number.

List-directed I/O: Input data can be 'free-field' and appropriate formats are chosen for output data. In both cases, FORMAT statements are unnecessary.

Internal files: allows the power of FORMAT number conversions to be used with character variables or arrays like the ACSII 'buffer'.

Error handling (ERR=, END=, IOSTAT=): An error specifier enables more control over actions to be taken when an I/O error is detected.

Additional format capabilities: For example, control over printing of leading zeros.

READ and PRINT to standard devices: Logical unit number can be defaulted to (possibly different) standard input and output devices.

New Declarations

PARAMETER statement: to give names to constants. Allows 'parameterization' of programs, including values that can control conditional compilation.

IMPLICIT statement: used to change the default implicit types within a program unit.

SAVE statement: causes the values of local variables in a subprogram to be saved from one call to the next, even if the subprogram is in a disc-resident segment.

Implied DO loops in DATA statements: Portions of arrays can be initialized and subscripts of arrays within the 'DO' can be expressions.

INTRINSIC statement: specifies that an intrinsic function is to be passed to a subprogram. Names declared EXTERNAL will not be intrinsics.

PROGRAM statement: optional statement to name the main program.

Array declarators: Upper and lower bounds of array dimensions can be specified in array declarators.

Array dimensions: Arrays can have up to seven dimensions.

HP 1000 Extensions to ANSI 77 FORTRAN

Mainframe Compatibility

Selection of ANSI 66 or 77 semantics: Where FORTRAN 77 is incompatible with the previous standard or common industry practice, HP's compiler provides alternative interpretations on a user-selectable basis.

Long names: up to 16 significant characters.

Embedded underscore: allowed in symbolic names.

ENCODE and DECODE statements: provide memory-to-memory formatting.

IBM-style direct access READ and WRITE: may be formatted or unformatted.

Byte-length data types: e.g. INTEGER*4.

Embedded comments: An exclamation point can be placed after any statement signifying an end-of-line comment.

Extended-range DO loops: permit transfer of control out of a DO-loop and then back in.

Hollerith data manipulation: Hollerith data can be used in DATA statements, READ/WRITE, and as arithmetic operands.

Double precision COMPLEX data types: as approved by the IFIP WG 2.5 on Numerical Software.

Quoted Hollerith constants in DATA statements: Both character and non-character variables can be initialized with ASCII in DATA statements.

Compliance With MIL-STD 1753

DO WHILE looping construct: allows execution of a DO-loop while a logical expression holds true.

END DO: used as a terminal statement of a DO-loop The matching DO may omit the statement number.

Nested INCLUDE: allows inclusion of text (e.g. COMMON declarations) from another file. INCLUDE is offered as both a statement and directive, and permits nonrecursive nesting.

Bit manipulation intrinsics: Functions for logical and circular shifting, set/clear/test bit, bit field extraction, bit field move, and masking operations.

IMPLICIT NONE statement: removes implicit types so that all variables must be explicitly typed. All implicit types assume the normal default values.

Octal and hexadecimal constants in DATA statements.

System Programming Capabilities

Conditional compilation is achieved through the coordinated use of named constants, constant folding, and dead code removal.

Aliasing of subprogram names to allow special characters.

Specification of non-standard calling sequences.

Extension of .AND.,.OR.,.NOT.,.EQV.,.NEQV., .XOR. to integer data.

Bit shifting, extraction and testing intrinsics.
EQUIVALENCEing of character and non-character data.

Intrinsic function to get actual number of parameters passed to a subprogram.

HP-IB (IEEE 488) device control via secondary addressing and control buffers.

Virtual Data Capabilities

Local variables and common blocks may be used in up to 128M bytes of virtual memory.

Double integer subcripts may be used to access arrays with dimensions greater than 32767 elements.

Program Form

Lower case accepted (mapped to upper case)

Descriptive error messages

Optional compilation of lines beginning with "D"

Integers may be defaulted to single or double length

Functional specifications

Applicable Standard

HP 1000 FORTRAN 77 is a superset of both ANSI FORTRAN X3.9-1978 and X3.9-1966.

Environment

Operating system: HP 1000 Computer Systems operating under RTE-6/VM or RTE-A that meet the minimum hardware requirements of the operating system, plus a list device.

Memory requirements:

Program size	Memory required
Up to 2000 lines	46 Kilobytes (23 pages)
Up to 5000 lines	64 Kilobytes (32 pages)

Compilation Speed (CPU time only)

2500-4000 lines per minute.

Optimizations by the Compiler

Constant expression folding: Compile-time evaluation of expressions that involve only constant values, named constants, and arithmetic, logical and relational operators. Some folding of character operations is also performed.

Subscript evaluation: Parts of subscripts that can be evaluated at compile time are removed from the generated code; subscript calculations are done with in-line code.

Dead code removal: Unreachable code within IFTHEN-ELSE, DO-WHILE, and DO constructs is removed (i.e. no code is generated).

Logical and arithmetic IF: Branch structure optimizations decrease program size and improve execution speed. For example, the compiler detects if statement numbers in an arithmetic IF statement are not distinct.

DO statements: The compiler takes advantage of constant initial, final and step values.

Data Types

Data Type	Precision (digits)	Size (bits)	Range
INTEGER*2	5-6	16	−32768 to 32767
INTEGER*4	9-10	32	−2,147,483,648 to 2,147,483,647
LOGICAL*2	—	16	.true,.false
LOGICAL*4	—	32	.true,.false
REAL*4	6.6-6.9	32	1.47×10^{-39} to $1.70 \times 10^{+38}$
REAL*8	16.3-16.6	64	
COMPLEX*8	6.6-6.9	64	
COMPLEX*16	16.3-16.6	128	
CHARACTER	—	1-32767 chars.	256 char., full ASCII
HOLLERITH	—	2 to 8 chars.	254 char., full ASCII

Note: For COMPLEX data, table applies to real and imaginary parts separately.

FORTRAN Library Functions

Trigometric		Conv.	Max/Min Mod etc.	Bit Manip.
*SQRT	DSINH	IFIX	*MIN	IAND
DSQRT	*COSH	FLOAT	MIN0	IOR
CSQRT	DCOSH	SNGL	MIN1	NOT
*SIN	*TANH	*DBLE	AMIN0	IXOR
DSIN	DTANH	*REAL	AMIN1	IEOR
CSIN	*ASIN	*INT	DMIN1	ISHFT
*COS	DASIN	AINT	*MAX	ISHFTC
DCOS	*ACOSN	DINT	MAX0	IBITS
CCOS	DACOS	DDINT	MAX1	BTEST
*TAN	*ATAN	IDINT	AMAX0	IBSET
DTAN	DATAN	*NINT	AMAX1	IBCLR
CTAN	*ATAN2	IDNINT	DMAX1	MVBITS
*EXP	DATAN2	ANINT	*MOD	
DEXP	*ASINH	DNINT	AMOD	
CEXP	DASINH	*IMAG	DMOD	**Char.**
*LOG10	*ACOSH	AIMAG	*SIGN	
ALOG10	DACOSH	CMPLX	ISIGN	CHAR
DLOG10	*ATANH	DCMPLX	DSIGN	ICHAR
*LOG	DATANH	CONJG	*DIM	INDEX
ALOG	*ABS		IDIM	LEN
DLOG	IABS	**Misc.**	DDIM	LLT
CLOG	DABS			LLE
*SINH	CABS	ISSW		LGE
		DPROD		
		PCOUNT		

* *Identifies generic form.*

Compiler Options

L: Produces listing

M: Produces mixed listing

T: Produces table of symbols (type, address, etc.)

C: Produces cross-reference of symbols

D: Compiles debug lines

Q: Adds relative addresses of statements to listing

E: EMA transparency; causes all subroutine parameters to use 32-bit addresses.

X or Y: Selects default double precision size to be 48 or 64 bits.

I or J: Selects default integer size to be 16 or 32 bits.

Ordering information

92836A FORTRAN 77

92836A FORTRAN 77, which must be ordered with Use Option 600, 601, 700, 701, 703, 704, 890, or 891, includes:

1. FORTRAN 77 software on Media Option 020, 022, 041, 042, 044, 050, or 051, **one of which must be ordered.**
2. FORTRAN 77 Reference Manual (92836-90001).

92836A Media Options

020: Provides 92836A software on 264x Mini cartridges.

022: Provides 92836A software on CS/80 cartridge tape.

041: Provides 92836A software on 1.2M byte flexible disc.

042: Provides 92836A software on 5-in. Minifloppy discs.

044: Provides 92836A software on 3.5-in. Microfloppy discs.

050: Provides 92836A software on 800 bpi magnetic tape.

051: Provides 92836A software on 1600 bpi magnetic tape.

92836A/92836R Use Options

600: Use in A600 system.

601: Upgrade from previous version of 92836A/R option 600 to latest version of same for user not on software support service.

700: Use in A700 or E/F-Series system.

701: Upgrade from previous version of 92836A/R option 700 to latest version of same for user not on software support service.

703: Upgrade from 92834A/R to 92836A/R for use in E/F-Series system for user on software support service.

704: Upgrade from 92834A/R to 92836A/R for use in E/F-Series system for user not on software support service.

890: General license to use in A900 or any other A/E/F-Series System, including right to purchase 92836R Opt 600/700/890 right to copy products for additional systems.

891: Upgrade from previous version of 92836__ Opt 890 or 700 to latest version of 92836__ Opt 890 for customer NOT on support service.

92836R Right to Copy FORTRAN 77 Compiler For Use on an Additional Computer System

The 92836R Right to Copy product, which must be ordered with Use Option 600, 601, 700, 701, 703, 704, 890, or 891, is available only to customers who have purchased a license to use 92836A without an upgrade option. 92836R consists of:

1. The license to make one copy of software purchased with 92836A for use on an additional system.
2. FORTRAN 77 Reference Manual (92836-90001).

Software support products available

See page 1-4 of Volume I of the HP 1000 Software Data book.

RTE FORTRAN 4X

HEWLETT PACKARD

For Program Development
in RTE-6/VM, RTE-IVB, or RTE-XL

product number 92834A

The FORTRAN 4X compiler is an extensively-enhanced version of the RTE FORTRAN IV compiler that runs in the RTE-IVB and RTE-XL operating systems. FORTRAN 4X provides 32-bit integer, file I/O, IF-THEN-ELSE capabilities, improved code generation, and other improvements which offer increased computational power and easier programming to the FORTRAN user.

Features

- **Fast compilation**
- **Local optimization**
- **Direct access I/O (both IBM and FORTRAN 77 Style)**
- **Many features of the ANSI 77 Standard, such as IF-THEN-ELSE**
- **16- and 32-bit integer; 16- and 32-bit logical; single, extended, and double precision real; and single and double complex data types**
- **Full I/O capability to/from remote nodes of a distributed system**
- **Compatibility with HP 1000 software subsystems, such as Image/1000, Graphics/1000-II, DATACAP/1000-II, and DS/1000-IV**

Extensions to standard ANSI FORTRAN 66

HP FORTRAN 4X is a superset of ANSI FORTRAN X3.9-1966, with the following enhancements:

Declarations

Byte length type declarations: All type declarations can be followed by *n where n is an integer constant; the following declarations are possible:

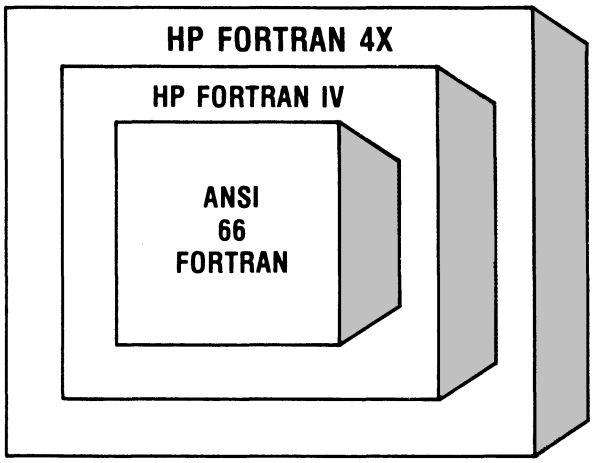
INTEGER*2	integer
INTEGER*4	double integer
LOGICAL*2	logical
LOGICAL*4	double logical
REAL*4	real
REAL*6 or DOUBLE PRECISION*6	extended precision
REAL*8 or DOUBLE PRECISION*8	double precision
COMPLEX*8	complex
COMPLEX*16	double complex

Double integer constants and variables are allowed wherever a single integer is legal.

IMPLICIT and IMPLICIT NONE declarations: The IMPLICIT declaration redefines the implied data type of symbolic names. The IMPLICIT NONE declaration requires that all variables be declared (typed).

Array dimensions: Arrays can have up to seven dimensions.

Array declarators: Lower and upper bounds of array dimensions can be specified in array declarators. The value of the lower bound dimension declarator can be negative, zero, or positive.



Alphanumeric literals in DATA statement: Strings of characters bounded by apostrophes can be used in place of Hollerith constants in DATA statements.

Long Hollerith in DATA statements: A single Hollerith constant may initialize more than one element of an array.

Executable Statements

General expressions are permitted for the initial value, increment, and limit parameters in the DO statement; as control parameter in the computed GOTO statement; in I/O lists of WRITE and PRINT statements; as array subscripts; as logical unit numbers.

IF-THEN-ELSE construct: The IF(...) THEN, ELSE, ELSEIF(...) THEN, and ENDIF statements are included in HP FORTRAN 4X.

Alternate returns mechanism: Allows program flow to return to a statement other than the one following the subroutine call statement.

DO increment parameter: Value of the DO statement increment parameter can be either positive or negative. (DO indices must be single or double word integers).

Logical operations on integer data: The logical operators .AND., .OR., .NOT., .XOR., .EQV., and .NEQV. may be applied to 16-bit or 32-bit integer data to perform bit masking and manipulation.

Additional instrinsic functions include bit shifting, ISHFT, number of actual parameters, PCOUNT, and a number of transcendental functions such as arcsine.

Generic function selection: A single name may be used for all versions of an intrinsic (e.g. SIN vs DSIN) for most FORTRAN library functions.

Direct access READ and WRITE statements are compatible with ANSI FORTRAN 77 and IBM style and may be formatted or unformatted.

ENCODE and DECODE statements provide memory to memory formatting.

List directed READ and WRITE are allowed.

ANSI FORTRAN 77 style I/O statements including OPEN and INQUIRE.

Error and End of File Control are provided by the END=, ERR=, and IOSTAT keywords.

HP-IB device control is provided by passing secondary addressing or control buffers within READ and WRITE statements.

Program Form

Lower case: Lower case keywords and names are accepted and mapped into upper case, thereby improving readability.

Compiler command: A compiler command line specification allows all integer and logical declarations without explicit length specifications to be considered as INTEGER*2 and LOGICAL*2, or INTEGER*4 and LOGICAL*4 respectively.

Debugging statements: Statements included in a program to aid debugging can be designated by the letter "D" in column 1 and compiled only when the associated command string option switch is set. Otherwise they are treated as comments.

Comment lines: An asterisk is accepted as the start of a comment line. Also, any FORTRAN statement can be followed in the same line by a comment that begins with an exclamation point.

Error directory provides brief explanation of each error detected.

Parameter (p1 = k1, p2 = k2, ... kn) defines names for constants, which may then be used in DIMENSION statements and anywhere else a constant may be used, including constant folding and conditional compilation.

Conditional compilation provides for different compilations of one source file to accommodate different requirements on different systems or for different applications.

Functional specifications

Applicable Standard

HP FORTRAN 4X is a superset of ANSI FORTRAN X3.9-1966.

Environment

Operating system: HP 1000 Computer System operating under RTE-6/VM, RTE-IVB, or RTE-XL, plus a list device.

Memory Requirements:

Program size	Memory required
Up to 2000 lines	38 kilobytes
Up to 5000 lines	56 kilobytes

Compilation Speed (cpu time only)

In M/E/F-Series operating under RTE-6/VM or RTE-IVB: 2000-3000 lines/minute.

In L-Series operating under RTE-XL: 500-1000 lines/minute.

Local Optimizations by Compiler

Constant expression folding: Folding of operations that involve only constant values is done for all arithmetic and relational operators which do not also involve exponentiation or complex data.

Subscript evaluation: Parts of subscripts that can be evaluated at compile time are removed from the generated code and subscript calculations are done with in-line code.

Logical and arithmetic IF: Branch structure optimizations decrease program size and improve execution speed. For example, the compiler detects if statement numbers in an arithmetic IF statement are not distinct.

Computed GOTO: In-line code is generated for three or fewer statement numbers.

DO statements: The compiler takes advantage of constant final values.

Character Set

Alphabetic characters: All upper and lower case letters (A through Z and a through z in which a = A, b = B, etc.).

Numeric characters: The ten digits 0 through 9.

Special characters: Blank; equals, plus, and minus signs; asterisk; slash; left and right parentheses; comma; decimal and exclamation points; currency symbol, quotation marks, and apostrophe.

Hollerith data: All characters are legal within Hollerith data, except null and carriage return.

Data Types

Integer: A 16-bit quantity, including sign, that ranges from −32768 to +32767.

Double integer: A 32-bit quantity, including sign, that ranges from −2,147,483,648 to +2,147,483,647.

Real: A 32-bit quantity with sign, exponent, and mantissa that ranges from $\pm 2^{-129}$ to $\pm 2^{+127}$ (about 1.47×10^{-39} to 1.70×10^{38}), providing 6.6 to 6.9 decimal digit accuracy.

Extended precision: A 48-bit quantity with sign, exponent, and mantissa with same range as Real, that provides 11.4 to 11.7 decimal digit accuracy.

Double precision: A 64-bit quantity with sign, exponent, and mantissa with same range as Real, that provides 16.3 to 16.6 decimal digit accuracy.

Complex: A 64-bit quantity consisting of two real data quantities, one for the real part of a complex quantity, the other for the imaginary part.

Logical: A 16-bit variable in which only the sign bit is used to determine the Boolean value, true or false.

Double logical: A 32-bit variable in which only the sign bit is used to determine the Boolean value, true or false. This data type is made available to help transport programs which EQUIVALENCE logical variables or which store Hollerith data in logical variables.

Hollerith: ASCII data which may be used as numeric values or actual parameters to subprograms.

Program Vocabulary Extensions With Respect to ANSI FORTRAN X3.9-1966

Specification statements:

$EMA (CMDAT,3) In RTE-IVB only, sets up Extended Memory Area (EMA) common to provide for quick referencing and manipulation of large data arrays.

$INCLUDE (file name) (directive or statement) Compiles the statements in the named file as if they were in the source at this point.

$TITLE Specifies a new title for the program listing.

$PAGE Causes a page eject in the program listing.

$LIST ON or $LIST OFF Controls the listing on a line-by-line basis. Also controls the mixed listing, if any.

PROGRAM CALST(4,90) Identifies program by name and sets program specification, such as priority, for the system.

IMPLICIT REAL*8 (A-T) Redefines implied data type of symbolic names; in this example, variable names with first letter A through T are implicitly defined as being double precision (REAL*8) variables.

IMPLICIT NONE Disables implicit data type designation, so the types of all variables must be declared.

Control statements:

```
IF (SALARY .LT. 1) THEN
   TAX = 0
ELSE IF (SALARY .LT. 10) THEN
   TAX = SALARY/2
ELSE
   TAX = SALARY
ENDIF
```

Executes one of two or more blocks of statements depending upon the value of a logical expression (may be nested to any level).

CALL SUBR (I,J,A,B,*10,*30,*25) Calls a subroutine and passes it actual arguments to replace the dummy arguments in the subroutine. Actual arguments may include statement numbers used as alternate returns.

RETURN (J+2) Returns control from a subprogram to the calling program unit. This statement may select an alternate return from those supplied in a CALL statement.

Input/output statements:

OPEN (6,FILE='OUTPUT',STATUS='NEW',NODE=2001) Associates an existing file with a logical unit, or creates a new file and associates it with a logical unit. The file to be opened can be either local or can be located at a remote node of a DS/1000 Distributed Systems network. In addition, the OPEN statement can contain specifications for file creation or subsequent processing, including file name, direct or sequential access, form (formatted or unformatted), record size, interpretation of blanks, and whether the file can be shared.

CLOSE (6, ERR=99) Disassociates a file from a logical unit. A status keyword provides for saving or deleting the file.

READ (U,F) R or **READ (U,*,END=22) R** Reads one logical record R from logical unit U, in format designated by statement F, or as free-field input designated by *, and assigns the input values to the elements in a list (formatted READ).

READ (U,REC=RECNUM) A,B,C or **READ (U'R) A,B,C** Reads specified logical record from logical unit U; may be either formatted or unformatted (direct access READ).

WRITE (U,ZBUF=CNBUFF,ZLEN=100) BUFFER Writes one logical record BUFFER to device U on the HP-IB bus, using control buffer CNBUFF of length 100 words.

PRINT 10, I, X, Y List-directed write to the default output device (normally a printer).

INQUIRE (U, ACCESS=ATYPE, NEXTREC=RECNUM) Sets specified variables to values of file attributes, including all information supplied in the OPEN statement, plus file position, whether it is open, and whether it exists if not open.

ENCODE (9,'("(T",I3.3,",I5)")',PFORM) COLUMN Writes the elements of the I/O list into a memory buffer, translating the data into ASCII format.

DECODE (5,100,ASCII) (CHARS(I),I=1,5) Reads the elements in the I/O list from a memory buffer, translating the data from ASCII format into internal binary format.

Logical operators (operate on logical or integer constants, variables, and expressions. NOTE: .AND., .OR., and .NOT. have standard meaning):

A .EQV. B Logical EQUIVALENT (true if A and B are both true or if A and B are both false)

A .NEQV. B Logical NOT EQUIVALENT (true if A is true and B is false, or vice versa)

A .EOR. B or **A .XOR. B** Logical EXCLUSIVE OR (true if A is true and B is false, or vice versa)

FORTRAN Library Functions

IFIX	AMIN0	*COS	*ACOS
FLOAT	AMIN1	DCOS	DACOS
SNGL	DMIN1	CCOS	*ATAN
*DBLE	*MAX	*TAN	DATAN
*REAL	MAX0	DTAN	*ATAN2
*INT	MAX1	CTAN	DATAN2
AINT	AMAX0	*EXP	*ASINH
DINT	AMAX1	DEXP	DASINH
DDINT	DMAX1	CEXP	*ACOSH
IDINT	*MOD	*LOG10	DACOSH
*NINT	AMOD	ALOG10	*ATANH
IDNINT	DMOD	DLOG10	DATANH
ANINT	*SIGN	*LOG	IAND
DNINT	ISIGN	ALOG	IOR
AIMAG	DSIGN	DLOG	NOT
CMPLX	*DIM	CLOG	IXOR
CONJG	IDIM	*SINH	IEOR
*ABS	DDIM	DSINH	ISHFT
IABS	*SQRT	*COSH	PCOUNT
DABS	DSQRT	DCOSH	ISSW
CABS	CSQRT	*TANH	
*MIN	*SIN	DTANH	
MIN0	DSIN	*ASIN	
MIN1	CSIN		

* *Identifies generic form.*

Compiler Options

L: Produces Listing.

M: Produces Mixed listing.

A: Produces Assembly listing.

T: Produces table of symbols (type, address, etc.)

C: Produces cross-reference of symbols.

D: Compiles debug lines.

F: Formats listing for Teleprinter.

Q: Adds relative addresses of statements to listing.

X and Y: The X option declares all double precision variables and constants to default to 48 bits; variables may be explicitly typed to 64-bit double precision, but constants may not. The Y option declares all double precision variables and constants to default to 64 bits; variables may be explicitly typed to 48-bit extended precision, but constants may not.

I and J: The I option declares that all integer variables and constants default to 16 bits; either may be explicitly typed to 32 bits. The I option declares that all integer variables and constants default to 32 bits; either may be explicitly typed to 16 bits.

Ordering information

92834A RTE FORTRAN 4X

92834A FORTRAN 4X, which must be ordered with Use Option 300, 301, 700, or 701, includes:

1. FORTRAN 4X compiler and library on software Media Option 020, 022, 041, 042, 050, or 051, **one of which must be ordered.**
2. RTE FORTRAN 4X Reference Manual (92834-90001).

92834A Media Options

020: Provides 92834A software on 264x Mini cartridges.

022: Provides 92834A software on CS/80 cartridge tape.

041: Provides 92834A software on 1.2M byte flexible discs.

042: Provides 92834A software on 5-inch Minifloppy discs.

050: Provides 92834A software on 800 bpi magnetic tape.

051: Provides 92834A software on 1600 bpi magnetic tape.

92834A/92834R Use Options

300: Use in L-Series system.

301: Upgrade from previous version of 92834A/R option 300 to latest version of same for user not on software support service.

700: Use in E/F-Series system.

701: Upgrade from previous version of 92834A/R option 700 to latest version of same for user not on software support service.

92834A Right to Copy FORTRAN 4X Compiler For Use on an Additional Computer System

The 92834R Right to copy product is available only to customers who have purchased a license to use 92834A without an upgrade discount option. 92834R, which must be ordered with Use Option 300, 301, 700, or 701, consists of:

1. The license to make one copy of software purchased with 92834A for use on an additional system.
2. RTE FORTRAN 4X Reference Manual (92834-90001).

Software support products available

See page 1-4 of Volume I of the HP 1000 Software Data book.

Pascal/1000

HEWLETT PACKARD

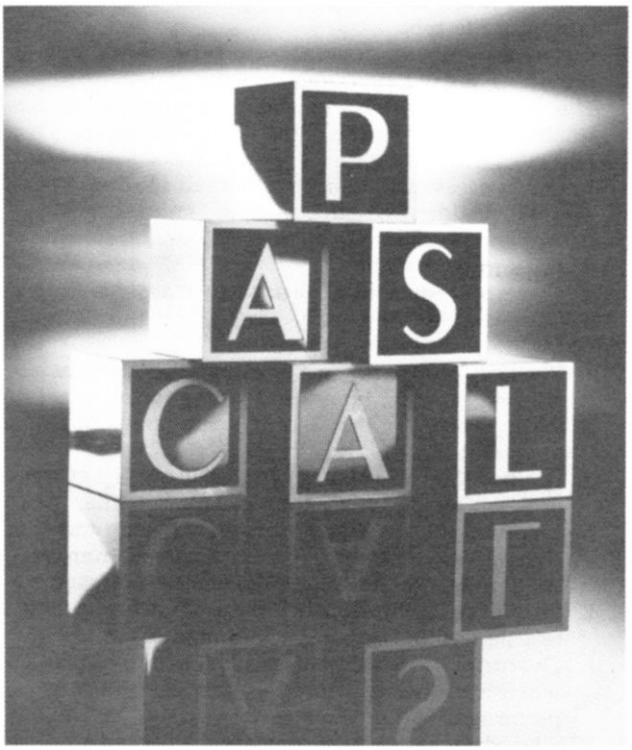
For program development in RTE-6/VM, RTE-A, RTE-IVB, or RTE-XL

product numbers 92832A, 92833A and 92854A

HP Pascal/1000 compilers implement the Pascal high-level, block-structured programming language. They can be used for program development in HP 1000 Computer Systems operating under Hewlett-Packard's RTE-6/VM, RTE-A, or RTE-A/VC+ (92833A compiler), RTE-IVB (92832A* compiler), or RTE-XL (92854A* compiler) real-time executive operating systems. Pascal/1000 is the implementation of HP Pascal for the HP 1000 computer family. HP Pascal is a superset of ANSI Pascal, and is supported on most HP computer systems. Pascal/1000 also provides some important extensions that take full advantage of the capabilities provided by the HP 1000. Pascal/1000 object programs can be executed in HP 1000 Computer Systems operating under HP's RTE-6/VM, RTE-A, RTE-A/VC+, RTE-IVB, RTE-IVE, RTE-IVE, RTE-XL, and RTE-L systems.

HP 92833A Pascal/1000 supports the Command Interpreter and hierarchical directories in RTE-A, RTE-A/VC+, and RTE-6/VM. The HP 92833A compiler also supports Code and Data Separation (CDS) capability for large programs in the RTE-A/VC+ system in addition to support of non-CDS program development. Under RTE-A/VC+, the Pascal compiler itself is a CDS program, allowing multiple compilations to share one copy of the compiler's code.

**The 92832A and 92854A Pascal/1000 compilers are Mature software products.*



Features

- **Easily-understood programming**
- **Powerful, compact syntax**
- **Modern block-structured language**
- **Logical organization that facilitates documentation, modification, and maintenance of programs**
- **Early detection of errors at compile and run time**
- **Fast debugging with 92860A Symbolic Debug/1000 (92833A Pascal/1000 only)**
- **Improved program reliability**
- **Separate compilation of modules with interface checking at compile time (92833A Pascal/1000 only)**
- **Clearly defined data structures complemented by flexible user-declared data types**
- **Generation of CDS output code for RTE-A/VC+ systems (92833A Pascal/1000 only)**
- **Access via Command Interpreter file system as well as FMGR file system (92833A Pascal/1000 only)**
- **Files in the Heap (92833A Pascal/1000 only)**
- **FORTRAN and Macro/1000 Assembly language subroutines**
- **Fast execution**
- **String data type (92833A Pascal/1000 only)**
- **Hexadecimal, octal, and binary data support (92833A Pascal/1000 only)**
- **16- and 32-bit integer, single and double precision real, Boolean, character, text file, and user-defined data types using arrays, records, sets, files, pointers, and variable-length strings**
- **Shared compiler in RTE-A/VC+ systems (92833A Pascal/1000 only)**
- **Compatibility with HP 1000 software subsystems, such as Image/1000, Graphics/1000-II, and DS/1000-IV.**

Pascal standards

Pascal/1000 is an extension of HP Pascal, which in turn is an extension of ANSI Pascal.

HP Pascal Extensions of ANSI Pascal

1. An OTHERWISE clause may be specified in the CASE statement.
2. Compiler options (ANSI, PARTIAL_, EVAL, LIST, PAGE, and INCLUDE) may be specified to control various aspects of the compilation and its output.
3. A constant expression may appear in an HP Pascal program anywhere that a constant may appear in ANSI Pascal.
4. Constants of structured data types may be declared in the CONST section of a block.
5. CONST, TYPE, and VAR sections of a declaration may be intermixed and repeated.
6. Halt procedure may be used for abnormal termination of a program.

7. The underscore (_) may appear in identifiers, but not as the first character.
8. A function may return a structured type (array, record, set, or string), but not a file type or a structure containing a file type.
9. Longreal numbers identical with type REAL, but providing greater precision, are supported.
10. The standard constant, minint, is defined to be the smallest integer representable on the machine.
11. File I/O includes:
    a. Direct access I/O, using predefined routines open, seek, readdir, writedir, maxpos, and last pos.
    b. Append procedure to open a file to write, starting at the end of the file.
    c. Close procedure to explicitly close any file.
    d. "Deferred get" procedure to support interactive input.
    e. Read procedure that accepts variables of enumerated types, PAC types, and string types.
    f. Write procedure that accepts expressions of enumerated and string types.
    g. Position function that returns index of the current position for any file which is not a textfile.
    h. Linepos function that returns the integer number of characters which the program has read from or written to a textfile since the last line marker.
    i. Prompt procedure flushes the output buffer of a textfile without writing a line marker.
    j. Overprint procedure causes a line to be overprinted when a textfile is printed. A carriage return is performed without line feed.
12. STRING data type, which consists of a packed array of CHAR with a declared maximum length and an actual length that may vary at run time, is supported. Several operators, procedures, and functions manipulate strings. Assignment (:=) operator may be used to assign strings or string literals to strings. Concatenation (+) produces a string composed of two other strings.
13. Record variant declaration in which the variant part of a record field list may have subranges of constant expressions as case constants.
14. String literals support permits the encoding of control characters or any other single ASCII character after the # symbol. The # character may be used to include any ASCII character within a string literal.

Pascal/1000 Extensions to HP Pascal

1. REAL and LONGREAL constant expressions are supported.
2. Subprograms or segments may be separately compiled.
3. Additional compiler options are supported (see table below).
4. Some user-callable library routines are provided in the Pascal runtime libraries.

Functional specifications

Environment

Disc Support for Program Development: A hard disc with at least 15 megabytes capacity is required to provide a reasonable environment for program development.

Program Development on RTE-A/VC+ system: The CDS version of the 92833A Pascal/1000 compiler runs in a shared code partition of 360 pages (720 kb) and a VMA data partition of 50–130 pages (100–260 kb) for each user. A separate 32 page (64 kb) partition is required for running the Pascal Monitor. Compilation speed approximates 1200 lines per minute. The speed experienced by each of several concurrently active users depends upon the number of users. For example, each of six users would realize an effective compilation speed of about 200 lines per minute.

Program Development on RTE-A or RTE-6/VM system: The non-CDS version of the 92833A Pascal/1000 compiler runs in a Virtual Memory Area (VMA) partition of at least 40 pages (80 kb). Compilation speed ranges from 250 to 1200 lines per minute, depending upon the size of the working set. Maximum compilation speeds are obtained with a working set size of at least 400 pages (800 kb). A separate 32 page (64 kb) partition is required for running the Pascal Monitor.

Program Development on RTE-IVB system: The 92832A Pascal/1000 compiler runs in a mother partition using the Extended Memory Area (EMA) managed by the RTE-IVB operating system. The EMA partition size must be at least 42 pages (84k bytes) resulting in a compilation speed of approximately 50 lines per minute. Additional pages increase compilation speed; exact memory requirements depend on the customer's needs. To achieve approximately 400 compiled lines per minute, a 220 page (440 kb) EMA partition is needed. A separate 17 page (34 kb) partition is required for running the Pascal Monitor.

Program Development on RTE-XL system: The 92854A Pascal/1000 compiler requires a system with at least 128 kb memory. One 14-page (28 kb) partition is required to run Pascal, yielding a compile speed of approximately 30 lines per minute. Compile speeds up to 150 lines per minute can be achieved with more partitions on a 512 kb system. This compiler can compile programs of up to 500 lines.

Pascal Program Execution

CDS relocatables: Run under RTE-A/VC+ only.

Non-CDS relocatables: Run under RTE-A/VC+, RTE-A, or RTE-6/VM and, after processing by the 92832A/92854A ALTER untility, under RTE-IVB/E and RTE-L.

Non-CDS relocatables that access only FMGR files: Run under RTE-XL.

Character Set

Alphabetic characters: All upper and lower case characters (A through Z and a through z).

Numeric characters: The ten digits 0 through 9.

Special characters: blank; currency symbol ($); apostrophe ('); left and right parentheses; comma (,); plus, minus, equals, less than, and greater than symbols (+, −, =, <, >); decimal point (.); slash; colon and semi-colon; left and right brackets; left and right braces; carat ( ); @ symbol; # sign; asterisk (*); and underscore (_).

Data Types

Integer: A 32-bit quantity, including sign, that ranges from −2,147,483,648 to +2,147,483,647.

Real: A 32-bit quantity with sign, exponent, and mantissa that ranges from $\pm 2^{-128}$ to $\pm 2^{+127}$, providing 6 to 7 decimal digit accuracy.

Longreal: A 64-bit quantity with sign, exponent, and mantissa that ranges from $\pm 2^{-128}$ to $\pm 2^{+127}$, providing 16 to 17 decimal digit accuracy.

Boolean: A 16-bit variable in which only the low order bit is used to determine the Boolean value true (1) or false (0).

Char: Values are the set of characters defined by the 8-bit ASCII character set.

Subrange type: A data type can be identified as a subrange of another ordinal type (Integer, Boolean, Char, or enumeration type) in which the least and largest values of the subrange are identified.

Array type: A structure consisting of a fixed number of components which are all of the same type, called the component type in which the elements of the array are designated by indices. The array type definition specifies the component type and the index type. Component type may be any type, including another structured type.

String type: An ASCII character series of variable length (up to 32767 characters) represented by a variable.

Record type: A structure consisting of a fixed number of components that can be of different types. For each component, called a field, the record definition specifies its type and an identifier.

Set type: Defines a range of values which is the powerset of a base type, which can be Integer, Boolean, Char, or subrange or any enumeration type.

File type: Defines a structure consisting of a sequence of components that are all of the same type. The number of components (length) of the file is not fixed by the file definition.

Compiler Options

See Table 1.

Table 1. Pascal/1000 Compiler Options

APPLIES TO 928___			COMPILER OPTION
33A	32A	54A	
Y	Y	Y	ALIAS: Specifies externally-accessible name of a Pascal program or module.
Y	Y	Y	ANSI: Issues warnings for non-ANSI constructs.
Y	Y	Y	ASMB: Specifies options to the assembler.
Y	Y	Y	AUTOPAGE: Automatically pages before each routine in the listing.
Y	N	N	BASIC_STRING: Converts Pascal string parameters to BASIC strings.
Y	Y	Y	BUFFERS: Defines number of file buffers.
Y	N	N	CDS: Requests generation of CDS instr.
Y	Y	Y	CODE: Enables code generation.
Y	N	N	CODE_CONSTANTS: Places structured constants in code space.
Y	N	N	CODE_INFO: Requests printing of generated code size information.
Y	N	N	CODE_OFFSETS: Prints the code offset of each Pascal line in the listing.
Y	N	N	DEBUG: Generates information needed by 92860A Symbolic Debug/1000.
Y	Y	Y	DIRECT: Use of faster calling sequence for a given procedure.
Y	Y	N	EMA: Specifies EMA and MSEG sizes.
Y	N	N	EMA_VAR: Allocates selected global variable in EMA/VMA.
Y	Y	Y	ERROREXIT: Specifies error return on an external routine.
Y	N	N	FAST_REAL_OUT: Use faster, less precise output routines for real numbers.
Y	N	N	FIXED_STRING: Converts Pascal string parameters to FTN7x strings.
Y	Y	Y	HEAP: Chooses small or large heap model for the program.
Y	N	N	HEAP_DISPOSE: Chooses heap management algorithm.
Y	Y	Y	HEAPPARMS: Specifies one or two word VAR parameter addresses.
Y	Y	Y	IDSIZE: Specifies the no. of significant characters in identifiers (1 to 150).
Y	Y	Y	IMAGE: Reserves buffer space for Image programs.
Y	Y	Y	INCLUDE: Requests inclusion of source from another file for compilation.
Y	N	N	INCLUDE_DEPTH: Specifies maximum depth of include file nesting.
Y	Y	Y	KEEPASMB: Saves the generated assembly file after compilation.
N	N	Y	LIBRARY: Identifies current compilation as searchable library of routines.
Y	Y	Y	LINES: Specifies the number of lines per page in the listing.
Y	Y	Y	LINESIZE: Specifies the maximum number of characters in a textfile line.
Y	Y	Y	LIST: Generates compiler listing.

Table 1. Pascal/1000 Compiler Options, continued

APPLIES TO 928___			COMPILER OPTION
33A	32A	54A	
Y	Y	Y	LIST_CODE: Prints emitted instructions in the compiler listing.
Y	Y	Y	MIX: Emits Pascal source as comments in the generated Assembly file.
Y	N	N	NOABORT: Specifies no abort error return on an external routine.
Y	Y	Y	PAGE: Page eject the compiler listing.
Y	Y	Y	PARTIAL_EVAL: Requests partial evaluation of Boolean operators.
Y	Y	Y	PASCAL: Specifies header information in the relocatable file.
Y	N	N	PRIVATE_TYPES: Defines end of common globals for 92860A Symbolic Debug/1000.
Y	Y	Y	RANGE: Enables range checking of variables, subscripts, and pointers.
Y	Y	Y	RECURSIVE: Enables recursive invocation of a routine.
Y	Y	Y	RESULTS: Specifies a file to contain compilation errors and results.
Y	N	Y	RUN_STRING n: Reserves n characters to be passed in the run-string to the program.
Y	N	N	SEARCH: Specifies list of files to search for imported modules.
Y	Y	Y	SEGMENT: Defines current compilation unit as a program segment.
Y	N	N	SEGMENTED: Identifies a program which uses modules and has overlays.
Y	N	N	SKIP_TEXT: Causes skipping of source code until $SKIP_TEXT OFF$ is found.
Y	N	N	SMALL_TEMPS: Specifies number of words to reserve for "small" temporaries in CDS programs.
Y	N	N	STANDARD_LEVEL: Chooses standard level to which compilation unit adheres.
Y	Y	Y	STATS: Prints status of compiler options and configuration information.
Y	Y	Y	SUBPROGRAM: Defines the current compilation unit as a subprogram.
Y	Y	Y	SUBTITLE: Prints a subtitle under the title in the listing.
Y	Y	Y	TABLES: Prints symbol table information.
Y	Y	Y	TRACE: Prints a procedure-call history of the executing program.
Y	Y	Y	TITLE: Prints a title at the top of each page in the listing.
Y	N	N	TRACE_BACK: Displays procedure call chain at the point of run-time error.
N	N	Y	VISIBLE: Makes level 1 routines externally accessible.
Y	N	N	WARN: Prints warnings in the listing.
Y	Y	Y	WIDTH: Specifies number of significant characters in source lines.
Y	Y	N	WORK: Specifies compiler's memory-resident workspace size.
N	Y	Y	XREF: Generates Pascal cross-reference.

Ordering information

92833A Pascal/1000 For RTE-A, RTE-A/VC+, and RTE-6/VM

92833A Pascal/1000, which must be ordered with Use Option 600, 601, 700, 701, 702, 890, or 891, includes:

1. Pascal/1000 compiler, library, error message file, and cross reference generator on software Media Option 020, 022, 041, 042, 044, 050, or 051, **one of which must be ordered.**
2. Pascal programmer's reference manual (92833-90005).
3. Programming in Pascal (tutorial manual) (97082-90002).
4. Pascal/1000 Software Numbering Catalog (92833-90006).

92833A Media Options

020: Software on 264x minicartridges.

022: Software on CS/80 cartridge tape.

041: Software on 1.2M byte flexible disc.

042: Software on 5-in. Minifloppy discs.

044: Software on 3.5 in. Microfloppy discs.

050: Software on 800 bpi magnetic tape.

051: Software on 1600 bpi magnetic tape.

92833A/92833R Use Options

600: Use in A600 or A600+ system.

601: Upgrade from previous version of 92833A/R option 600 to latest version of same for user not on software support service.

700: Use in A700 or E/F-Series system.

701: Upgrade from previous version of 92833A/R option 700 to latest version of same for user not on software support service.

702: Upgrade from 92832A/R to 92833A/R for use in E/F-Series system for user not on software support service.

890: General license to use in A900 or any other A/E/F-Series System, including right to purchase 92833R Opt 600/700/890 right to copy products for additional systems.

891: Upgrade from previous version of 92833A/R option 890 or 700 to latest version of 92833A/R option 890 for customer NOT on software support service.

92832A Pascal/1000 For RTE-IVB (Mature product)

92832A Pascal/1000 includes:

1. Pascal/1000 compiler, library, error message file, and cross reference generator on software Media Option 020, 050, or 051, **one of which must be ordered.**
2. Pascal programmer's reference manual (92832-90005).
3. Programming in Pascal (tutorial manual) (92832-90001).
4. Pascal/1000 Configuration Guide (92832-90006).
5. Pascal/1000 Software Numbering Catalog (92832-90004).

92832A Options

001: Provides discount to a user upgrading from previous version of 92832A to latest version if not on software support service.

020: Software on 264x mini cartridges.

050: Software on 800 bpi magnetic tape.

051: Software on 1600 bpi magnetic tape.

92854A Pascal/1000 For RTE-XL (Mature product)

92854A Pascal/1000 includes:

1. Pascal/1000 compiler, library, error message file, and cross reference generator on one of software Media Options 022 through 051, **which must be ordered.**
2. Pascal programmer's reference manual (92832-90001).
3. Programming in Pascal (tutorial manual) (92832-90002).
4. Pascal/1000 Configuration Guide (92854-90003).
5. Pascal/1000 Software Numbering Catalog (92854-90004).

92854A Options

001: Provides discount to a user upgrading from previous version of 92854A to latest version if not on software support service.

022: Software on CS/80 cartridge tape.

041: Software on 1.2M byte flexible disc.

042: Software on 5-inch Minifloppy discs.

051: Software on 1600 bpi magnetic tape.

92832R/92833R/92854R Right to Copy Pascal/1000 Compiler For Use on an Additional Computer System

The 92832R/92833R/92854R Right to copy product is available only to customers who have purchased a license to use 92832A/92833A/92854A without discount option. 92832R/92833R/92854R consists of (92833R must be ordered with Use Option 600, 601, 700, 701, 702, 890, or 891):

1. The license to make one copy of software purchased with 92832A/92833A/92854A for use on an additional system.
2. All manuals furnished with 92832A/92833A/92854A.

Support service products available

See page 1-4 of Volume I of the HP 1000 Software Data book.

BASIC/1000C

HEWLETT PACKARD

For program development in RTE-A and RTE-6/VM operating systems

product number 92857A

BASIC/1000C is a BASIC language subsystem for multi-user conversational program development, testing, debugging, execution, and compilation in hard disc based HP 1000 Systems operating under RTE-A or RTE-6/VM.

Features

- **Interpreter with editor and multifunction debugger for fast, friendly, conversational program development**
- **Compiler for very fast run-time performance and source security**
- **Compatibility with approximately 100 statements of HP 9826/9845 Desktop Computer BASIC**
- **HP-IB statements**
- **Multi-user system with multi-user interrupt handling**
- **User-controlled error handling statements**
- **Labelled common**
- **15-character variable names and line labels**
- **Integer, double integer, and two-word and four-word floating point data types**
- **Matrix statements**
- **Support of large programs**
- **Support for large data arrays**
- **Symbolic Debug/1000 for compiled BASIC programs**
- **Support for heirarchical file system**
- **Access to compiled subroutines written in BASIC, FORTRAN, Pascal, and Macro/1000**
- **Compatibility with other HP 1000 software subsystems via subroutine calls**

Functional description

The Power Of HP 9800 Series Desktop BASIC

The statement and capability set of BASIC/1000C is built upon a kernel set of approximately 100 statements and functions that are compatible with the BASIC used on the HP 9826A and similar Hewlett-Packard 9800 Series Desktop Computers. Capabilities include single and double integer and two- and four-word floating point data types for a choice of precision, the convenience of multi-character variable and label names, input/output to specific devices, including HP-IB devices, user-defined interrupt handling, formatted and unformatted I/O, and access to disc file storage for data and programs. In addition to providing very good computational power on HP 1000-originated applications, BASIC/1000C's substantial compatibility with Desktop BASIC facilitates migration of applications from single-user HP 9800 Series Desktop Computers to multi-user HP 1000 Computer Systems.

The Ease And Convenience Of Conversational Programming With the Interpreter

BASIC programs are entered into the system via a terminal or a file. The 92857A BASIC/1000C Interpreter checks each statement as it is entered. If a statement contains a syntax error, the line editor is invoked and a message is issued to help the user correct that statement. A built-in debugger helps to correct program execution errors. The debugger can be used to display and change variables, trace program flow control, step "n" lines in the program, set breakpoints, and continue program execution from any line number within the program.

The Efficiency And Speed Of Compiled Programs

HP 92857A BASIC/1000C is the first BASIC subsystem for HP 1000 Computer Systems to include both an interpreter and a compiler. After BASIC/1000C programs have been entered, tested, and debugged with the interpreter, the compiler can be used to translate the source code to relocatable object code for loading and execution in the same way as compiled FORTRAN or Pascal programs. Using compiled BASIC programs also helps to make the source code more secure, an important consideration for OEMs and software houses.

Multi-user Operation With On-Line Program Development

Multi-user operation is supported in the RTE system by individually-identified copies of the BASIC/1000C Interpreter, each serving a different user. The interpreter operates in either conversational (program development) mode or run (program execution) mode. Under RTE, several copies of the BASIC/1000C Interpreter can be used for program development while others are running programs. At the same time, developed programs can be compiled by multiple copies of the BASIC/1000C Compiler and previously-compiled BASIC, FORTRAN, Pascal, or Macro/1000 programs can be executing.

Plenty of Space for Program Development and Data

Within the limits of available main memory and disc memory, BASIC/1000C supports programs of up to megabyte size and data arrays of up to six subscripts, with up to 32767 elements per subscript. This is done by using the VMA/EMA and program segmentation facilities of the RTE operating system. To use the large program or large data array capability, interpreted programs require no special action; compiled programs require the use of compiler directives to manage VMA/EMA and the use of BASIC statements to manage program segments. On RTE-A with VC+, program segments are managed by the operating system and no special action is required by the user. These compiler directives and statements are ignored by the interpreter so that a program can be interpreted or compiled.

Plus the Ability to Take Advantage of Existing Software

BASIC/1000C can call subroutines written in other programming languages. Thus, applications in BASIC can take advantage of existing software, or subroutines can be written in the most efficient language, such as Macro/1000 Assembly language, and even optimized through the use of program profiling, where maximum execution speed is required.

How BASIC/1000C Relates to Other BASIC/1000 Subsystems

With respect to capability, BASIC/1000C is generally a superset of BASIC/1000D and BASIC/1000L. BASIC/1000C differs from BASIC/1000D and L in that it offers multi-user real-time interrupt handling instead of the single-user real-time task scheduling of BASIC/1000D and L. Also, some BASIC/1000C functions have different keywords and/or different ordering of parameters. Because of these differences, programs written for use with BASIC/1000D or L will require some modification for use with BASIC/1000C.

How BASIC/1000C Extends the Scope of Desktop BASIC

BASIC/1000C matches HP 9800 Series Desktop BASIC as closely as is practical. However, unlike Desktop BASIC, which is designed for the single-user, BASIC/1000C takes advantage of the more powerful real-time, multi-user environment of the RTE operating system and its subsystems. This results in some differences in program statement repertoire and structure. Also, RTE software subsystems, such as Image, are accessed by program calls, rather than direct, firmware-based BASIC statements as in 9800 Series Desktop BASIC. To accommodate these differences, programs written in Desktop BASIC will require some modification to run in BASIC/1000C.

Functional specifications

Environment

Operating system: HP 1000 A-Series Computer System with hard disc operating under RTE-A or HP 1000 E/F-Series Computer System operating under RTE-6/VM that meets the minimum hardware of the operating system plus sufficient memory for all concurrent program development and program compilation sessions.

Interpreter memory requirements: The interpreter consists of two programs, the BASIC Editor, which requires a 32-page (64kb) partition for each user, and the BASIC Executor. Partition requirement for the Executor is 200 pages (400kb) for each user. Unless a program uses strings or arrays larger than 2kb, no noticeable performance improvement is realized in a partition larger than 200 pages.

Compiler memory requirements: The compiler requires a partition of 190 pages (300kb). Compilation speed ranges from 500 to 650 lines per minute.

Disc requirements: The compiler, interpreter, and auxiliary files require at least 10 Mb of disc space to load and run.

Program Scheduling In The RTE Environment

Interpreter and compiler scheduling: The BASIC/1000C Interpreter and Compiler are operator-scheduled programs, multiple copies of whpch can be scheduled at the same time by different users.

Interrupt-scheduled execution of subprograms within user's programs: BASIC/1000C includes an ON INTR statement that can be used to cause the execution of a subprogram in response to an HP-IB SRQ interrupt from a specified device in RTE-6/VM or HP-IB SRQ, GPIO, and terminal interrupts in RTE-A.

Scheduling of compiled programs: In addition to having the ON INTR scheduling capability of the interpreter, compiled BASIC/1000C programs can be scheduled within the RTE operating system by operator, time, event, or another program, in the same way as any other compiled or assembled program.

Program Data Types

Data Type	Precision (digits)	Size (bits)	Range
Single Integer	5-6	16	−32768 to 32767
Double Integer	9-10	32	−2,147,483,648 to 2,147,483,647
Short	6.6-6.9	32	$\pm 1.46937 \times 10^{-39}$ to $\pm 1.70141 \times 10^{+38}$
Real	16.3-16.6	64	$\pm 1.46936793852786 \times 10^{-39}$ to $\pm 1.70141183460469 \times 10^{+38}$
String	1-32767 chars.	8-bits per char.	256 char., full ASCII

Program Data Capacity

BASIC/1000C supports arrays with up to six subscripts, up to 32767 elements per subscript, for a total array size of approximately ($2^{27}-600{,}000$) bytes, using the EMA/VMA capability of the RTE system.

Program Form

Acceptability of lowercase: Except for keywords, which must be in uppercase, all program content can be in uppercase and lowercase. Names resembling keywords which are not keywords must contain at least one lowercase character.

Descriptive error messages: Error messages describe the nature of errors, in most cases sparing the user the inconvenience of consulting the manual to interpret error numbers. However, error numbers are included in most error messages to provide a reference to more detailed information.

Numeric type defaulting: Floating point constants, and variables that are not specifically given a type in the program, have a default floating point type that may be set by the user.

Variable and label names: Names of variables and labels may use up to 15 significant characters, including uppercase and lowercase letters, digits, and underscores. An all caps variable or label must not also be a keyword.

BASIC/1000C and RTE Subsystems

The BASIC/1000C Interpreter and Compiler work with the following RTE subsystems:

1. 92841A Graphics/1000-II Version 1.0 Device-Independent Graphics Library
2. 92069A Image/1000 Data Base Management System
3. 91750A DS/1000-IV Network Software
4. 92860A Symbolic Debug/1000 (compiled BASIC programs only).

The BASIC/1000C Interpreter also works with the 92842A Graphics/1000-II Version 1.0 Advanced Graphics Package with a limit of 60k bytes of AGP routines accessible from any one BASIC program.

Compiler Options That Affect the Program Listing

Lines per page.

List on/off.

Page feed for control of program listing.

$BASIC or **$TITLE** for headings on load map or listing.

Compiler Options That Affect Program Structure or Execution

Default two/four-word floating point for setting size of all floating point constants and default-type variables.

EMA for specifying common blocks to be stored in EMA/VMA.

$PROGRAM directive for specifying program name.

$CDS ON/OFF for specifying VC+ support for A-Series systems.

$DEF for specifying external function, return type, and parameter type.

$SUB for specifying external subroutine parameter type.

$SEGMENT for specifying segment boundaries in single-level disc segmentation.

$REALTIME ON/OFF for specifying interrupt handling (requires extra code).

Program priority designation for RTE system.

Range on/off for control of run-time range checking for expression overflow. Turn-off of range checking significantly improves program execution speed and reduces memory requirements.

Ordering information

92857A BASIC/1000C Subsystem

The 92857A BASIC/1000C Subsystem, which must be ordered with one of Use Options 600-603, 700-703, or 890-891, includes:

1. BASIC/1000C Interpreter and BASIC/1000C Compiler on one of software Media Options 022-051, which must be ordered.
2. BASIC/1000C Reference Manual (92857-90001).
3. BASIC/1000C Configuration Guide (92857-90002).
4. BASIC/1000C Quick Reference Guide (92857-90003).

92857A Media Options

022: Provides software on CS/80 cartridge tape.

041: Provides software on 1.2M byte flexible discs*.

042: Provides software on 5-in. Minifloppy discs*.

044: Provides software on 3.5-in. Microfloppy discs.

050: Provides software on 800 bpi magnetic tape.

051: Provides software on 1600 bpi magnetic tape.

* Hard disc is required for use of BASIC/1000C.

92857A/92857R Use Options

600: Use in A600 system.

601: Upgrade from previous version of 92857A/R option 600 to latest version of same for user not on software support service.

603: Upgrade from 92076A BASIC/1000L to 92857A for use in A600 system for user on software support service.

700: Use in A700 or E/F-Series system.

701: Upgrade from previous version of 92857A/R option 700 to latest version of same for user not on software support service

703: Upgrade from 92076A BASIC/1000L or 92101A BASIC/1000D to 92857A for use in A700 or E/F-series system for user on software support service.

890: General license to use in A900 or any other A/E/F-Series System, including right to purchase 92857R Opt 600/700/890 right to copy products for additional systems.

891: Upgrade from previous version of 92857__ Opt 890 or 700 to latest version of 92857__ for customer NOT on support service.

92857R Right To Copy BASIC/1000C For Use On One Additional Computer System

The 92857R Right-to-Copy product, which must be ordered with one of Use Options 600, 601, 700, 701, 890, or 891, is available only to customers who have purchased a license to use 92857A with the same Use Option. 92857R consists of:

1. The license to make one copy of software purchased with 92857A for use on an additional system.

2-4. Same as items 2 through 4 supplied with 92857A.

Software support products available

See page 1-4 of Volume I of the HP 1000 Software Data book.

BASIC/1000L and BASIC/1000D

HEWLETT PACKARD

For program development in RTE-XL, RTE-A, RTE-6/VM, or RTE-IVB

product numbers 92076A and 92101A

Hewlett-Packard's BASIC/1000L and BASIC/1000D are Mature software subsystems for conversational development, and execution of Real-Time BASIC programs in computer systems managed by disc-based RTE-XL/A/IVB and RTE-6/VM real-time executive operating systems.

Features

- **Concurrent multi-user development and execution of Real-Time BASIC programs**
- **Conversational programming**
- **Read/write from non-disc peripherals**
- **Time and event scheduled operation for single user**
- **High-level subroutine calls for instrumentation, including multi-instrument clusters bus-connected via HP-IB***
- **Easy access to disc file storage for programs and data or to Image/1000 data base**
- **Character string manipulation with string variables**
- **Support of bit manipulation**
- **Usability of subroutines or functions in FORTRAN or Assembly language**
- **Print with format control**
- **Ability to run BASIC or non-BASIC program as a subprogram**

Functional description

Basic/1000L is a program-compatible subset of the disc-based BASIC/1000D. These BASIC language subsystems support the following capabilities.

Single-user and multi-user operation. The RTE host systems all support single-user operation. In systems with enough memory, multi-user operation can be provided by individually-identified copies of BASIC/1000L or D, each serving a different user. All active copies run concurrently with each other and with other programs in the RTE system, but only one copy can use time and event scheduling.

On-line program development. BASIC/1000L and D operate in either conversational (program development) or run (program execution) mode. In RTE, several copies of BASIC/1000L or D can be used for program development while another is running a program, so BASIC-programmed operations can be extended on-line.

Conversational program development. Real-Time BASIC programs are entered into the system via the system console or another terminal. BASIC checks each statement entered. If a statement contains an error, BASIC returns a message to help the user re-enter that statement correctly, in a conversational process. Program execution errors are flagged and corrected with similar ease.

Character string manipulation. Strings up to 255 characters long can be represented by variables. This provides a shorthand representation of frequently-used strings that can save programming time and effort. It also makes possible the extraction of string segments using subscripts and character-by-character comparison of two strings.

Real-time multi-tasking. The host RTE system provides a multi-program, multi-partition environment in which BASIC/1000L or D operates. This environment provides for multi-user operation via multiple copies of BASIC. User's program code in each of these copies or in single-user BASIC/1000L or D is not just a single task, but can be subdivided into as many as 16 tasks that are BASIC subroutines. This subdivision helps the user to match the frequency, timing, and basis for execution of programmed task actions to the diverse needs of real-time applications. For one of the copies of BASIC in the system, task executions may be scheduled as a function of time or event interrupt (such as contact closure). BASIC/1000L and D recognize priority levels from 1 through 99.

Print with formatting. A useful BASIC/1000L and D capability is the PRINT USING statement for specifying the format in which the variables specified in the statement are to be printed. This format can be in a literal string, a string variable, or in a special statement called the IMAGE statement. With PRINT USING:

- Numbers can be printed in integer, fixed point, and floating point representations.
- The exact position of plus and minus sign can be specified.
- String values can be printed in specified fields and literal strings and blanks can be inserted wherever needed.
- Full control of carriage returns and line feeds is possible.
- Arbitrary long lines can be printed without the carriage returns and line feeds normally provided by the PRINT statement.

Program testing. User requests are provided for tracing program execution, inserting up to four breakpoints, and for simulating execution of subroutine calls. These capabilities are helpful for testing programs on a system different from the target system in which they will be used.

Program statement renumbering. In BASIC/1000L and D, the user can systematically change program statement numbering with a simple command, without retyping statements, a capability that greatly facilitates insertion of additional program statements where needed.

Disc storage of programs and data. BASIC programs are easily saved in named disc files, in either source or semi-compiled (faster-executing) form. The user can also create files on the disc for data storage and retrieval access with simple PRINT and READ program statements. Files are easily renamed or purged to meet changing needs.

**HP-IB (Hewlett-Packard Interface Bus) is Hewlett-Packard's implementation of IEEE Standard 488-1978, "Digital Interface for programmable instrumentation", identical ANSI Standard MC1.1, and IEC recommentation 625-1.*

Program linking. BASIC/1000L and D include CHAIN and INVOKE statements for automatically linking programs together so they run as one long program. The CHAIN statement in the current program retrieves a named program from the disc and starts it running from the first statement or any later statement number that is specified in the CHAIN statement. The INVOKE statement is more capable than the CHAIN statement. It can call BASIC or non-BASIC programs. INVOKEd and CHAINed programs can access previously-opened BASIC data files and use previously-enabled TRAPs. An INVOKEd program may INVOKE another program. When the current executing INVOKEd program terminates, control is returned to the program that called it.

Program editing. Using the interactive editor of BASIC, the user can edit characters within a statement line, leaving some characters unchanged, inserting characters, and replacing or deleting characters as desired.

Data base access. BASIC/1000L and D include an interface to the Image/1000 Data Base Management System. This interface connects BASIC program calls to subroutines of Image/1000, including the routines that open or close the data base, locate, read, update, add, or delete data, and lock or unlock the data base.

Functional specifications

Environment

BASIC/1000L: Disc based RTE-A or RTE-XL system.

BASIC/1000D: Disc based RTE-IVB or RTE-6/VM system.

Basis of BASIC Task Scheduling For Execution

By operator, another task, time, or event (only one copy of BASIC can use time and/or event scheduling) in order of task priority.

BASIC Task Priority Levels

1 through 99, the lowest number designating highest priority.

Program Data Types

- REAL data — a 32-bit quantity with sign, exponent, and mantissa, ranging from $\pm 2^{-127}$ to $\pm 2^{+127}$, with 6 to 7 decimal digit accuracy.
- STRING data — ASCII strings up to 255 characters long represented and manipulated by variables.
- OCTAL data — a 16-bit quantity including sign that can be entered into programs, manipulated, and output using the bit manipulation statements provided in HP Real-Time BASIC.

Program Character Set

- The 26 upper case letters A through Z.
- The ten digits 0 through 9.

Software Support

BASIC/1000L and D do not support the 92400A Sensor-Based DAS Utility Library or the 92413A ISA Fortran Extension Package. The 92840A Graphics/1000 Graphics Plotting Software is not supported in BASIC/1000L, but 92841A and 92842A Graphics/1000-II Packages are supported.

Ordering information

92076A BASIC/1000L System (for use in RTE-A or RTE-XL System)

BASIC/1000L consists of:

1. BASIC/1000L software on one of Media Options 022 through 051, **which must be ordered.**
2. BASIC/1000L Real-Time BASIC Reference Manual (92076-90001).
3. BASIC/1000L Software Installation Manual (92076-90002).

92076A BASIC/1000L Media Options

022: Provides BASIC/1000L software on CS/80 cartridge tape.

041: Provides BASIC/1000L software on 1.2M byte flexible discs.

042: Provides BASIC/1000L software on 5-in. Minifloppy discs .

044: Provides BASIC/1000L software on 3.5-in. Microfloppy discs.

051: Provides BASIC/1000L software on 1600 bpi mag tape.

92101A BASIC/1000D (for use in RTE-IVB or RTE-6/VM system)

The 92101A BASIC/1000D System consists of the following items:

1. BASIC/1000D software on punched tape (alternate Media Options 020 and 022 are available).
2. Multi-User Real-Time BASIC Programming and Operating Manual (92060-90016).
3. 92101A Software Numbering Catalog (92101-90001).

92101A BASIC/1000D Options

001: Provides discount to user upgrading from previous version of 92101A to latest version if not enrolled in the Software Subscription Service or Comprehensive Software Support service.

020: Provides BASIC/1000D software on 264x Mini cartridges instead of punched tape.

022: Provides BASIC/1000D software on CS/80 cartridge tape instead of punched tape.

Software support products available

See page 1-4 of Volume I of the HP 1000 Software Data book.

Symbolic Debug/1000

HEWLETT PACKARD

HP Program Development Software product number 92860A

Symbolic Debug/1000 is an interactive, symbolic debugger for source-level FORTRAN, Pascal, compiled BASIC, and Macro programs on RTE-6/VM and RTE-A based HP 1000 systems. Variables are displayed or modified using names from the original program. Load maps and program listings are not needed. One and two word integer, two, three and four word reals, logical, complex, character, Hollerith and most structured data types are supported. Symbolic Debug resides in a separate partition from the program being debugged to eliminate program code space intrusion. A single-stepping, source-line capability displays the current and adjacent lines during execution. Conditional breakpoints can be used to monitor variable values and stop the program at a specified value. Using the profiling capability, the user can determine which subroutine is using the most program time and optimize the code to decrease execution time. A small, simple command set, the use of dozens of English error messages, and an on-line "help" facility make Symbolic Debug/1000 a friendly and powerful programmer's productivity tool.

```
   40           RE   = DCOS  (ANG)
   41           IM   = DSIN  (ANG)
   42    2      IF  ( .NOT.  NEW  .AND.  K*K0  .GE.  1 ) GO TO 4
   43 C       ------------------------------------------
   44 C         COMPUTE TWIDDLES IF NECESSARY ...
   45 C       ------------------------------------------
   46           U(1) = DCMPLX( RE , -SIGN(IM,DBLE(K)) )
>  47           DO  3  I  =  2,L2N
   48    3      U(I)  =  U(I-1)*U(I-1)
   49           K0 = K
   50 C       ------------------
   51 C         BUTTERFLIES.
   52 C       ------------------
   53    4      SBY2 = N
   54           DO 7 STAGE = 1,L2N

DEB.A> b 47/fft
Breakpoint set at 47/FFT
DEB.A> p
Break at 47/FFT
DEB.A> d L2N new re u(1)
L2N = 5      NEW = true      RE = 0.980785282244344
U(1) = (0.980785282244344,-0.195090312760225)
DEB.A> m L2N 6
L2N: 5 => 6
DEB.A>
```

Features

- **Interprets all code types and symbols used**
- **Can display source code during execution**
- **Non-intrusive — no Symbolic Debug code resides in user space**
- **Supports EMA and RTE segmentation**
- **Supports all simple and most structured data types**
- **Program profiler isolates slow subroutines**
- **Source line-by-line single stepping capability**
- **Up to 50 conditional breakpoints to stop program at specified variable value**

Functional description

Symbolic debug. Symbolic Debug recognizes the names, types, and locations of all of the variables and routines used in the program, eliminating the need for load maps, symbol table dumps, and mixed listings. The value of a variable can be examined as fast as its name can be typed.

Interactive debugging process. The user interacts with the program as it runs and can examine or alter variable values while the program runs without having to insert statements into the code. Bugs can be found fast since there's no need to recompile and load every time a new bug occurs.

Separate partition. Symbolic Debug resides in a 32-page memory partition separate from the program. No code space is lost and no extra statements are added in order to debug. The program being debugged runs exactly the same as it would normally. No bugs are introduced by the debugger, and more importantly, bugs don't disappear when the debugger is present, only to reappear when the debugger is not used. There is no need to restructure a program just to debug it.

Source-level symbolic. Symbolic Debug recognizes what line of source code is about to be executed, and identifies it on the CRT display. Programs can be debugged in the language in which they were written, without the need for inverse assemblies or mixed listings. There is no need to list files at all.

Detects RTE program violations. After a detected violation such as an attempt to access protected memory, memory locations can be examined to determine the cause of the problem. Symbolic Debug pinpoints the line of source code that caused the error, giving the operator an interactive tool for catching system violations.

Standard and conditional breakpoints. Up to 50 breakpoints enable Symbolic Debug to monitor program variables and halt program execution if a variable reaches a specified value. A large number of possible paths can be trapped and values can be quickly tracked through the program to determine where they go wrong.

Supports transfer files and message logging. Non-interactive debugging sessions may occur where users can submit debug commands in a file, and have results logged in another file. This automates the debugging process, so users don't have to wait for bugs whose symptoms may take hours or even days to occur.

Built-in profile monitor. Helps isolate slow parts of the program. High-level analysis of activity distribution within the program helps to identify time-consuming subroutines that should be optimized in order to improve execution time. For example:

Profile for program TEST

Routine	Amount	Histogram
OTEST	39%	**
SUBR	27%	**************************
OTES1	16%	***************
UTILITY	9%	*********
OTES0	3%	***
Other (your code)	2%	**
Other (libraries)	3%	***

Profile for module OTEST: 39% of total time spent here

Line No.	Amount	Histogram
7	20%	********************
8	11%	**********
9	33%	**********************************
13	36%	************************************

Debug command summary

B <Location>	Sets breakpoint at specified location.
C<Location>	Clears breakpoint at specified location
D <Locations>	Displays variable.
E	Aborts your program and exits Debug.
F/<string>	Finds string in source file.
G <Location>	Allows your program to proceed from specified location.
H	Displays histogram.
I <f1 [f2]>	Executes a set of commands from a file (f1) and optionally logs the output to f2.
L <Location>	Lists a screenful of source code in your program.
M <Loc> <val>	Modifies the value of variable.
O	Enters overview mode (enables profile monitor).
P <line>	Allows your program to proceed to the next breakpoint or specified line
S	Steps to the next line of source code.
T <Location>	Shows location executed without stopping program.
V <number>	Changes the number of source lines displayed on screen.
W	Shows callers of the current subroutine.
?	Help facility

Environment

Operating System

HP 1000 Computer System operating under RTE-A or RTE-6/VM, revision code 2226 or later.

Supported Program Languages

92836A FORTRAN 77, Macro/1000, Pascal/1000* and BASIC/1000C (Compiled code)*.

**Effective with A.84 revision.*

Ordering information

92860A Symbolic Debug/1000

92860A Symbolic Debug/1000, which must be ordered with Use Option 600, 700, or 890, includes:

1. Symbolic Debug/1000 software on one of Media Options 020-051.
2. Symbolic Debug/1000 User's Reference Manual (92860-90001)
3. Symbolic Debug/1000 Configuration Guide (92860-90002)

92860A Media Options

020: Software on 264x Minicartridges

022: Software on CS/80 cartridge tape.

041: Software on 1.2M byte flexible disc.

042: Software on 5-in. Minifloppy Discs.

044: Software on 3.5-in. Microfloppy discs.

050: Software on 800 bpi mag tape.

051: Software on 1600 bpi mag tape.

92860A/92860R Use Option

600: Use in A600+ or A600 system.

601: Upgrade from previous version of 92860A/R Opt 600 to latest version of same for customer NOT on support service.

700: Use in A700 or E/F-Series system.

701: Upgrade from previous version of 92860A/R Opt 700 to latest version of same for customer NOT on support service.

890: General license to use in A900 system or any other A/E/F-Series system, including right to purchase 92860R Opt 600/700/890 right to copy products for additional systems.

891: Upgrade from previous version of 92860A/R Opt 890 or 700 to latest version of 92860A/R for customer NOT on support service

92860R Right to Copy Symbolic Debug/1000 for Use on an Additional Computer System

The 92860R Right to Copy product, which must be ordered with Use Option 600, 700, or 890, is available only to customers who have previously purchased a 92860A product. 92860R consists of:

1. The license to make one copy of software purchased with 92860A for use on an additional computer

2 and 3. Same as for 92860A.

Software Support Products Available

See page 1-4 of Volume I of the HP 1000 Software Data book.

Macro/1000 Assembler

HEWLETT PACKARD

For program development in RTE-A, RTE-6/VM, and RTE-XL

included in RTE operating system

Macro/1000 is an extensively-enhanced superset of RTE-IVB Assembly Language developed for HP 1000 Computer Systems operating under RTE-A, RTE-6/VM. and RTE-XL. This assembler is designed to give users complete control over each computer instruction while significantly enhancing productivity through full macro capabilities and high order language type constructs. Macro/1000 offers full upward compatibility with the RTE-IVB Assembler, ASMB, yet assembly performance is approximately double that of the ASMB processor.

Features

- **Structure**
- **Macro capabilities enhancing code portability and program readability**
- **Library of commonly used macros**
- **Conditional assembly instructions**
- **Modularity**
- **File and string manipulation utilities**
- **16 character variable names**
- **Symbolic Addressing**
- **Enhanced error control and reporting**
- **Extremely fast compilations**
- **Compatibility with HP 1000 software subsystems such as ASMB, Image/1000, Graphics/1000-II, DATACAP/1000-II, and DS/1000-IV**

Functional description

Macro/1000 is a superset of the RTE-IVB Assembler with the following extensions:

Powerful Macro Facility that allows the passing of constants, labels, expressions and complete instructions as parameters. Macros can be nested within macros, with the level of nesting limited only by the amount of memory in the program space. Macro/1000 further allows recursion and cross-recursion between macro definitions to enhance the programmer's capabilities.

Conditional Assembly allows the user to programmatically direct Macro/1000 to assemble or ignore a set of statements, based on the evaluation of stated condition tests within the body of the program using the AIF/AELSE/AELSEIF/AENDIF construct. This capability allows several variations of a program to be generated from the same source. Using AWHILE/AENDWHILE, users can specify a set of instructions to be assembled repetitively as long as the conditional requirements of that statement hold true. The REPEAT/ENDREP command directs the processor to assemble a set of instructions a fixed number of times.

String Manipulation Utilities: Four new assembly-time string manipulation utilities have been added, all of which may be used in expressions. The Length construct is a flexible measure of the character count in a string allowing various changes to the string's content without further edits to code used for string manipulation. The Substring utility allows the programmer to operate on segments of a character string. Upper Case Map changes lower case characters to upper case characters, and the Type attribute is used to determine whether an assembly time variable has been declared integer, character or not yet declared.

File Manipulation Instructions offer the ability to merge separate files and assemble them together as independent macros or subroutines. The INCLUDE statement will copy a designated file of code, macro definitions, or data into the source at a desired point. The MACLIB statement allows the user to name a specified file as a macro library.

New Psuedo Operations: Macro/1000 provides 23 new psuedo ops which give the programmer more control over assembly time options. Included are commands to specify where a table of literal values are located, to repeat a sequence of code, fix the size of an Extended Memory Area (EMA), or define an address to EMA.

Functional specifications

Environment

Operating System: Macro/1000 executes on any HP 1000 Computer System operating under RTE-A, RTE-6/VM, or RTE-XL.

Memory Requirements: 36k bytes, minimum; best performance is achieved with more memory, up to a maximum of 64k bytes.

Compilation Speed (CPU time): 5000-6000 lines/minute in M/E/F-Series operating under RTE-A or RTE-6/VM; 1000-2000 lines/minute in L-Series operating under RTE-XL.

Macro Program Execution: Generated code runs under RTE-A, RTE-6/VM, and RTE-XL (rev code 2140).

Character Set

Alphabetic Characters: All upper and lower case characters (A through Z and a through z).

Numeric Characters: The ten digits 0 through 9.

Special Characters: blank; equal, plus, minus, less than, and greater than symbols; period; comma; asterisk; slash and backslash; left and right parentheses; single and double quotes; left and right brackets; colon and semicolon; currency symbol ($); percent sign; question mark; exclamation point; ampersand; at sign (@); sharp sign (#); and carat (↑).

Program Form

Lower Case: Lower case keywords and names are accepted and mapped into upper case for improved readability.

Comment Lines: Comments can be transcribed into a source program in various ways. An asterisk in the first position or semicolon as the first nonblank character is accepted as the start of a comment line. Additionally, characters following a backslash (\) in an operand field are considered comments.

Error Directory: Macro/1000 incorporates one of the most descriptive error message directories available on HP 1000 Computer Systems in addition to the capability to declare and list user-defined errors.

Assembler Options

R: Produces relocatable assembly

A: Produces absolute assembly

L: Produces program listing

C: Produces cross-reference table of symbols, labels and op codes

Q: Omits op code fields from instruction values appearing in listing

T: Produces symbol table listing

M: Creates Macro library

Ordering information

Macro/1000 software and the Macro/1000 Reference Manual (92059-90001) are included with all HP 1000 Computer Systems operating under RTE-A, RTE-XL (effective with rev. code 2140), and RTE-6/VM.

Software support

Software support is included with RTE-A, RTE-6/VM, and RTE-XL support services.

Edit/1000

HEWLETT PACKARD

For program development in RTE-A, RTE-6/VM, RTE-IVB, and RTE-XL

included in RTE operating system

Edit/1000 is a powerful screen editor designed to help the programmer develop software quickly and accurately with minimal effort. Edit/1000 helps the user to create and manipulate files of upper and lower case ASCII characters. Lines, strings, and characters can be inserted, deleted, replaced, copied, or moved within the file. Files to be edited can be source language programs or text material.

Features

- **Convenient screen mode entry and editing of files**
- **Powerful character string search and replacement capabilities**
- **Commands for copying and moving lines within the edit file**
- **Ability to create a new file or back up a partly-edited file without leaving edit mode**
- **On-line commands summary, with ability to obtain more information about any command, easily accessible to the user without leaving the editor or having to refer to a manual**

Functional description

Operating Modes

Edit/1000 interacts with the user through edit commands and is designed to operate in the following modes:

- Screen mode, in which the user types in a screen of text and modifies the text using any of the HP terminal's editing features.
- Line mode, in which edit commands operate on groups of one or more lines.

Screen Mode

In the screen mode, the editor treats the terminal screen as a window through which the user can view a section of text, such as a subroutine. A cursor within this window indicates the character at which editing will take place. The user controls the cursor with the help of the terminal and can also move the window forward or backward any distance. The editing facilities of the terminal are used, which can speed up the editing job if the the system is very heavily loaded and thus slow in responding. In addition, the user is working directly with single keystroke commands which users often find faster and more convenient to use than the line edit commands of the editor.

Advanced Line Editing Capabilities

In addition to the basic text maintenance functions of retrieve, insert, add, change, and delete, Edit/1000 provides sophisticated character string search and replace capabilities that can be used to change any string of characters in the file to another string, or can selectively change character strings within specified columns only. This greatly simplifies the change of variable names throughout a program or terminology throughout a text file.

Merging of source files, merging and breaking of lines, and powerful copy and move functions facilitate electronic cut and paste rearrangement of program code or text in the edit file. Copying of repeated table headings or copying and editing of similar text to various appropriate points in the edit file can be a particularly powerful time and effort saver.

Display of lines before and after the current line, viewing the previous or next screen, tab setting, and the ability to undo modifications are other highly useful capabilities built into the Edit/1000 editor.

Interactive Instructions

Edit/1000 provides for on-line interactive explanation of its commands to help the casual user who may have forgotten the available command set or the usage of certain commands.

On-Line File Creation and Edit File Backup

Edit/1000 also provides for creation and storage of the current edit file without leaving the editor. Similarly, the current state of the edit file can be backed up, also without leaving the editor. This is particularly helpful in safeguarding the time and effort invested in a partly completed edit.

Functional specifications

Environment

HP 1000 Computer System operating under RTE-A, RTE-IVB, RTE-6/VM, or RTE-XL.

Character Set

All ASCII characters.

Ordering information

Edit/1000 software and User's Manual (92074-90001) are included on HP 1000 Computer Systems operating under RTE-A, RTE-IVB (effective with rev. code 2026), RTE-XL (effective with rev. code 2140), and RTE-6/VM.

Software support

Software support is included with RTE-A, RTE-IVB, RTE-6/VM, and RTE-XL support services.

HEWLETT PACKARD

Printed in U.S.A. 6/84
5953-8711