The 71B is a brand new machine with computing power far beyond most microcomputers and scientific calculators found on the market today.

HP 71B: A FISTFUL OF POWER

BY PHIL KOOPMAN

At first blush, Hewlett-Packard's pocket-size HP 71B looks like a cross between the HP Series 41 calculators and a stripped down HP 75 portable.

It's not. The 71B is a brand new machine with computing capability far beyond most microcomputers and scientific calculators. And for the first time with a small computer, HP has made available the Internal Design Specifications (IDS) with source code and schematics.

While one can't actually call the 71B a pocket IBM 370, it's close. Aptly code-named Titan, this battery-powered device addresses up to half a megabyte of memory. Maximum on-board memory is 320K bytes—via four plug-in ROM and RAM memory modules—including the internal 64K ROM and 17.5K RAM memory. The 71B works as an immediate-mode calculator or as a computer. The base unit, complete with HP's Extended BASIC, the operating system (OS) in 64K of ROM and 17.5K of RAM, sells for under $550.

The 71B signals a major change in direction for HP in the design of its small computers and calculators. Sal D'Auria, product manager at HP, calls it "the bridge between programmable calculators and computers." In addition to making the 71B an "open" machine, HP abandoned the well-known postfix notation used in its scientific calculators in favor of a sophisticated algebraic calculator mode. The calculating engine is also all new. Rather than the multichip 16-bit processors used in computers like the HP 86, the go power for the 71B is a new 4-bit CMOS microprocessor.

Backing HP's push for third-party support is a complete software-development system that comes in two versions. Adding the optional HP-75-style strip-card reader and HP-IL interface allows up- and down-loading of programs to most personal computers via the software-based HP 71 Development Utility. Tacking on the development ROM allows software development in BASIC, FORTH and assembly language. Within this environment the 71B becomes a slave to the development micro—the display and keyboard are transferred to the host computer. If the host machine doesn't have an HP-IL interface, an RS-232-C adapter hooks the HP-IL loop to most popular micros such as the IBM PC or even the low-cost Atari.

HIGH-POWERED CALCULATIONS

Like numerous pocket computers such as those from Sharp and Radio Shack, the 71B can be used for straightforward programming in BASIC—but with a vast difference. Its on-board BASIC in ROM, with over 240 commands, is geared for scientific, engineering and statistical calculations. Unlike most other ROM-based BASICs, the 71B's BASIC can be extended for special applications. These extensions can be either soft, via data loaded from the card reader into ROM, or added with ROM the way HP extends BASIC with keywords that control the HP-IL interface loop.

HP's BASIC for the 71B is an extended version of the ANSI minimal-BASIC standard. In addition to offering statistical functions, HP tacked on the expected trigonometric functions and unexpectedly built-in floating-point math conforming to the unfinalized IEEE standard. Most microcomputers that have IEEE-style floating-point math use a special co-processor chip, like Intel's 8087, used with 8086/88 microprocessor-based IBM-PC type computers. And the cost of a co-processor chip alone for these math functions is about half the price of the 71B.

Coupled with ROM-based applications packages such as HP's Math Pac, the 71B becomes a high-powered computer that cranks out calculations rivaling those that can be done on most general-purpose computers, including minis and even some of the small mainframes.

The owner's manual follows the step-by-step approach so that beginners and experts rapidly become familiar with the operation of the 71B.

Plugging in Math Pac lets the 71B operate on complex variables and arrays, advanced real- and complex-valued functions and real and complex arrays, and solve systems of equations. Adding to the operations provided by Math Pac gives the ability to solve the roots of polynomial equations and user-defined functions, and perform numerical integration and do finite Fourier transforms. And many of these functions, incorporated into the 71B as BASIC keywords—binary-base logarithmic functions, for example—can be used in the one-shot immediate-calculation mode.

THE HEART OF THE 71B

The 71B is built around a 4-bit CPU running at 465 kHz optimized for fast processing of binary-coded decimal (BCD) data. The data path is only 4-bits wide—most microcomputers use at least 8-bit data paths—making it a "nibble" (4-bit) processor. Internally, the nibble-oriented memory is addressed with a 20-bit program counter. This 20-bit address can handle up to half a megabyte of memory.

To yield the high accuracy needed for scientific calculations, the 71B has a bevy of large-capacity registers. There are four 64-bit work registers, five 64-bit scratch registers, a pair of 20-bit data-pointer registers, a 16-bit input register and one with 12 bits used for data output. Matching the 20-bit program counter is the return-address stack, eight levels deep, also with a 20-bit capacity. Rounding out the hardware are the status registers. There are four hardware status bits, a carry bit and 16 program-status bits, or flags. The lower 12 flags of the status register can also be used as a 12-bit data register.

The low power consumption of these CMOS chips and associated RAM and ROM modules allows extended operation on dry batteries. Also incorporated into the 71B is a power-down "sleep" mode that retains all data currently in memory. A program command, manual operation of the off key or a 10-minute time-out puts the 71B into the sleep mode so that it uses only a trickle of energy. The unit also runs off a 120V AC source with an optional power adapter.

An integral part of the 71B, like its big brother the HP 75, is a clock and calendar. It operates continuously, even in the sleep mode. Accessible with BASIC commands, the clock/calendar allows applications programs to work with real-time data—for example, tagging a data input with the time and date of entry.

The LCD single-line display is a dot (pixel) matrix with 8 by 132 elements. It provides a 22-character window on the contents of the 96-character input buffer. Control keys allow jumping to either end of the line or scrolling across the line. The display's scroll rate, contrast and viewing angle are all programmable. For graphics or redefinition of the character set, the display can be addressed on a pixel-by-pixel basis. Data can also be read directly from the display into a program. This gives display operation like that of a memory-mapped CRT display used with most microcomputers.

The keyboard has a modified calculator-style qwerty layout plus a 16-key numeric keypad. With the exception of the two control keys, the default key definitions yield three functions for each key. Typing aids, accessed by shifting the regular keys, allow entry of the more commonly used BASIC keywords with a single keystroke. Editing, scrolling (both up and down as well as horizontal scrolling of the 22-character display window) and upper- and lower-case character control are also part of the basic keyboard.

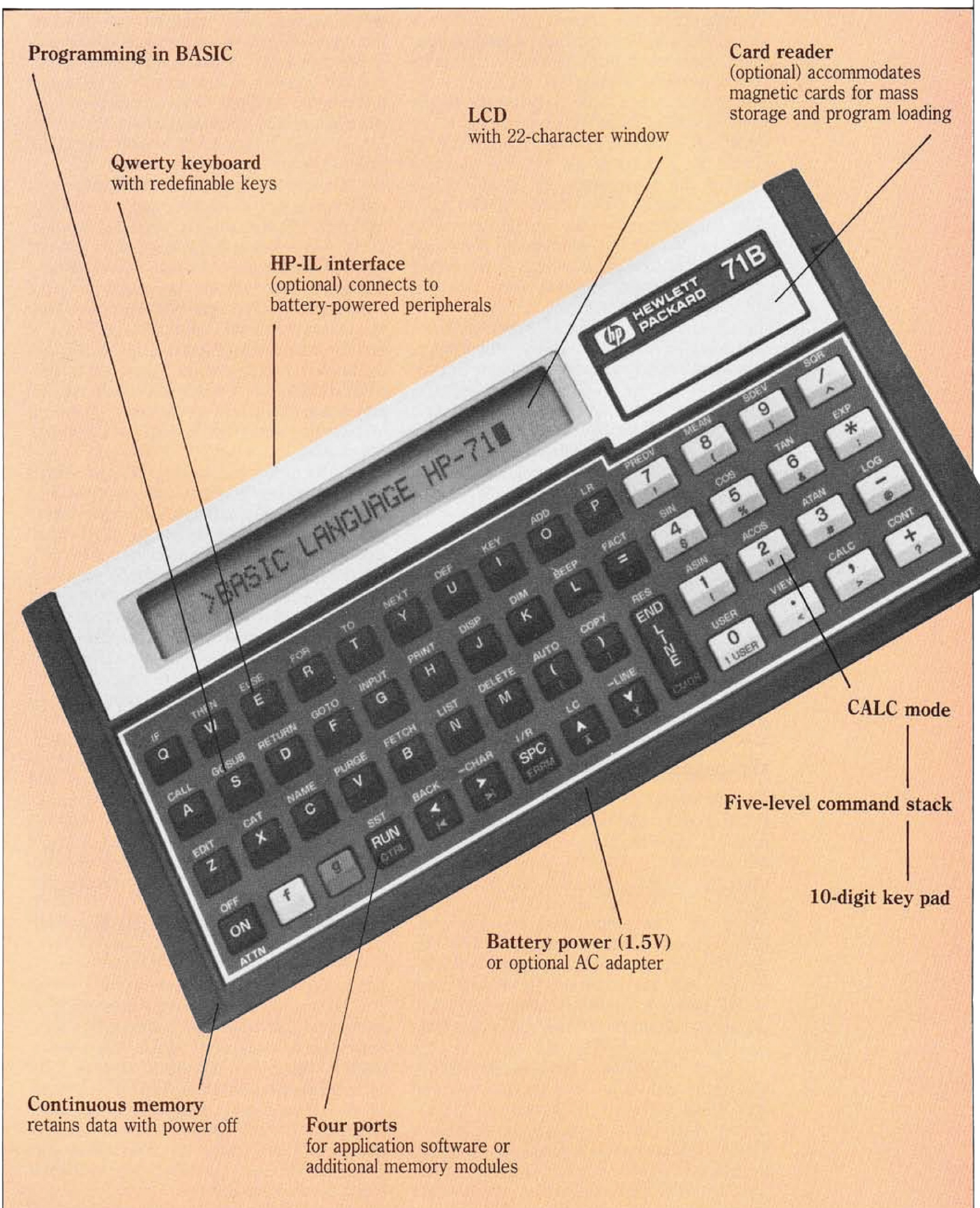
The 12-ounce works of the 71B come wrapped in a plastic case measuring 3⅞ by 7½ by 1 inch. Accessory hatches allow the addition of four ROM/RAM packs, a manual-pull strip-card reader and an HP-IL interface for communication with the outside world. All add-on options and the four AAA battery cells fit within the case.

A NOT-SO-BASIC BASIC

Adding to the flexibility of the 71B's BASIC is the ability to "call" subprograms as done in FORTRAN. This permits the development of a library of often-used subprograms or entire programs that can be loaded into memory for access by other programs. Unlike most implementations of BASIC, the 71B permits multiple programs and subprograms to be in memory at the same time. These programs can be chained, the way most BASICs permit, or use the FORTRAN-style call of subprograms.

The subprogram concept gives enormous flexibility to the programmer. It enables parameters to be passed between the executing program and the subprogram and the reverse. Also permitted is the use of a complete BASIC program as a subprogram call. Once executed, a BASIC program such as a keyboard-definition routine, returns control to the calling program. A forced reload of the caller, required if the CHAIN command were used, isn't needed.

When a subprogram call is executed, the


Programming in BASIC
Qwerty keyboard
with redefinable keys
HP-IL interface
(optional) connects to
battery-powered peripherals
LCD
with 22-character window
Card reader
(optional) accommodates
magnetic cards for mass
storage and program loading
CALC mode
Five-level command stack
10-digit key pad
Battery power (1.5V)
or optional AC adapter
Continuous memory
retains data with power off
Four ports
for application software or
additional memory modules
HEWLETT PACKARD
71B
>BASIC LANGUAGE HP-71

In the reference manual, all keywords are defined, flow-charted and explained with examples—a badly needed format for advanced programmers.

OS first checks the current program executing. If the call is not found within that program, a search is made of all other program files in memory. Failing that, a check for the subprogram is then made of each of the plug-in ROM modules. If the subprogram call is not found, and passes no parameters, then the OS checks for a program file with the specified name of the subprogram. To avoid conflicts between programs, the OS creates two operating environments—one for the executing program, one for the subprogram. New variables that do not disturb those of the calling program are created unless these variables are passed as parameters.

The 71B also provides for recursive programming—a process where a subprogram calls itself. In the same way a main or subprogram calls another, the OS creates a separate environment for the recursive call of the same subprogram. Typically, recursive calls are used for advanced calculations such as Fibonacci series. A single subprogram containing the calculation algorithim calculates a value, passes the data to itself and does the next calculation.

Recursive programming is also useful for non-math programming. A good example is a recursive on-line help routine. Entering the word HELP or pressing a HELP key calls a subprogram that displays the basic commands for that part of a program. If the user needs more detail, HELP should be entered again while still in the HELP subprogram. This recursive call gives greater detail, such as how to use specific editor commands.

KEYBOARD REDEFINITION

Another rarely found feature of the 71B's BASIC is the ability to use either labels or line numbers. Specific numbered program lines can be labeled—other parts of the program using the routine simply tack the label rather than the line number onto a GOTO or GOSUB statement. The programmer also has complete control over the keyboard and the display. Topping off the flexibility of HP's BASIC for the 71B are additional extensions such as the ability to redefine the keyboard and display and do language translation of program messages.

User redefinition of the 71B's keyboard is a powerful feature for both programming and applications software. All keys, with the exception of the shift keys, can be redefined. These definitions do not disturb the standard keyboard functions since the keyboard can be switched back and forth between the standard definitions and user definitions. Also, the new definitions can be intermingled with those of the regular keyboard since redefinition is on a key-by-key basis for the unshifted keys and the two types of shifted keys. For example, the lower-case character set can be redefined as additional BASIC keywords. This switching can be done manually via the keyboard or under program control.

The keys can be redefined either by keycode number or by the key or key-combination markings. FC, for example, is the combination of the *f* shift key and the letter C key. Two types of control over the new definitions are provided. The entire keyboard can be switched to the new definitions and left that way or switched for one-time use. Press a redefined key, the redefinition executes and the keyboard returns to its normal set of definitions. Additional BASIC commands let the programmer view and edit the new definitions.

The keyboard can be redefined in three ways:

1. By adding typing aids, like those built into the machine for single-stroke entry of BASIC keywords. This is a boon for programming, particularly if you take advantage of the label capability of BASIC or are working in FORTH.

2. For use in the immediate-execution mode. Pressing the redefined key adds the new definition to the end of whatever command sequence is in the display buffer. The computer then executes the entire line.

3. For direct execution of a complex command sequence, like loading and running a specific program. This data is executed without being displayed. One use for direct-execution definitions is as a user response to a program's input request. When you press the key, the command sequence is displayed and executed.

User redefinition of the keyboard is a powerful feature for both programming and applications software.

Adding to the keyboard's flexibility are commands that scan and control keystroke entry. These commands enable the user to confirm that a key has been depressed or determine which key was activated. A third command lets a program "press" a key. For example, a program might enter the screen-clear key (ATTN) during operation. When the program ends, the ATTN-key command is executed, clearing the display.

The programmer is also given complete

control over the display and 256-character set. Codes for characters 0 through 127 are preset and are also the default characters for 128 through 255. However, 128 through 255 can be redefined provided that each character's dot pattern on the display can be contained in a pattern six dots wide by eight dots high.

The display window is also under program control. Parts of the display, like a text message to enter a number, can be protected. This prevents the user of an application from backing up the cursor and entering part of the message as a program-input response—a common feature on terminals for data entry but not usually available on microcomputers.

Display and print-control extensions to HP's BASIC are also part of the base 71B by providing commands for print-formatting a string, like IMAGE and PRINT USING, as HP does in its larger computers. The display- and print-line width are also programmable. Adding to this is the ability to control the cursor and change the end-of-line character.

Rounding out the memory-mapped style of the 71B's display is graphics control. Characters can be read directly from the display by a program. Similarly, individual columns of dots can be read and input to a program. Conversely, the dot patterns can be replaced on the display in any position—again a feature normally found as the GET and PUT functions for a memory-mapped microcomputer's video display.

LOCK AND LOAD

Two other commands available on the machine are also unusual for a pocket computer. A user can "lock" the computer with a password. The lock can be activated even with a program running, preventing unauthorized access to program data, at the same time allowing an applications program to be run. Once activated, the 71B requests the password when turned on. If the password's not correct, the machine turns itself off. Without the password, the only way the lock function can be bypassed is to clear all memory. With no program or data in memory, there is nothing left to protect.

A final touch is added with the startup command. When the 71B is turned on, it executes a predefined command sequence, such as loading and running a specific program. This command functions in much the same way as a command file for some of the more advanced microcomputer operating systems that provide mainframe-style features.

If one delves deep into the extensive IDS documentation—the shipping weight was over

SPECIFICATIONS

DIMENSIONS
19 cm (7.5 in) × 9.7 cm (3.8 in) × 2.5 cm (1 in)

WEIGHT
340 g (12 oz) with batteries

POWER

Batteries	four 1.5V, AAA batteries (replaceable by user)
Battery current	.03 mA (off) .75 mA (idle) 10 mA (operating)
Average alkaline battery life	60 operating hours (battery life depends on use)

OPERATING REQUIREMENTS

Operating temperature	0° to 45°C (32° to 113°F)
Storage temperature	−40° to 55°C (−40° to 131°F)
Humidity	0% to 95% relative humidity

DISPLAY

Capacity	8 × 132 dot matrix; 22 characters, each 5 × 8 matrix, scroll to view 96-character buffer; 14 annunciators.

CHARACTER RANGE
A-Z, a-z, 0-9, plus 65 special characters.

NUMBER RANGE
−9.99999999999E−499 to
+9.99999999999E+499, plus zero.
Numbers are shown with a maximum of 12 digits, or a 14-digit mantissa and a three-digit exponent. Calculations to 12-digit accuracy.

MEMORY CAPACITY
Built in: 17.5K bytes
Maximum extended memory: 33.5K bytes (with four 4K-byte memory modules)

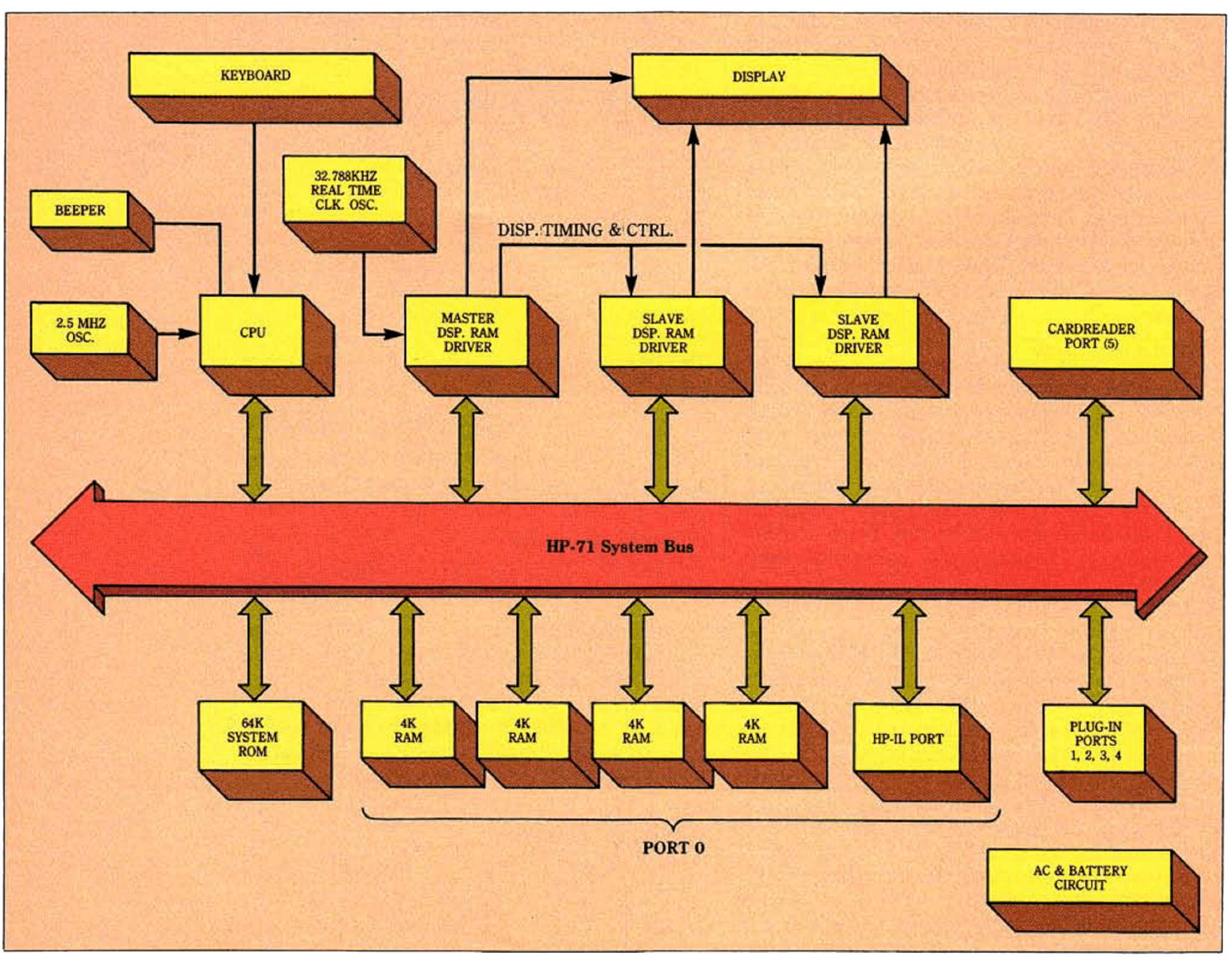
30 pounds—many details of special functions are revealed. One of these is the ability to use a foreign language as program inputs without rewriting the program. The same applies to the computer's error messages.

This ability is an example of soft extensions to HP's BASIC and the firm's commitment to the international market. Available from the HP User's Library as LEX File 82, the BASIC extension adds the keyword MSG$. Its operation is similar to that of the regular error-message command, ERRM$. Where the error-message function returns the last error or warning message to a program, the message-string function returns a message from a predefined message table rather than from text within the applications program itself. When you change the message table in memory, the language of the program changes.

This message-table concept allows a program to be used in any language—translation of input and output are automatic. As an example, the 71B's text editor stores all of its commands, program responses and a help facility in a message table. User inputs are compared to the message table with the MSG$ functions. This language-translation function drives the text editor in any language. Program changes aren't needed—just a new message table in the desired language.

The scheme also allows multiple message tables to be part of an applications program. Message-table selection is controlled via a keyword in the controlling translator. These tables are ROMable, making it possible to supply a specific language, like Spanish, in ROM, or to include multiple languages within the same applications program in ROM as

HP 71B **system block diagram**



Adding to the flexibility of the 71B's BASIC is the ability to "call" subprograms as in FORTRAN, easing the burden of the programmer.

HP does with the 71B's text editor.

THE OPERATING SYSTEM

While the hardware of a computer is important, what makes up a complete computer is the meld of the operating system (OS) software and hardware. This determines the true architecture of the computer and controls its capabilities. As in most modern microcomputers, the core of the 71B's operating system is in ROM. Extensions to the system, such as external file handling on mass-storage devices via the HP-IL interface, are added to the OS core with additional ROM. These additions can also be in software and loaded into the system from magnetic cards or from a mass-storage device such as a tape drive.

The core, or kernel, of the 71B's OS performs control functions. The kernel and the BASIC interpreter reside in the internal 64K ROM that is part of the basic machine. The OS processes up to four different file types. It interprets or processes BASIC, binary machine code (BIN), language extension (LEX) and FORTH files. BASIC files are developed and translated with the standard system. Developing BIN, LEX and FORTH files requires the FORTH/Assembler ROM. Executing FORTH files also requires the FORTH/Assembler ROM.

The BASIC files are stored in token form. The English-like BASIC words are translated into space-saving tokens to reduce the amount of memory needed and to speed up program execution. BASIC or BIN program files can be executed as a program or a subprogram.

The LEX files add extensions to BASIC and also the message-table capability. When a LEX file is added to the machine, the OS registers its presence. This tacks on the added commands to BASIC, making the extensions an integral part of the interpreter.

A key part of the OS is the presence of software "hooks," called "polls" by HP. These polls let a LEX file intercept machine operation at key points and then modify how the machine works. There are over 80 points in the OS operation—at the end of parsing a line of BASIC code, for example—where all LEX files are checked for an intercept and revectoring of the computer's operation to a different or special operation. Conversely, LEX or BIN files can access routines such as a BCD math calculation or file-handler within the OS. Effectively, these entry points use parts of the OS as efficient, debugged machine-code subroutines. HP guarantees that these entry points into the OS, as documented, will be maintained in future releases of the OS. If the user prefers, the entire OS can be switched out and replaced with a custom operating system.

Also part of the OS is the basic file-handling structure. The 71B uses a memory-based file system that handles files in serial form within the memory—no central file directory, common with disk-based systems, is used. In effect, programs in memory are one large file, with the splits between the programs identified with that program's name.

The 71B treats the files within a ROM module in a similar manner. Each ROM module contains its own file chain in the same format as those in regular memory. With a RAM module there are two options. The files, still in the chain format, can be maintained as a separate environment or can be merged and become part of the main file chain. This pooling of files is an automatic operation of the OS.

File partitioning in RAM or ROM allows flexibility in program design. All of the system's five memory ports can be controlled by a program or by the user. RAM modules, including the internal 16K RAM, can be partitioned in 4K increments for use as either system RAM or independent RAM. All RAM, with the exception of the internal system RAM tagged as Port 0, can be extended. Ports with RAM or ROM can be cut into or removed from the system's software. By specifying a particular port or a partitioned section of a port, a program can more quickly locate data. It also allows a user to remove a memory module without disturbing data in the remainder of the system RAM.

The 71B signals a major change in direction for HP . . . It's the bridge between programmable calculators and computers.

In addition to the tokenized BASIC program code, the OS handles data files. These data files can be sequential or random-access. Full control of data access is provided, including resetting of the random-access file pointers.

There are three types of files: regular data files that contain numeric values or strings of characters, text files formatted so they can be read by other computers such as the HP Series 80, and a special data file, SDATA, that has the same format as data files created by the HP Series 41 programmable calculators.

HP made the 71B an "open" machine and abandoned the well-known post-fix notation in favor of a sophisticated algebraic calculator mode.

The text files can also be accessed sequentially or randomly; however, data can only be stored in a text file sequentially. Data files can also be used for array storage. This data is stored linearly, allowing data access with or without the array format. These data-file arrays can be one- or two-dimensional.

ALL THOSE TINY KEYS

The 71B's keyboard has keys like those of the HP Series 41. Even though the alphabetical keys are closely spaced, it's quite easy to type in text, and typing allows single-keystroke entry of the more common BASIC keywords.

However, when you switch between regular typed inputs and then use a shift key to access either a keyword input or a character like the heavily used quote mark, there's a break in the input sequence. Since the keywords and special characters aren't in a typewriter-style format, you have to search for them, which almost forces you to take your fingers off the keyboard. It's like going from touch typing to hunt-and-peck.

With the tremendous program capacity of the 71B, the small keyboard makes entering large programs a slow and tedious process. Except for learning the system's operation and entering small programs, the small keyboard becomes a limiting factor for programming. For use with ready-to-run programs like Math Pac, the small keyboard presents no problem, just as thousands of users of the Series 41 have little difficulty. Fortunately, designers recognized the difficulty of extensive program development with the 71B's small keyboard and corrected it with a utility program.

THE BASIC DEVELOPMENT UTILITY

According to HP, the design guideline for the 75 was to make the machine as small as possible and still maintain touch-typing ability. This guideline was obviously impossible for the pocket-sized 71B. To solve the problem, HP came up with a utility program. BASIC programs are written as text files on micros such as the HP 86 or IBM PC. This data is then down-loaded to the 71B via the HP-IL data link. The 71 Development Utility controls either up- or down-loading of BASIC programs in a text-file format. The program comes on magnetic cards, so the 71B must be expanded to include a card reader. And the HP-IL interface also has to be added.

However, the internal works of the 71B, as in most HP micros, operates with tokenized BASIC rather than BASIC source code in an ASCII-text format. The 71 Development Utility also contains a TRANSFORM function. This function changes the received ASCII-text BASIC source code into the tokenized form used by the 71B. It reverses the process so that the 71B's programs can be up-loaded to the host development computer in the ASCII-text format.

The 71 Development Utility manual shows how to use the software with different interfaces to the HP 86/87, including HP-IB and HP-IL, and how to run it with non-HP computers, too. Source code, written in Microsoft BASIC, is given for using the IBM PC as the host computer. This code, with possible minor changes, is easily transported to other micros, like the Atari or Commodore machines, since most versions of Microsoft BASIC are similar. However, since direct connection of the HP-IL isn't possible with most non-HP machines, an HP-IL to RS-232-C convertor will be needed. And the host computer will, of course, require an RS-232-C interface.

ADVANCED SOFTWARE DEVELOPMENT

Taking the BASIC Development Utility one step further, HP's engineers and software team also provided for large-scale program

GOOD FEATURES

1. Small size
   Portability
   Battery operation
2. Multiple languages: BASIC, FORTH, assembly language.
3. Complete software and technical information available, including source code and schematics.
4. Total on-board memory capability of 320K bytes allows large-scale program operation, including complex mathematical calculations.
5. Development systems make large-scale programming feasible by third parties.

BAD FEATURES

1. Keyboard: Keywords and special characters aren't found on the key with the corresponding initial letter—THEN, GOSUB and PRINT, for example, are on W, S and H. This slows down data entry by forcing a change from touch typing to hunt and peck.

development. Adding the FORTH/Assembler ROM to the 71B lets the user link the tiny machine, via the HP-IL interface, as a slave computer to a regular micro.

The FORTH/Assembler ROM contains a FORTH operating system, an assembler—the same as used to develop the 71B's OS—and a text editor. It also adds to BASIC the keyword KEYBOARD IS.

Instead of just writing code in a text-file format and then down-loading it to the 71B, adding the FORTH/Assembler ROM lets the host computer take full control. The display is transferred to the host with the DISP IS command (contained in the HP-IL ROM) and keyboard control is taken over with the KEYBOARD IS statement. This permits real-time program development and debugging on the host computer—remember, the host is still working as a dumb terminal. All operations are performed within the 71B. In fact, a computer isn't even required. A terminal alone will do.

PROGRAMMING IN FORTH

FORTH's fast operation compared with BASIC and its compact compiled form makes it attractive for writing applications programs. The FORTH operating system coexists with the 71B's BASIC operating system. This allows a program to switch between FORTH and BASIC without program or data loss. Programs written in one language can execute programs written in the other.

This ability to interchange between FORTH and BASIC also solves one problem that exists with all versions of FORTH. Without large-scale non-standard extensions, FORTH is more suitable as a control and processing language—text-string handling with FORTH is primitive. With the OS handling interlanguage processing, HP's FORTH then has full access to the BASIC interpreter's string-handling routines.

FORTH is also a screen- or block-based language in most disk-based versions. It contains the language, and most versions are also the operating system for the computer. Instead of using conventional data files, most versions of FORTH use a block of disk sectors organized in 1K-byte capacity "screens". Again, the OS-controlled interplay between BASIC and FORTH adds a new dimension to HP's version of FORTH—conventional file handling and file access. This is another feature that exists only with extensions to FORTH rather than being part of the base language. The bottom line is that HP's implementation of FORTH melds the best features of FORTH and BASIC.

The implementation of HP's FORTH differs slightly from most versions of FORTH. Most versions are for regular 8-bit microprocessors and use 16-bit addressing. HP's FORTH apes the structure of the 71B and provides for 20-bit addressing. This allows FORTH to access the entire 1 meganibble address space of the 71B. It also changes the way the FORTH stack works—numeric quantities are 20-bits, not the usual 16-bit values. While the usual FORTH primitives, like @ (pronounced fetch) are retained, their operation is in the 20-bit mode. Byte-oriented FORTH words, like C@ (character fetch) work in the same way as conventional 16-bit FORTH implementation. Since the 71B is a nibble-oriented machine, HP added nibble words. An example is N@; it places the contents of the addressed nibble on the stack the way C@ places the contents of a particular byte on the stack.

THE FORTH-BASIC INTERPLAY

The interplay between FORTH and BASIC is built around a set of keywords that pass data between the two languages. To use BASIC's floating point routines, for example, FORTH passes a string containing the numeric data to be evaluated, followed by the keyword BASICF. BASIC evaluates this data and returns the answer to the X Register in the FORTH floating-point stack.

Similarly, BASIC can access FORTH. The keyword FORTHX is a BASIC statement that returns the contents of FORTH's floating-point-stack X Register. Strings and other data can be passed between FORTH and BASIC in the same way.

The interaction between the two languages also extends to control of machine functions like the data display mode—decimal, engineering or degrees/radians. Either language can set the output mode. The FORTH word that switches output between degrees and radians also switches any calculation results from BASIC to that same mode—and vice-versa.

While this interplay is complex, it has been simplified for many of the commonly used routines like floating-point math and string handling. FORTH words like F+ (add floating-point numbers) and STR$ (convert a double number into its ASCII string) automatically control the interplay.

With a FORTH/Assembler ROM in use—it must be present to run FORTH programs—the 71B can save FORTH values and programs. Like the environments created for subprograms, all FORTH data is saved when the

Hewlett-Packard is backing its push for third-party support for the 71B with a complete software development system that comes in two versions.

user or program switches between FORTH and BASIC, and vice-versa. The RAM-based part of FORTH, including the user dictionary of added high-level FORTH definitions (words), is held in a file called FORTHRAM. This data is maintained during power down (sleep) or when switching between languages. Only purging this file destroys the user-defined parts of FORTH.

The assembler, written in FORTH, interestingly enough, provides the complete command set used to develop the 71B's OS. It can be used to create binary files, LEX files to extend BASIC or to add new machine-code primitives to FORTH.

The text editor included in the FORTH/Assembler ROM creates general-purpose text files. These files can be source code for assembly language, BASIC or high-level definitions of FORTH words or just straight text files—for any purpose.

DOCUMENTATION

The two-volume documentation (and a quick reference guide) that comes with the 71B is a pleasant surprise. In the past, HP's documentation—for the 85 and 86, for example—was designed for step-by-step learning. A nice approach for the beginner but cumbersome for an experienced programmer. In some of the manuals, the file-handling commands were scattered about in different chapters—difficult going, at best. The only place to go for help was the brief programmer's card. Or you could go into the index and dig.

In sharp contrast are the manuals for HP accessories like the I/O ROM. Here, all keywords and operations are explained with flowcharts and examples in one or two pages.

Obviously, HP has been listening to its users. The owner's manual follows the step-by-step approach so that beginners or experts can rapidly become familiar with the 71B's operation. As well as covering the base configuration of the 71B, the owner's manual shows how to install and use the optional tape reader and optional HP-IL interface.

In the reference manual, all keywords are defined, flow-charted and explained with examples. This is a useful and brief format badly needed for advanced programmers, particularly when you have been working with other languages and need to be reminded how a specific command works. The same format is also used for the 71B's HP-IL interface.

While the manual for the 71B Development Utility doesn't go into flow-charting of the commands, its step-by-step examples should make it easy for even the beginner to transfer BASIC programs in text form between a micro and the 71B. For more advanced users and those with non-HP micros, the manual details the data-format requirements and how to customize the source code for specific hardware configurations. An appendix gives the source code for the 71B and a specific program for the IBM PC.

The FORTH/Assembler manual is brief and to the point. HP assumes that anyone doing this level of programming is experienced. It doesn't lead the beginner by the hand. The assembler documentation makes heavy reference to the IDS manuals for the 71B. Within these manuals are the assembly-language details and many sample programs, such as a Spanish-language translation LEX file of the 71B's error messages.

For advanced programming, the IDS manuals are an absolute necessity. Example material is brief, descriptive and clearly written. If you're familiar with computers, the manuals are easy to use. The IDS sections include overviews of the major functions of the 71B—for example, the file system—and then breaks them down with details of particular parts, such as the operation of BASIC's parser. The source code of the modules is well commented and easy to understand if the reader is experienced with assembly language and has learned the mnemonics of the 71B's assembler.

The 71B is built around a 4-bit CPU running at 465 kHz optimized for fast processing of binary-coded data.

The IDS manuals also contain the software hooks (entry points) to each of the software modules needed to take full advantage of the 71B's unique abilities. The combination of the entry points and detail code of each module of the system, lets the user evaluate particular modules to determine if they do the job. If not, it's easy enough to create a new module—for example, adding a special-purpose BASIC keyword—by modifying and/or combining the code of one or more modules.

SOFTWARE SUPPORT

HP is backing the 71B with a bevy of what it calls Solution Books. It's expected that the same general topics covered for the Series 41 and the 75 will be available. The books provide source code with detailed operating instruc-

tions for a variety of applications such as navigation and stress analysis. The Solution Books also have the source code in a format for loading via a bar-code wand (reader). After loading, this data can be stored on magnetic cards.

ROM-based packages are also in the works. Typical are the keyword extensions made to BASIC via Math Pac. It's expected that these packages will range from $40—the price of Applications Pacs for the 41s—to less than $300 for a complex program like VisiCalc.

EXPANDING THE HP-71B

The base 71B, priced about $550, does not include a card reader, HP-IL interface or extended memory in ROM and RAM modules. To communicate with the outside world and read in programs, most users will add the HP-IL interface and the card reader. Adding the HP-IL interface allows you to use the existing HP line of peripherals, including mass-storage devices, video display adapter ($225) and connection of the 71B to other computers and instruments. HP's Thermal Plotter/Printer ($450) and other printers with the HP-IL interface can also be hooked to the 71B. These include a strip printer and a full-page printer. The 71B's HP-IL interface is $125. The RS-232-C adapter for the HP-IL loop is $295.

The strip-card reader uses 10-inch hand-pulled magnetic cards similar to those used with the HP 75. Each card has a 1.3K-byte capacity and the card reader costs about $150. Add-on RAM modules will cost about $75 for 4K bytes. A bar-code reader compatible with the HP-IL interface will also be available.

For BASIC program development via the HP-IL and a microcomputer, you need the 71 Development Utility package. It costs $35. The package includes source code and a prerecorded program furnished on a magnetic strip card. A cassette version for use with the HP-IL mass-storage tape drive ($450) will be available from the HP User's library. For advanced programming, you need the FORTH/Assembler ROM. According to HP, it's expected to cost about $150.

HP THIRD-PARTY SUPPORT

All details of the 71B, including source code for the OS, BASIC and FORTH, and the schematics, are available. The OS design is documented in three volumes. Volume I gives a detailed description of how the software is organized and how it works. Examples, like the Spanish-translation LEX file, are included. Volume II details the entry points and the "poll" interfaces. Volume III—over three-inches thick with four reduced-size pages per sheet—gives complete source listings for all modules of the 64K OS. A hardware-specification IDS is also available.

Similarly, there are two IDS manuals for the HP-IL interface module. Volume I gives detailed design specifications and entry points for the HP-IL BASIC extensions. Volume II gives the source-code listings. In addition to the FORTH/Assembler owner's manual, IDS volumes with source listings for this ROM module will also be available.

Its on-board BASIC in ROM, with over 240 commands, is geared for scientific, engineering and statistical applications.

At present, HP's third-party support is limited to the User's Library and the IDS manuals. HP will make up special keyboard overlays and custom ROM packages in 16K, 32K, 48K and 64K. According to HP, the minimum order for custom ROMs will be at least 50 pieces. HP will also supply custom-recorded versions of the 10-inch magnetic strip cards.

ROM programmer modules or the plug-in ROM carriers will not be available as separate parts—HP will supply only the hard-programmed ROM in the carrier. However, it's a distinct possibility that the manufacturers of accessories for HP equipment will supply ROM programmers. HP says that there are ROM programmers that interface to the HP-IL loop. Quite possibly, the companies that make them will adapt the programmers to the HP ROM chips and make available carriers to plug the ROMs into the 71B. Another possibility is that new support products will appear. Programmers for the new EEPROMs (electrically erasable programmable read-only memory) are also a distinct possibility. The OS of the 71B makes provision for identifying this type of memory.

Volumes I and II of the IDS manuals cost $50 each. Volume III, the complete OS listing, costs $200. The IDS for the HP-IL sells for $75. The FORTH/Assembler manual, included with the ROM, can be bought separately for $20.

Phil Koopman is a computer designer and programmer. He is president of ECS/Software and vice president of engineering of INN Control Systems, Inc., Wood-Ridge, NJ.