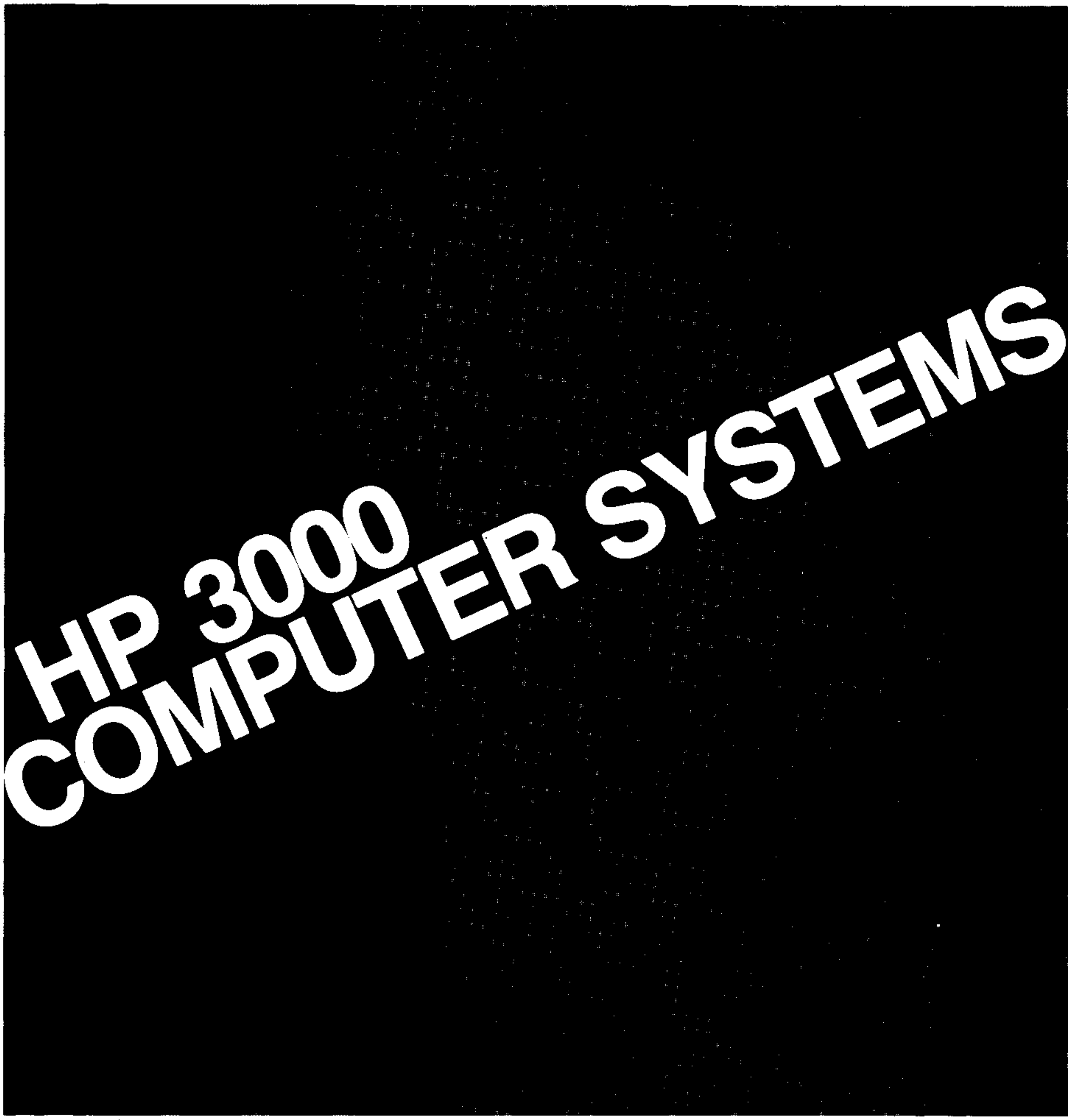
HP 3000 Computer Systems

HEWLETT hp PACKARD

Transaction processing systems
for business and industry

General Information Manual



HP 3000 Computer Systems General Information Manual

Part No. 30000-90008
5953-0550(47)
Printed in U.S.A. 6/78

HEWLETT hp PACKARD

5303 Stevens Creek Blvd., Santa Clara, California 95050

Preface

Today's business decisions cannot be based on yesterday's data. Immediate access to the most up-to-date information is a necessity for the financial planning, sales forecasting, production scheduling, and other complex tasks which are part of each business day. The HP 3000 interactive business computer systems are uniquely qualified to meet these demands for accurate, timely information. Designed specifically for terminal-oriented business data processing, the HP 3000 systems are based on an integrated hardware and software concept. With six programming languages, a complete data base management and inquiry facility, concurrent batch and on-line transaction processing, interactive program development, and an advanced distributed processing capability, the HP 3000 systems can bring information directly to the people who need it, when they need it.

All of these capabilities, which constitute the HP 3000 computer systems based on the Multiprogramming Executive III (MPE III), are introduced in this manual. The detailed specifications of the individual hardware and software products offered with the system are also presented.

Section I, the system overview, is a thorough discussion of the system architecture, operating system, processing environments, and data communications capabilities.

Section II, the reference sheets, contains the specifications of the MPE operating system; individual languages; utility programs; data entry, data base management and data communications subsystems; and the hardware peripherals.

The appendices outline the various operating system and machine level commands and intrinsics and expand on the hardware features of the computer system.

Table of Contents

System Overview

Chapter 1: System introduction and architecture

The HP 3000 computer system is a general purpose virtual memory computer with true multiprogramming and multilingual capabilities. It can simultaneously handle many transaction processing, timesharing, and batch operations in any of several programming languages. The result of an integrated hardware-software design effort, the HP 3000 is based on the specific demands of people who want to use a business data processing system in a multiprogramming environment.

The HP 3000 incorporates such advanced features as a hardware stack architecture, variable-length code segmentation in a hardware-assisted virtual memory scheme, user protection, and dynamic storage allocation. Hardware and software work together in an interrelated manner with hardware performing many of the operations conventionally performed by software, such as environment changes on interrupts or subroutine calls.

A powerful disc-based operating system, Multiprogramming Executive III (MPE III), optimizes the concurrent operation of many related or unrelated transaction processing, timesharing, or batch programs. Such system resources as main memory, the central processor, and peripheral devices are dynamically allocated to each program as needed. Typically each user is independent of all others on the system. For those cases where you need to interact and cooperate with others on joint efforts, however, the operating system provides extensive facilities for sharing information.

Major system components

Viewed from the inside out, the HP 3000 computer system consists of hardware, the MPE operating system, the HP software subsystems, and finally the user programs (see Figure 1-1).

System hardware includes the CPU, main memory, and the available peripherals as detailed in Appendix D.

The Multiprogramming Executive operating system (MPE) is the general purpose, disc-based software system that supervises the processing of programs that run on the HP 3000 computer system. MPE also provides an extensive and powerful set of system functions. The major features of the operating system are:

- **Multiprogramming: concurrent transaction processing, timesharing, and batch processing**
- **Virtual memory**
- **Stack architecture: separation of code and data, variable length segmentation, data stacks**
- **Concurrent multilingual capability: COBOL, RPG, BASIC, FORTRAN, APL, and SPL**
- **A uniform, device-independent and language-independent file system with file backup and security**
- **System security and complete accounting of resources**
- **Dynamic resource control**
- **Friendly, but powerful command language including user-defined commands, conditional job control, on-line HELP facility, and meaningful error messages**
- **Device and file independence**
- **Input/output conveniences: spooling of input and output, private disc volumes, magnetic tape labels**
- **Complete and automatic terminal management, local and remote**
- **Automatic scheduling (under the control of the installation's management)**
- **System tailoring (under the control of the installation's management)**
- **Power fail/auto restart**

Details of the MPE operating system are presented in Chapter 2.

The multiple layers of HP software subsystems illustrated in Figure 1-1 include a set of utility programs consisting of:

- **EDIT/3000—A powerful and easy to use text editor**
- **FCOPY/3000—A program for general file copying**
- **SORT/3000—A facility for ordering records in a file and merging sorted files**
- **Compiler Library—A set of subroutines providing commonly needed operations for language compilers**
- **Scientific Library—A set of subroutines providing frequently used mathematical functions**

HP 3000 systems offer six high-level programming languages: COBOL, RPG, BASIC, FORTRAN, SPL (the HP 3000 Systems Programming Language) and APL. The Data Entry Library (DEL/3000) subsystem enables you to design and maintain CRT terminal data entry screens and provide edit checking of entered data.

You also have a wide choice of data management facilities, with both sequential and random access methods included as part of the MPE file system itself. The KSAM/3000 (Keyed Sequential Access Method) subsystem extends the file system by providing files which may have one primary and up to 15 alternate keys, with retrieval based upon the value of the data. The award winning IMAGE data base management system (DBMS) was the first DBMS offered on a minicomputer in 1974. QUERY complements IMAGE by providing an English-like interactive inquiry language for an IMAGE data base.

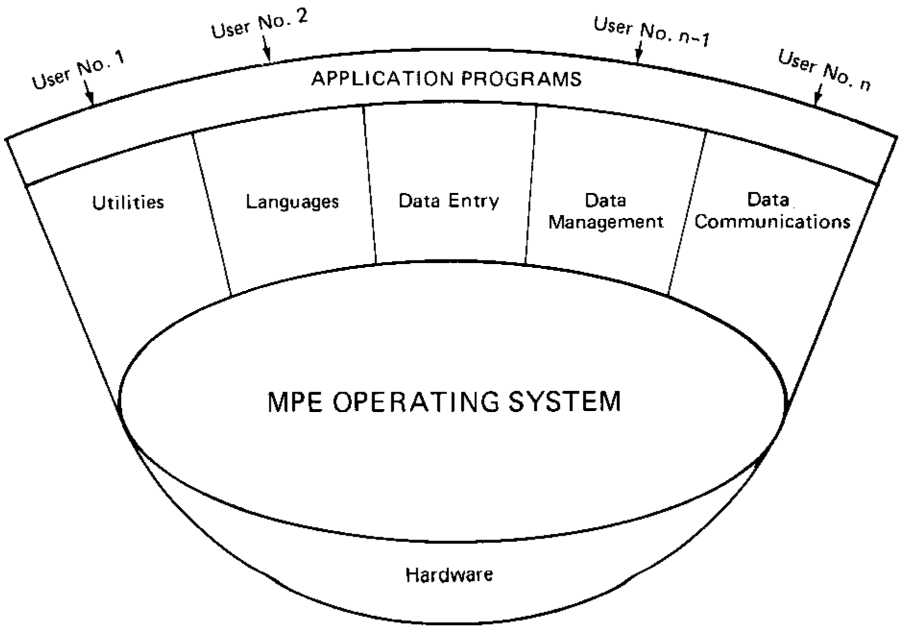


Figure 1-1. HP 3000 Computer System: Major Components

The utilities, languages, data entry and data management software subsystems are discussed in Chapter 3, Processing Environments.

Data communications subsystems extend the basic asynchronous terminal communications under MPE to include synchronous multidrop terminal communications (MTS/3000), RJE/3000 (IBM 2780/3780 emulation), MRJE/3000 (IBM HASP II and JES2 emulation); and Distributed Systems/3000 (DS/3000) for data communications between Hewlett-Packard computer systems in an HP Distributed Systems Network. Details of the data communications subsystems are found in Chapter 4.

The final element in this system is the user. You interact through the subsystems via the compilers and utilities and through MPE, usually via the file system, to ultimately utilize the hardware and the peripherals of the system. The kinds of tasks that you perform usually fall into one of the following three categories: program development, the execution of application programs, or system management (for example, building accounts).

To help you with these tasks, all of the system resources previously described are at your fingertips when you sit down at a terminal to access an HP 3000 computer system.

System features

The HP 3000 incorporates many features otherwise found only on very large computer systems. These features are summarized in the next few paragraphs and are discussed in detail in the remainder of this and subsequent chapters.

Stack architecture: The HP 3000 computer system employs a hardware stack (linear storage area for data) rather than numerous general registers for the execution of most instructions. This stack architecture provides private, hardware protected data storage for each user, as well as an automatic method for moving this data to and from the central processing unit. Major benefits of this approach include:

- Fast execution
- Code compression
- Hardware-protected execution
- Dynamic allocation of subprogram data space
- Ease of parameter passing
- Highly efficient subroutine linkage
- Rapid interruption and restoration of user environments
- Recursion (a subprogram being able to call itself)
- Reentrancy (reentering a program for another execution before some previous execution has completed)

Virtual memory: The MPE virtual memory utilizes both main-memory and an extensive storage area on magnetic disc to offer a total memory space that far exceeds the main memory size of the computer system. Programs and data are subdivided into units (segments in MPE) of code or data that are dynamically moved by MPE into main memory for execution, thus allowing programs which are larger than main memory to be compiled and executed. Also as a result of virtual memory, many very large programs may be executed concurrently with one another and the operating system.

Multiprogramming: The HP 3000's powerful operating system is designed to allow concurrent use of all system resources by more than one person. Not only can many users access the system simultaneously via terminals (either hardwired or over phone lines), but several batch jobs can be run at the same time the terminals are operating. Each individual uses the computer as if it were his own private machine. In other words, you need not depend on, or even be aware of, others using the system. The number of programs that can be processed concurrently depends on such factors as hardware configuration, program operating modes (batch or interactive), and the applications involved.

General purpose versatility: The key contribution of the HP 3000 is its exceedingly flexible yet efficient operating system. MPE simultaneously permits and controls interactive program development, batch programs, data inquiry, updates to a complete data base management system, and data communications, as illustrated in Figure 1-2.

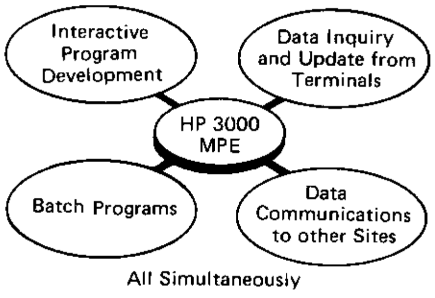

Interactive Program Development
Data Inquiry and Update from Terminals
HP 3000 MPE
Batch Programs
Data Communications to other Sites
All Simultaneously

Figure 1-2. HP 3000 MPE Operating System

Multiple language processing: Realizing that no single language is suitable for all applications, the HP 3000 was designed to give you the flexibility to choose the language(s) you require. Programs can be written in COBOL, RPG, BASIC, FORTRAN, SPL, or APL. Applications written in different languages, or a combination of languages, can be run simultaneously.

Operating simplicity: By relieving you of many program control, input/output, and other housekeeping responsibilities, the MPE operating system makes the HP 3000 extremely easy to use. MPE monitors and controls program input, compilation, run preparation, loading, execution and output. It also controls the order in which programs are executed and allocates and maintains usage records of the hardware and software resources they require.

Security: Each HP 3000 user operates in an environment protected from interference by others. System access is controlled by an account/group/user structure with optional passwords which may be added at each level. Program protection is provided by the hardware. File security is based upon a series of passwords (lockwords in MPE) and hierarchical access restrictions that allow you to specify the degree of security desired for the files you create.

Input/output conveniences: Because MPE treats all input/output devices as files (or groups of files in the case of mass storage devices), you may access these devices by file names rather than by device types or logical unit numbers. The file names specified in programs are independent of the devices used for file input and output and need only be associated with these devices at the time the programs are run. This means that you write programs without immediate concern for the physical source of input or destination of output and run them in either batch or interactive mode without changing the names of the files they reference.

Fault control memory: The HP 3000 employs high-speed semiconductor memory modules which provide automatic fault detection and single-bit correction with no loss in performance.

Concurrent I/O and CPU operations: Within the HP 3000 system many input/output operations can be performed concurrently with CPU operations. The

hardware enabling this operates under control of the MPE operating system which handles all queuing and device scheduling.

Stack Architecture

The HP 3000 with its hardware stack implementation is referred to as a stack machine. A stack is a linear storage area for data. It is so named because data items are placed on the "top," "pushing down" the data items already present. Data items are removed from the top, "popping up" those data items remaining. If you have ever worked with a Hewlett-Packard calculator that has an "ENTER" key, then you have worked with a stack. Consider the following calculation:

$$\frac{6 * 10}{2 * (3+12)}$$

Using a stack, no temporary intermediate values need to be named or stored in registers until required later in the computation. An example of how this problem would be evaluated in a stack is presented in Figure 1-3. Note that the top of stack (TOS) moves downward in keeping with the HP 3000 conventional representation. All operations can be performed without naming specific operands, as

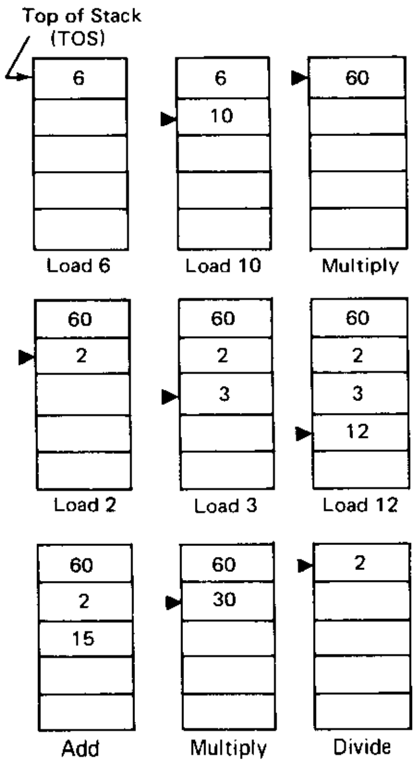


Figure 1-3.
Stack Evaluation of (6*10)÷[2*(3+12)]

the top two stack data values are implicitly used.

The benefits of a stack architecture are numerous. First, as seen above, the storage allocation is dynamic. Local storage is allocated only upon entry of a procedure and is automatically freed upon exit so that areas of memory are not tied-up and can be re-used by other parts of the program. Temporary storage of intermediate values is automatically provided. Thus compilers don't have to be concerned with saving and restoring registers for intermediate results.

Code compression is made possible by the omission of operands in many of the instructions on a stack machine. No extra registers are required for subroutine parameters and temporary variables. A register machine requires 50 to 100 percent more bits for code than a stack machine.

The subroutine is one of the most important concepts in software. Modular, structured programming has as its principle idea the partitioning of a large program into many small, understandable modules which can be called as subroutines. The best mechanism for subroutine calls and execution is the stack, because all subroutine return addresses, I/O parameters, and local variables can be pushed onto the stack. This leads to easy parameter passing and provides highly efficient subroutine linkage.

Fast execution on the HP 3000 is a benefit of having several CPU registers to hold the top part of the stack, rather than leaving it all in main memory. The HP 3000 provides each user with hardware protection of his data stack. Rapid interruption and restoration of user environments is made possible by storing the operating environment in a special block as an extension to the user's stack.

Separation of code and data

In most computer systems, programs consist of an inter-mixing of instructions and data. For example, within a subroutine there are program locations reserved by the compiler for return addresses of other subroutines and space set aside for the storage of local variables.

The HP 3000 approach separates a program into those elements that do not need to be altered and those that do. The result is that an HP 3000 program consists of a separate code area and data area (data stack), as illustrated in Figure 1-4.

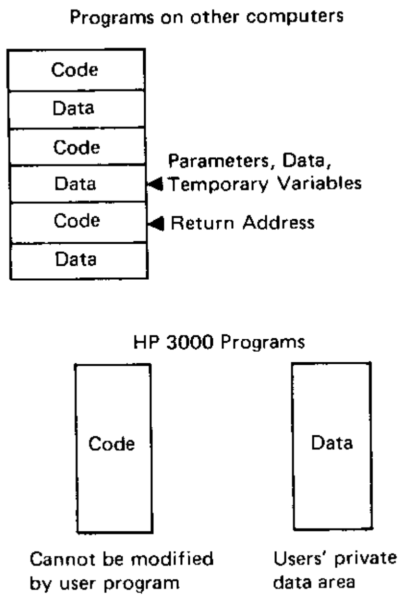

Programs on other computers
Code
Data
Code
Data
Code
Data
Parameters, Data, Temporary Variables
Return Address
HP 3000 Programs
Code
Data
Cannot be modified by user program
Users' private data area

Figure 1-4.
Separation of Code and Data

Code consists of the executable instructions that make up a program or subprogram. The values and arrays used by the program or subprogram are referred to as data. Code cannot be altered by your program as there is no instruction available in the HP 3000 instruction set to allow such a modification. However, since you, and you alone, must be able to manipulate your data, the system provides you with a unique, private, modifiable data area (data stack).

In the HP 3000, code and data are maintained in strictly separate domains and cannot be intermixed (with the exception, however, that program constants may be present in code segments). This fact, plus the fact that code is non-modifiable while active in the system, permits code to be shareable among several users. HP 3000 code is also re-entrant. Re-entrant means that when a program is interrupted during execution of a code segment and another user's execution needs the same segment, that segment can be used, is completely protected against modification, and will be returned intact to the previous user's execution. Since re-entrancy allows only one copy of heavily used programs to be shared by many users concurrently, main memory can be used with optimum efficiency. Re-entrancy and stack-structured data together make possible subprogram recursion, a subprogram calling itself, which is essential for efficient compilers and system software. Also, since code is non-modifiable, exact copies of all active code can be retained on disc, thus allowing code to be overlayed without having to first write it back out to the disc.

Processes

Programs are run on the basis of processes created and handled by the operating system. The process is not a program itself, but the unique execution of a program by a particular user at a particular time. If the same program is run by several users, or more than once by the same user, it forms part of several distinct processes.

The process is the basic executable entity. It consists of a process control block (PCB) that defines and monitors the state of the process, a dynamically-changing set of code segments, and a data area (stack) upon which these segments operate. Thus, while a program consists of data in a file and instructions not yet executable, a process is an executing program with the data stack assigned. The code segments used by a process can be shared with other processes, but its data stack is private.

For example, each user working on-line through the BASIC language is running his program under a separate process; all use the same code (the only copy of the BASIC interpreter in the system) but each has his own stack.

Processes are invisible to the programmer. In normal operation you have no control over processes or their structure, however, optional capabilities are available to permit you to create and manipulate processes directly.

Variable-length segmentation

Variable-length segmentation of code and data is used to facilitate multiprogramming. This method, in comparison with paging schemes, minimizes "checkerboard" waste of memory resources due to internal fragmentation. It also makes it possible for the operating system to deal with logical instead of physical entities. This means, for example, that a particular subprogram can always be contained within one segment rather than arbitrarily divided between two physical pages, thus minimizing the amount of swapping that needs to be done while executing that subprogram. The location and size of all executing code segments is maintained by MPE in a code segment table while the location and size of all associated data segments is maintained by MPE in a data segment table. These tables are known to both hardware and software. Software uses them for dynamic memory management by the operating system. Hardware uses them to perform references and transfers between segments and to make sure that all segments required for current execution are present in main memory. Code segments may be up to 32,760 bytes in length. Data segments may be up to 65,528 bytes in length.

Segments are stored on disc and are brought into main memory only when needed. This design results in a virtual memory environment which appears to be many times larger than the maximum size of the physical main memory. It should be noted that virtual memory in MPE does not contain code segments—only the data segments which must be swapped. The code segments stay in their original positions, anywhere on the discs.

Code segmentation

Code segmentation allows you to divide your program into several segments using control statements to the compiler at compilation time or commands to the MPE operating system. Then the operating system takes over the management of these segments. That is, MPE determines whether or not a code segment should be in main memory or in disc memory. Thus you can write programs much larger than the available main memory of your HP 3000. In the example in Figure 1-5, a program has been divided into five code segments. Code segments 1, 4 and 5 are in main memory while code segments 2 and 3 are in disc memory. The code segment table (CST), which is resident in main memory, points to the active code segments. If the code segment is in main memory, the code segment table points to the beginning of that code segment. If the code segment is on disc, the code segment table points to its disc address. The management of code segments and the code segment table is completely transparent to your program. When a subroutine call is made from one code segment to the next, the instruction that makes the call examines the code segment table to determine whether or not the new code segment is in main memory. If it is, control is transferred immediately to that code segment. If the code segment table indicates that the required segment is on disc, then the instruction interrupts the operating system and informs it that a required code segment is not in memory. It is then up to the operating system to make space for that code segment in memory for execution, and transfer it to main memory so execution of the process can continue.

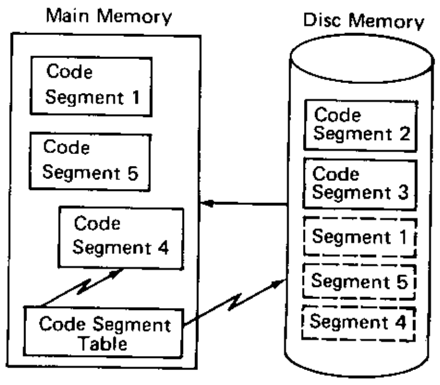


Figure 1-5.
Code Segmentation

You have control over segmentation; that is, you determine where the program is divided. The result is that segmentation can be tailored to your program's logic. The size of segments can be optimized to memory size and you can avoid thrashing (unnecessary swapping of code segments due to poor segmentation—across a DO loop, for example).

Data stack

The data area of an HP 3000 program consists of a data stack, an area of main memory that expands and contracts as the program requires. In all programs there is a certain amount of global or common data that is required throughout the life of the program. This global area represents the absolute minimum size of the data stack. Beyond this, the data area grows to meet the needs of subroutines as they allocate storage for their own working areas. When a subroutine has finished its job, this area is cut back to make use of the same memory space for the next subroutine. The end result is that much less memory is required for program execution.

Programming tasks such as the dynamic

allocation of storage are also much more straightforward than with conventional systems. For example, in many programs it is not known until execution time how much storage is required for a matrix. In a conventional system enough storage is allocated for the maximum possible size of that matrix. Therefore, a certain amount of storage is wasted because it is not required in all cases. In the HP 3000 the MPE operating system can very easily allocate only the amount required for that particular execution and that particular matrix. Figure 1-6 shows how a data stack expands and contracts during the execution of an HP 3000 program. If this same program were to be executed on a conventional register machine, the total amount of storage (that is, the global storage represented by X for one subroutine plus Y for the other subroutine) would have to be permanently allocated. This is inefficient because the local storage area of one subroutine is not required unless that subroutine is being executed. Therefore, there is no reason why this local storage area should occupy main memory. As can be seen by this example, even the maximum amount of memory required at any point in time by the HP 3000 program is generally less than the constant amount of storage area required by a conventional program.

In general, a stack is a storage area in which the last item entered is usually the first item removed. In actual use, however, programs have direct access to all elements in the stack by specifying addresses relative to several CPU registers (the DB, S, and Q registers). The stack structure provides an efficient mechanism for parameter passing, dynamic allocation of temporary storage, efficient evaluation of arithmetic expressions, and recursive subprogram calls. In addition, it enables rapid context switching to establish a new environment on subprogram calls and interrupts. In the HP 3000, all features of the stack (including the automatic transferring of data to and from the CPU registers and checking for stack overflow and stack underflow) are implemented in the hardware.

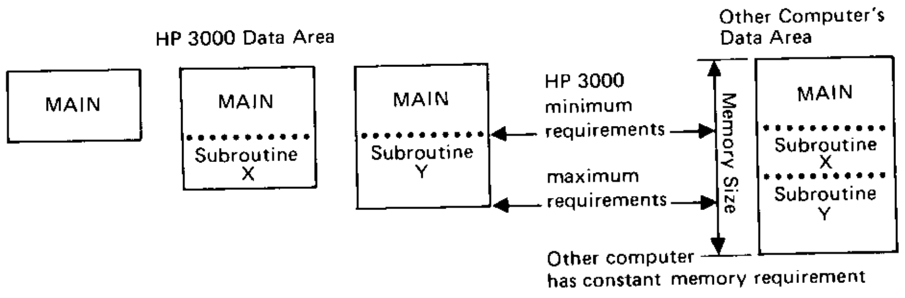


Figure 1-6.
Data Stack and Dynamic Allocation

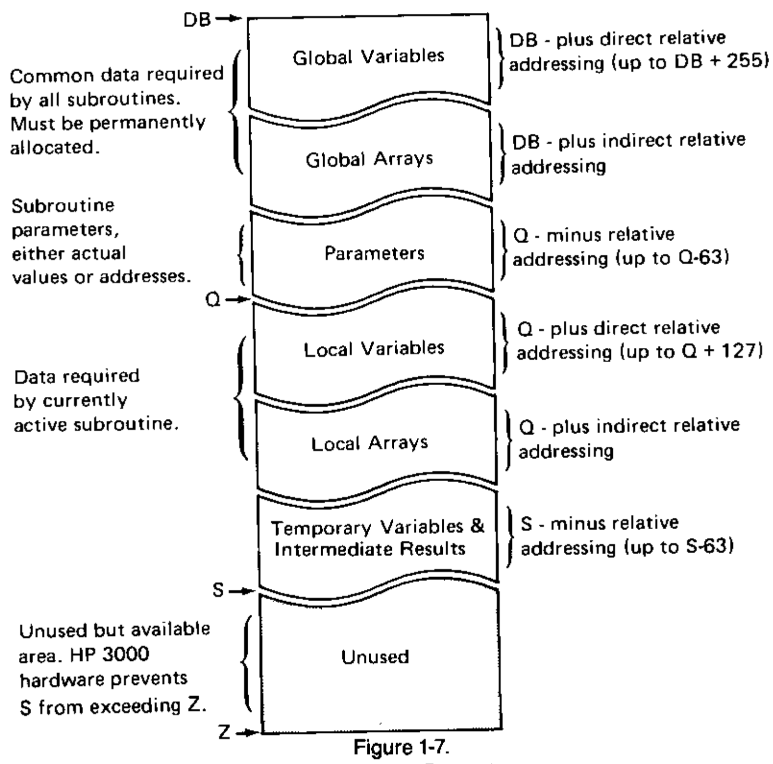

DB
Global Variables
DB - plus direct relative addressing (up to DB + 255)
Common data required by all subroutines. Must be permanently allocated.
Global Arrays
DB - plus indirect relative addressing
Subroutine parameters, either actual values or addresses.
Parameters
Q - minus relative addressing (up to Q-63)
Q
Local Variables
Q - plus direct relative addressing (up to Q + 127)
Data required by currently active subroutine.
Local Arrays
Q - plus indirect relative addressing
Temporary Variables & Intermediate Results
S - minus relative addressing (up to S-63)
S
Unused but available area. HP 3000 hardware prevents S from exceeding Z.
Unused
Z

Figure 1-7.
Data Stack Contents

When programming in a high-level language such as COBOL or RPG, all manipulation of the stack is automatically done for you by the language processor. You can, however, manipulate the stack directly by writing subprograms in SPL (Systems Programming Language for the HP 3000).

Figure 1-7 illustrates the general structure of a data stack as viewed from a subprogram. The white area represents filled locations, all containing valid data, while the shaded area represents available unfilled locations.

You can see that the contents of the data stack fall into four general areas. They are the global data required by all subroutines during the execution of a program, parameters that are passed to subroutines, the local data area required by the currently active subroutines, and the temporary storage required for the evaluation of expressions and intermediate results. The remaining area of the stack is unused and represents the amount of expansion possible in the stack without operating system intervention.

The stack area is delimited by the locations defined as DB (data base) and S (stack pointer). The addresses DB and S are retained in dedicated CPU registers. The Q-minus relative addressing area contains the parameters passed by the calling program. The area between Q and S contains the subprogram's local and temporary variables and intermediate results.

The data in the DB location is the oldest element on the stack. The data in the S location is the most current element. The location S is also referred to as the top of stack or TOS. Conventionally, the top is shown in diagrams downward from DB; this corresponds to the normal progression of writing software programs where the most recently written statement is farther down the page than previous (older) ones.

The area from S+1 to Z (the shaded area) is available for adding more elements to the stack. When a data word is added to the stack, it is stored in the next available location and the S pointer is automatically incremented by one to reflect the new TOS. This process is said to push a word onto the stack. To delete a word from the stack, the S pointer is simply decremented by one, thus putting the word in the undefined area.

To refer to recently stacked elements of data, S-minus relative addressing is used. Under this convention, S−1 is the second element on the stack, S−2 is the third, and so on. S-minus relative addressing is one of the standard addressing conventions. The others are DB-plus relative addressing and Q-minus and Q-plus relative addressing (the Q-register separates the data of a calling program or subprogram from the data of a called subprogram).

Since the top elements of the stack are the most frequently used, there are several CPU registers which may at various times contain the topmost stack elements. The use of CPU registers in this way increases the execution speed of stack operations by reducing the number of memory references needed when manipulating data at or near the top of the stack. These registers are implicitly accessed by many of the machine instructions and whenever the top stack locations are specifically referenced. Data stacks are automatically expanded by the operating system during execution up to a maximum size of 64k bytes (k = 1024).

Virtual memory

Virtual memory is a very efficient memory management scheme which, in addition to main (semiconductor) memory, uses disc storage as secondary memory. Users' program code and data are divided into variable-length segments which reside in secondary memory. As a program is being executed, only those segments of code and data which are required at a particular time actually reside in main memory; the other related segments remain on disc until they in turn are required. When a particular code segment is no longer needed, it is simply overlayed by another segment. This can be done because in the HP 3000, code segments are nonmodifiable and re-entrant. When the segment is needed again, it is simply copied from the disc on which the program resides. Since programs are copied into main memory directly from disc memory, they need not be copied prior to execution to a special "swapping disc." Data segments, however, are dynamic, and their contents can change during execution. Therefore, when a particular segment is no longer needed, it is automatically copied back to the system disc (replacing the previous version of that segment on the disc), and the main memory space of that segment is then available for other segments. The process of transferring segments between secondary memory and main memory is referred to as swapping. Whenever a segment is referenced, the hardware checks to see whether it is in main memory; if it is not, the operating system is invoked to bring it in. Thus the management of the virtual memory is totally automatic and transparent, and the system can reference a virtual memory space far larger than the real memory available.

Microprocessor

The heart of the HP 3000 hardware is the microprocessor. Each HP 3000 machine instruction is mapped into a unique microprogram that resides in control memory and consists of a series of 32-bit instructions that are executed by the microprocessor. One of the first operations in each microprogram is the execution of a NEXT instruction which begins fetching the next instruction from the HP 3000 program. Thus, when the first microinstruction is finished, the next instruction is ready for execution. This parallel execution of one instruction while the next one is being fetched, or pipelining, speeds up the overall execution time.

The primary benefit of a microprocessor is that each instruction does not require its own unique hardware logic. Instead, the instructions share the logic of a common processor called the microprocessor. This means that there is much less hardware involved in order to achieve a very sophisticated and powerful instruction set. Another advantage is that the instruction set is tailored to the compilers and the operating system. Also, the operating system's work is reduced by putting some of the more complicated operations in micro code. For instance, the operation of determining whether or not a code segment is resident in core or on disc is taken care of for the operating system by an instruction. Finally, with a microprocessor, rather than an instruction set implemented in hardware, it is easy to add new instructions. This allows Hewlett-Packard to develop new technology CPU hardware and incorporate it under MPE without changing user software.

Registers

The architecture of the HP 3000 employs a set of specific purpose registers rather than a set of general purpose registers. Each register is included in the system to efficiently perform a single specific function.

All addressing of code and data is done relative to hardware address registers. Thus by simply changing the base addresses in these registers, segments are dynamically relocatable in memory. The few instances where absolute addresses are required are privileged operations handled by the operating system.

Approximately one-half of the HP 3000 registers are accessible to user programs and/or the operating system and its related software. The remaining registers are used, for example, by the interrupt system and for microprogram processing. The registers are summarized in Appendix D.

Instructions

There are over 200 unique HP 3000 instructions which are microcoded in a read-only memory (ROM) under separate microprocessor control. Many of these instructions have multiple actions, several addressing modes, indirect addressing, and/or indexing which give a high complexity-to-instruction ratio. Code compression is achieved through the use of no-address (stack) instructions which implicitly use the contents of the stack registers as operands. All instructions except the stack operations are in 16-bit format; the stack operations may be packed two per 16-bit word to further enhance the code density.

A complete set of arithmetic instructions provides integer (16-bit two's complement), double integer (32-bit two's complement), logical (16-bit positive integer), 28-digit packed decimal (BCD coded digits packed two per byte), floating-point (32 bits including a 23-bit precision mantissa), and extended precision floating-point (64 bits including a 55-bit precision mantissa) arithmetic.

Other instructions are designed to facilitate string processing, subprogram linkage, and loop control. Certain special instructions are designated as privileged, meaning that they were designed specifically for use by the operating system. They may, however, be used by programs which the installation permits to run in privileged mode. Some of these special instructions, such as the DISP instruction for entry to the MPE dispatcher, instructions for enabling/disabling process switching, and instructions for data transfers between data segments, contribute greatly to the efficiency of the operating system.

Appendix E of this manual contains a complete list of all of the HP 3000 machine instructions.

Microcode

The entire instruction set of the HP 3000 is in the form of microcoded operations in read-only memory. This microcode is executed by a microprocessor in response to machine instructions fetched into the CPU. By allowing microprogrammed hardware to execute certain repetitive functions, such as subprogram linkage, string processing, and buffer transfers (traditionally software-implemented), the amount of code and execution times are greatly reduced.

In addition to the instruction set, many system operations that in the past were programmed in software have been microcoded. These operations are requested by machine instructions that each, in turn, execute multiple microinstructions built into the central processor hardware. Some of the standard system functions which have been microprogrammed include the interrupt handler, a cold-start loader, the saving of critical environment information on power failure, automatic restart upon restoration of power, and a set of microdiagnostics that can be invoked from the front panel of the system.

The microprogrammed instructions routinely check for addressing bounds violations during execution and automatically interrupt to error handling routines if violations occur. These memory protection checks are usually overlapped with the operand fetch and therefore do not slow execution.

Chapter 2: The operating system: MPE III

The Multiprogramming Executive, MPE III, is the general purpose, disc-based operating system that supervises the processing of user programs on HP 3000 computer systems. Designed to take advantage of HP 3000 features such as virtual memory and a stack architecture implementing separation of code and data, MPE allows multiple users to concurrently access all of the computer system resources.

The HP 3000 operating under MPE is extremely versatile, performing transaction processing, timesharing, and batch processing concurrently. Programs which support multiple terminals are both easy to create and easy to manage using MPE's full set of terminal management functions. Since the system controls device status monitoring, padding characters, buffer management, and error handling, the application program simply makes regular file read/write requests. MPE also offers you two ways to construct a transaction-oriented application: central control, in which an application program manages multiple terminals; and individual control, giving each terminal user the ability to run separate programs.

MPE relieves you of many program control, input/output, and other housekeeping responsibilities by monitoring and controlling program input, compilation, run preparation, loading, execution, and output. MPE also regulates the order in which programs are executed and allocates the hardware and software resources they require. It does this by providing an account/user/group structure, a command interpreter, a file system, a scheduling mechanism for the CPU, and a memory allocation manager.

Total system security allows you to operate in an environment protected from interference or illegal access from other users. Program protection (in memory) is provided by the hardware, with access to the system controlled through a set of passwords on account, user and group names at log-on. File security is based on a series of file passwords (called lockwords in MPE) and hierarchical access restrictions that allow you to specify the degree of security desired.

HP 3000 system resources such as main memory, the central processor, and peripheral channels and devices are dynamically allocated to each program as needed. Typically each user is independent of all others on the system. For those cases where you need to interact and cooperate with others on joint efforts, however, MPE provides facilities for sharing information.

All input/output (I/O) to physical devices (peripherals) is handled by the MPE I/O system. It receives I/O requests from other system software, queues them if necessary, and performs the transfer of data to or from the device. Because all I/O devices are treated by MPE as files, you can write programs without immediate concern for the physical source of input or destination of output and run them in either batch or interactive mode without changing the names of the files they reference.

Asynchronous terminal communications are automatically handled by MPE. Synchronous data communications to terminals, other Hewlett-Packard computer systems (including other HP 3000s), and non-HP computers is provided through optional software subsystems which extend the MPE data communications capabilities.

MPE provides many tools for overall management and control of the system, including a power fail/automatic restart routine. In addition, system managers and system supervisors can access an accounting facility, logging facility, and SYSDUMP program which is used to back up the system software.

Figure 2-1 illustrates the major components of the Multiprogramming Executive III operating system.

Multiprogramming

One of the major ways in which operating efficiency is achieved in an HP 3000 is by multiprogramming. Multiprogramming is a method whereby several related or unrelated programs are in partial states of completion and resources are being allocated among them to balance the system load and to meet response and throughput requirements. While one program is awaiting an occurrence, such as the completion of an input/output operation, control of the central processor is directed to the program which is currently highest in priority of those waiting for the CPU. The controlled competition among several programs for the processing, storage, input/output, and programming facilities of the system helps to ensure that as much of the system as possible is always kept busy performing useful work. As a result, the

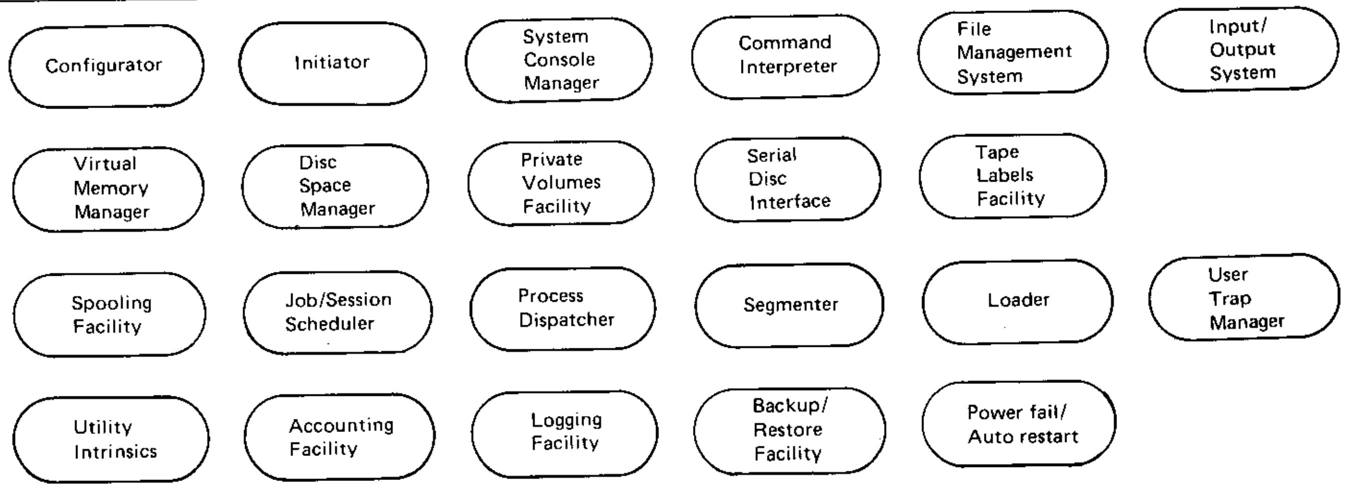


Figure 2-1. HP 3000 MPE III Operating System

total throughput of the system (that is, the total volume of work performed by the system during a given interval of time) is significantly increased.

MPE is designed to efficiently and fairly allocate, schedule, and dispatch control of the central processor, storage, and input/output devices among many programs being executed concurrently. It operates in conjunction with the architecture of the central processor to provide complete protection against one program destroying or interfering with another.

An example of how MPE works together with the instruction set to facilitate the multiprogramming capability is seen in the system's memory management scheme. The code segment table that is permanently resident in main memory contains status information on the frequency of use of code segments. MPE periodically references this status information to determine which code segments are the most often used. These most often used code segments will be left in main memory while the ones that are seldom used are candidates to be overlayed by new code segments. Also, MPE recognizes the fact that code segments may be written over without saving them, because code is not modifiable. However, if it is necessary to use the space occupied by a data segment, that segment must first be written onto disc. This means that MPE first tries to provide new memory by overlaying code segments rather than by using the space occupied by a data segment.

The number of programs that can be processed concurrently depends on such factors as the hardware configuration, program operating modes (interactive or batch), and the applications involved. MPE is designed so that the maximum number of concurrently running programs can be expanded or contracted by changing a single system configuration parameter.

MPE also allows the concurrent execution of programs from two types of input media, traditional batch input devices and interactive terminals, since all programs are independent of their input mode. In fact, the same system code is used to accomplish particular functions in either mode, resulting in storage economy and reduced overhead.

Interactive processing
In interactive processing you enter commands and data through a keyboard terminal and receive immediate responses to your input. This type of interaction, called a session, is particularly useful for program development, text editing, data entry, information retrieval, computer-assisted instruction, and many more applications where a direct dialogue with the system can help solve a problem. The operating system itself (including access to spooled devices) and all of its language, utility, data base management, and telecommunications programs are accessible by way of sessions.

A session begins when MPE recognizes and accepts a HELLO command from an on-line terminal. After you are connected to the MPE command interpreter and after a successful log-on sequence, you may enter commands to use compilers, enter subsystems such as the text editor, run programs, or modify your file domain. The session exists until you type the BYE command, the operator forcibly aborts the session, or a second HELLO command or a JOB or DATA command is encountered.

As an example, consider the case where you might wish to create a COBOL program and then compile, prepare, and run it. This can easily be done by way of a session. Refer to Figure 2-2 which shows the various commands entered during the session. You initiate the session by pressing the RETURN key on a terminal that is connected on-line to the system. MPE responds by displaying a prompt character (a colon). You then log on to MPE by entering a HELLO command containing at least your assigned user and account names. Then you call the HP 3000 text editor and enter two editor commands followed by the COBOL statements that constitute your program. When the entire source program has been entered, you save it on disc under the file name YOURFILE by entering a KEEP editor command and then terminate the editor subsystem. Now your source program exists as a disc file in the system. To compile, prepare, and execute the program you simply enter a COBOLGO command specifying the file name YOURFILE. This one command first invokes the COBOL compiler which compiles the program, then invokes the MPE segmenter which prepares the compiled program into an executable form, and finally executes the prepared program. When the program has finished executing, MPE displays the message END OF PROGRAM followed by another colon prompt character. You then can terminate the session by entering the BYE command.

The above example is somewhat simplified since it does not include the various informational messages, compilation output, and program output generated by MPE, the text editor, the COBOL compiler, and the program itself. The fact remains, however, that if the source program (entered by way of the editor) contains no errors, the entire session can be performed by entering just eight MPE and text editor commands in addition to the COBOL statements which constitute the program.

Batch processing

Batch processing lets you submit to the computer, as a single unit, commands that request various MPE operations such as program compilation and execution, file manipulation, or utility functions. Such a unit is called a job. Jobs contain all necessary instructions to MPE and all references to programs and data required for their execution. Once a job is running, you need to supply no further information.

Jobs are often read through batch input devices such as card readers or tape units. A unique feature of MPE, however, allows you to enter batch job streams through your terminal during the course of an interactive session.

Several jobs can be submitted to the system from multiple devices concurrently. MPE schedules each job for execution according to its job input priority. Commands are provided for monitoring job selection, and the operator can dynamically adjust the job selection criteria.

When a job enters execution, the commands within it are sequentially executed on a multiprogramming basis. MPE generates the job output on a local device such as a line printer, tape unit, or disc unit, or on a local or remote terminal. When one job is temporarily suspended, perhaps to await input of data, another (if available) immediately enters execution.

MPE executes many sessions and batch jobs simultaneously. The only significant difference between a session and a batch job is that during a session you can interactively alter the course of processing, whereas in a job the command stream is fixed and the job is executed in its entirety as pre-defined in the job control statements without active user intervention.

On the HP 3000, batch processing is a logical extension of the interactive functions available. Any capability (except BREAK) available in one mode is available in the other, using all the same MPE commands. The standard input and output devices ($STDIN, $STDLIST) are automatically redefined from the terminal to the card reader and line printer. All languages, utilities, and applications development software (such as IMAGE/QUERY) can be run in batch mode with no changes.

MPE users

All users of the MPE operating system are organized into unique accounts. The system manager controls these accounts by creating them, deleting them, and changing their characteristics. Each account has associated with it a list of users and a list of groups. Users are individuals who are given the capability of accessing the HP 3000 system under a particular account name. Groups are used to partition the set of files belonging to an account. It is the responsibility of the account manager to create, delete, or change the characteristics of the users and groups in his account.

The account: Accounts are defined by the user with system manager capability. Each account has a unique name and an optional password which must be used to access the HP 3000 on behalf of this account. MPE stores a list of user names and group names recognized by this account, a maximum job priority (the highest priority at which any job in this account may be scheduled), limits on the account's usage of disc file space, CPU time, and connect time. Running counts of each resource that the account has actually used are also maintained.

```
: HELLO YOURNAME.YOURACCT ← Initiates the session.

: EDITOR ← Invokes the HP 3000 text editor.

/ SET FORMAT=COBOL ← Specifies that you will be entering COBOL source statements.

/ ADD ← Specifies that you wish to create a new text file.

       (COBOL source statements)

/ KEEP YOURFILE ← Saves the text file on disc under the name YOURFILE.

/ EXIT ← Terminates the text editor.

                   Causes the COBOL source program contained in YOURFILE
: COBOLGO YOURFILE ← to be compiled, prepared, and executed.

: BYE ← Terminates the session.
↑
└Prompt characters issued by MPE and the text editor.
```

Figure 2-2. Sample Session

The user: Once an account has been defined by the system manager, a user with account manager capability identifies valid users of that account by name and, optionally, by password. In addition, each user may have a specified home group which is assigned when he does not specify a group while logging onto MPE. Each user also has a maximum job priority which may not be greater than that of the account he belongs to.

The group: Groups are used to partition the file domain of an account. Each account "owns" a unique set of files separate and distinct from every other account. This ownership of files is indirect in the sense that only groups may own files directly. Therefore, every file belongs to a group, every group belongs to an account, and every account belongs to the system. Since each group defines a private file domain, file names must be unique only within a particular group.

Groups are also specified by the account manager. Each group within an account has a unique name and, optionally, a password. The account manager may also define limits on permanent disc file space, CPU time, and connect time used by a particular group. MPE maintains running counts of resource usage for each group within an account. The sum of these group counts always equals that of the account in total, and cannot exceed the account limit.

To illustrate how accounts, groups, and users interrelate, consider the following example of a system which includes interactive terminals dispersed throughout a company. As shown in Figure 2-3, the user with system manager capability has assigned six accounts: Marketing, Production, Engineering, Finance, Personnel, and Quality Assurance. For the Marketing Department account (detailed in Figure 2-3), a user with account manager capability has defined three users who can access the system. Each user has a private group (assigned as his home group) where he stores his private programs and files. In addition, the marketing users can access programs and files in the other groups in the account. Two groups were created to contain programs and data files for two current projects. Another group can be accessed for work on schedules, budgets and other administrative tasks. The marketing users can also access a public group (with no password required at log-on time) for storage of general purpose programs (utilities) used by all.

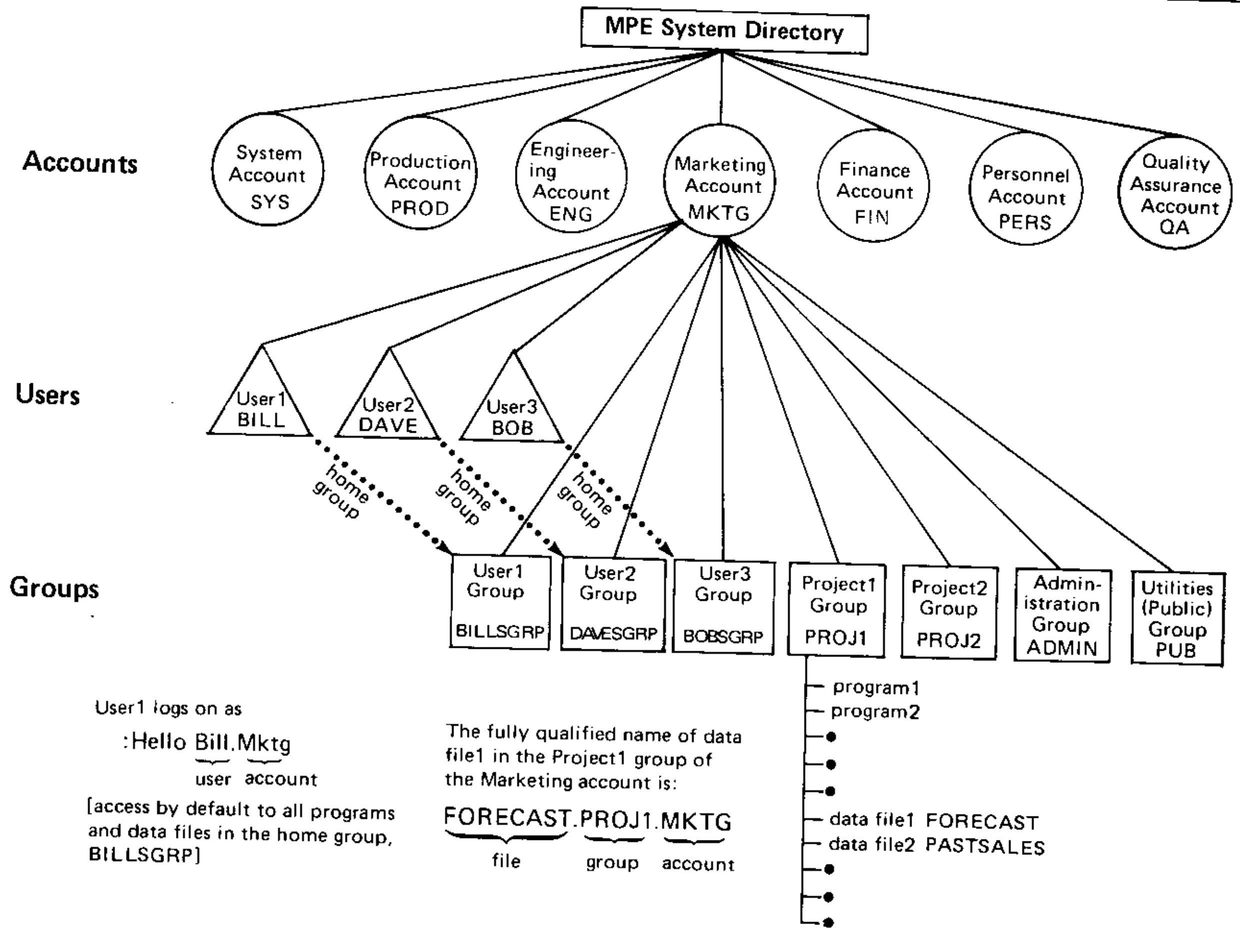


Figure 2-3. MPE Account/Group/User Structure

User 1 (Bill) in the Marketing account (MKTG) could log on to the HP 3000 from a terminal with the following command:

```
:HELLO  BILL.MKTG
 \       \    \
MPE prompt  User  Account
```

By default, User 1 would then have access to all programs and data files in his homegroup, BILLSGRP. Access to programs or files in another group, for example PROJ1, can be gained by using the fully qualified file name which specifies both the account and group under which it was created. To access data in the data file 1 (FORECAST) in the Project 1 group (PROJ1) of the Marketing account (MKTG), the user would describe it as:

```
FORECAST.PROJ1.MKTG
  /       /     \
File    Group   Account
```

Alternatively, User 1 could log on to the HP 3000 and request access to all programs and files in the Project 1 group by appending the group name to his log-on, as follows:

```
:HELLO  BILL.MKTG,PROJ1
 \       \    \    \
MPE Prompt  User  Account  Group
                          [optional]
```

Now he has direct access to all files in the PROJ1 group.

To summarize, you can log on to the system using only your user and account names, in which case you are automatically placed in your homegroup. Or, you can log on specifying your user name, account name, and a group name (either your homegroup or any other to which you have access). To access files outside your log-on group, you must use the fully qualified file name consisting of the name of the file, the group and the account. To access files within your log-on group, you need only specify the file name alone.

Security provisions relating to system access and file access are described in later sections. Total security is provided by MPE so that each user operates in an environment protected from interference or illegal access from others.

User capability sets: A typical HP 3000 installation has a large variety of users. They range from a user who simply wants to run a compiled COBOL program to perform data entry to a systems programmer who wants to take advantage of all of the operating system features. Therefore, a set of capabilities is assigned to each account and to users within that account, so that the user with system manager capability can maintain control over the system, assigning special capabilities only to those people who need them and understand how to use them correctly.

The total capability set is divided into three categories: 1) user attributes, 2) file access attributes, and 3) capability-class attributes for access to MPE resources. The user attributes include system manager for overall system management; system supervisor for day to day, operational control of the system; account manager for overall account management (users and groups); account librarian with special file access capabilities for maintenance of account files; group librarian with special file access capabilities for maintenance of group files; and diagnostician, with the ability to run diagnostic programs under MPE for on-line checkout of HP 3000 hardware components.

There are two file access attributes—save, which permits the user to save files by declaring them permanent; and non-shareable device, which allows a user to acquire non-shareable devices, such as a line printer.

The final user capability set, capability-class attributes, refers to the ability to access special MPE facilities. Included are interactive access, local batch access, process handling, extra data segment acquisition, multiple resource identity numbers, privileged mode, and data communications.

The majority of users in a typical HP 3000 installation will simply have classes for interactive access and local batch access. MPE simplifies the creation of new accounts, users, and groups by establishing a set of default capabilities for each. If a user later needs additional capabilities, these can be easily added.

The presence of capability sets greatly simplifies the use of the system from the standpoint of each individual user by defining the extent to which he must understand MPE and permitting him to ignore those aspects of the system that do not apply to him.

File system

The general purpose of any computer system is to input, process, and output information. Under MPE, this information may be created and used by the operating system itself, by compilers or other systems, by user programs, or by users themselves. To handle all information in a uniform, efficient way, MPE treats it as groups of data called files. Specifically, a file is a collection of information or data identified by a name recognized by MPE. It may be helpful to think of a file in the traditional way that people in business and commerce have for many years—as a filing cabinet containing information of various kinds. Instead of a filing cabinet, of course, MPE uses media such as disc, cards, and tape for storing the information. On any of these media, a file may contain MPE commands, system or user programs, or data—alone or in any combination.

All user access to input/output devices is by way of the device-independent MPE file system. Device-independence means that any program can read data from a card reader, terminal, magnetic tape, or disc, for example, without changing the program. This allows you to develop application programs without concern for the type of input or output device it will ultimately use. Whether the device is a card reader, terminal, magnetic tape or a disc, the program is the same.

All location of data, buffering, data transfers, and deblocking are handled automatically by MPE. When you ask to read a named disc file, you are only implicitly specifying the actual disc address of the file; the MPE file system determines the explicit address and performs the read. At another level, when you ask the file system for a certain type of device by specifying a device class name (e.g. disc, line printer, etc.), the file system takes care of allocating an actual device.

The file system permits sequential access to all files. Disc files with fixed-length records may also be accessed randomly.

Files can be accessed from any of the available programming languages. For the high level languages, you must follow the conventions of the language, not MPE. The particular compiler invokes MPE file system intrinsics as implied by the language constructs. Many programs can simultaneously access the same file, and MPE automatically resolves any contention problems which might arise. A single program can access over one hundred files concurrently, if necessary.

File commands allow programs to reference files without specific knowledge of their actual names or characteristics. You can redefine a file without changing (or recompiling) a program, thus eliminating reprogramming and unnecessary program logic. For example, a program's input file, CARDIN, could be designed to be a card reader, but later changed to a disc file through the use of a FILE command. File specifications can also be altered at run-time by using file commands. To illustrate, a program could be coded to open a file assuming its record length is 128 characters. If at run time it is determined that this file only has 64 characters, you can override the file opening with a FILE command that designates the file to be 64 characters. Reprogramming is thus eliminated.

Commands are provided to BUILD, PURGE, and RENAME disc files. You may specify how disc space is to be dynamically acquired, whether to deal with logical or physical records, whether to include carriage control characters, and so forth. The RENAME command, which is restricted to the user who created the file, can also establish or change a lockword on a file or move a file to another group in the account.

The names and locations of all files in the system are maintained in a system file directory. The LISTF command can be used to display varying amounts of file directory information about an individual disc file, all files in a particular group, or all files in a particular account.

A temporary disc file set is also available to each job and session. New disc files may be placed in the temporary file set programmatically or by using a FILE command. If a temporary disc file is not saved into the account's permanent disc file set, it will be deleted at the end of the job or session. Temporary file space is not included in the account or group file space totals or limits.

Extensive disc file back-up facilities are provided for all types of disc files. The STORE command copies a file, a file set, or several file sets to a serial storage device (magnetic tape or a serial disc). The RESTORE command retrieves the files from a serial storage device.

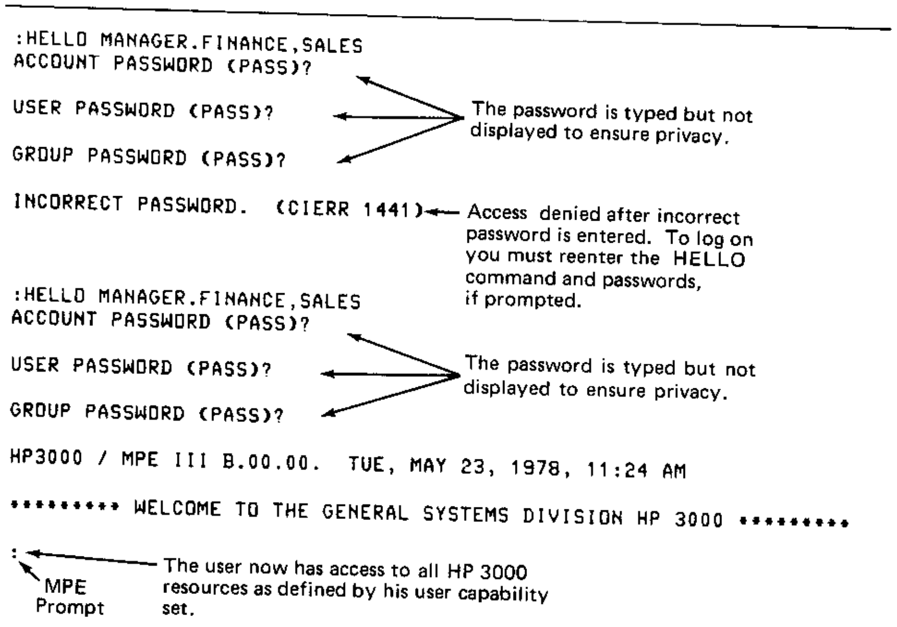
The file system itself is actually a collection of procedures (subroutines) which reside in the system segmented library or SL. They allow you to open a file, obtain status information, read or write data, perform control functions, and close the file. When a program contains any statements or constructions that input or output data, file system routines are required. The loading operations done by MPE to run the user program search the library and establish linkages to allow these SL routines to be referenced during program execution. The code segments containing these file system procedures are shareable and may be used by several programs at the same time.

A full set of intrinsics (procedures) are also available to programatically access, manipulate, and obtain information on files. The MPE file system includes the most widely used set of system intrinsics on the HP 3000.

Security

The MPE operating system provides each HP 3000 installation with a very flexible security system which allows you to operate in the multi-user environment without interference or illegal access from other users.

System access: MPE's multi-level system security is based upon an organizational structure of accounts, groups, and files. An account consists of one or more groups, and each group contains actual data or program files. These files may be shared by anyone who can successfully log on to the system and pass additional file security constraints. Access to the system is granted only to individuals with a valid log-on identification consisting of account, group and user name, each of which may require a password. Figure 2-4 illustrates both an unsuccessful and successful log-on procedure.



```
:HELLO MANAGER.FINANCE,SALES
ACCOUNT PASSWORD (PASS)?

USER PASSWORD (PASS)?

GROUP PASSWORD (PASS)?

INCORRECT PASSWORD.  (CIERR 1441)


:HELLO MANAGER.FINANCE,SALES
ACCOUNT PASSWORD (PASS)?

USER PASSWORD (PASS)?

GROUP PASSWORD (PASS)?

HP3000 / MPE III B.00.00.  TUE, MAY 23, 1978, 11:24 AM

********* WELCOME TO THE GENERAL SYSTEMS DIVISION HP 3000 *********

:
```

The password is typed but not displayed to ensure privacy.

Access denied after incorrect password is entered. To log on you must reenter the HELLO command and passwords, if prompted.

The password is typed but not displayed to ensure privacy.

The user now has access to all HP 3000 resources as defined by his user capability set.

Figure 2-4. MPE System Access Security

File security: MPE provides two general methods of file security. The first is the use of passwords. The creator of a file can establish a password (referred to in the MPE documentation as a lockword) which must be correctly supplied whenever anyone makes reference to that file.

The second method of file security is the use of file access mode and user type restrictions, as outlined in Table 2-1. For each account he creates, the system manager specifies the file access modes allowed under that account and the types of users to whom they are available. Similarly, when an account manager creates a group he specifies the file access modes allowed under that group and the types of users to whom they are available. Finally, the creator of a particular file specifies the file access modes allowed for that file and the types of users to whom the file is available. Thus, the total security provisions for any file depend upon specifications made at all three levels—the account, group, and file levels. You must pass tests at all three levels to successfully access a file in the requested mode.

The file system security and account/group/user structure provide many classes of security for user files. Access to files may be controlled at several levels which range from unrestricted access by anyone to controlled access available only to the creator of the file. For example, you could make your data file available to any other user in a "read-only" mode, while only members of your immediate account could append data to the file.

Many times a need exists to save general purpose or utility programs in public groups or accounts which may be accessed by all users system wide or all the users within an account. MPE allows for this by defining a special system account named "SYS" and a public group named "PUB" which may exist under any account, with less-restrictive default security provisions than other accounts and groups.

User interface

The simplicity of the MPE command language greatly enhances the system's usability. You interface with MPE through commands (for general functions external to your programs) and intrinsic calls (for specific functions invoked during program execution). Common system commands are used to initiate and terminate jobs and sessions, re-specify file characteristics, compile and execute programs, and call various utility subsystems. You need to learn only one set of conventions for using these facilities because they all use the same command formats, special characters, and error-diagnostic methods. Intrinsic calls implement such functions as reading, writing to, and updating files; skipping forward or backward in a file; or returning system table information to your program. They are available not only to SPL (Systems Programming Language) but also to the higher level languages, COBOL, RPG, FORTRAN, BASIC, and APL. This combination of intrinsics and commands creates two levels of access to MPE, providing you with a very powerful mechanism for the implementation of a wide variety of applications.

MPE commands: You can perform many system functions by using MPE commands. When a session or job is being executed, a portion of MPE called the command interpreter reads each command image from the terminal or the batch input device (or spooled disc file), checks its validity, and then causes the appropriate action to take place. After the requested action is successfully completed, control always returns to the command interpreter which then processes the next command. In addition, the command interpreter may be called programmatically so that certain system commands may be executed from a running program.

You can interrupt execution of a program at any point in a session by pressing the BREAK key on your terminal (some terminals have an ATTENTION or INTERRUPT key instead of the BREAK key). This returns control to the MPE command interpreter, allowing you to issue commands to abort the program (without disrupting the session), perform various file or utility operations, or resume program execution. The BREAK key will also stop some MPE commands, such as STORE, RESTORE, and LISTF during execution.

A colon (:) is the MPE prompt character. When submitting a batch job all MPE commands within that job have a colon in column one; this helps MPE to distinguish the statement as a system command and also aids in detecting the end of data within a batch job stream. During a session, however, MPE automatically displays the colon at your terminal to alert you that MPE is ready to accept another command.

There are MPE commands for:

- HELP
- Initiating and terminating jobs or sessions
- Running system programs (such as compilers)
- Running user programs
- Creating, managing or deleting files
- Displaying file information
- Displaying job and session status

Table 2-1. MPE File Access Modes and User Types

	Access modes
Reading	Allows you to read files.
Appending	Allows you to add information and disc extents to existing files.
Writing	Allows you to delete or change information already present in existing files.
Executing	Allows you to run programs stored in existing files.
Locking	Allows you exclusive access to a file if you desire it (if requested in the file reference, no other user can access the file concurrently).
Save Files	Allows you to save permanent disc files within your group.

	User types
Any User	Makes the specified access modes available to any user of the system.
Account Member	Restricts the specified access modes only to users of the account in which the file resides.
Account Librarian	Restricts the specified access modes only to the account librarian of the account in which the file resides.
Group User	Restricts the specified access modes only to users of the group in which the file resides.
Group Librarian	Restricts the specified access modes only to the group librarian of the group in which the file resides.
Creator	Restricts the specified access modes only to the creator of the file.

- Displaying device status
- Transmitting messages
- Placing restrictions on resource consumption
- Debugging
- Accounting of system resource consumption
- System tailoring, documenting, and back-up
- System tuning
- System console operation (job control, device control)
- Spooling
- Logging
- Opening or closing communications lines
- Establishing communication between a local computer and a remote computer

Appendix B of this manual gives a complete list of all MPE commands with a brief description of each.

Command error messages: If a command error is detected during a session, MPE responds by displaying an error message and specifying which parameter is in error. MPE then prompts you for another command and you may continue. In a session erroneous commands can be interactively edited and reissued with the REDO (edit) command. If a command error is detected during a batch job, the error number and a descriptive message are printed on the job listing device, and all card images from that point through the next EOJ, DATA, or JOB command are usually ignored. The CONTINUE command can, however, be used to cause execution to continue despite a command error. Conditional job execution is possible using job control words, discussed in a later section. MPE fetches the command error messages from a catalog (CATALOG.PUB.SYS). You can change this catalog to build your own error messages (custom messages, translations, etc.) through an HP 3000 utility program.

On-line HELP facility: The MPE on-line HELP facility provides you with graduated, increasingly detailed information on any MPE command or set of commands, assuming you have a basic familiarity with the command language. If you have forgotten some detail of syntax or semantics, or even the name of a particular command, you can get concise descriptions of the commands on line at your terminal. The HELP messages displayed coincide with the information contained in the HP 3000 MPE user manuals. Figure 2-5 illustrates the two ways you can use the HELP facility. In the "immediate" mode, you merely enter the HELP command followed by a parameter. Information detailing that parameter is displayed immediately. In the HELP "subsystem" mode, you enter the HELP command alone (with no parameters). The system then displays a menu of parameters for you to choose from, prompts you with a "greater than" (>) sign, and waits for your input. When you enter your parameter selection the system responds with the information you requested.

User-defined commands: MPE allows you to define your own commands to fit your specific needs by combining several MPE commands into a command procedure and assigning it a name. You can then use this procedure name as a command. Thus, it is possible to enter one command name (defined by you) and cause several MPE commands to be executed. Such a command, once defined, may be entered by any user. It is also possible to redefine existing MPE commands; for example, you could redefine the COBOL command as a null command to disallow its use. This user-defined commands facility allows more

```
:HELP
Information is available on the following classes of commands:
  Running Sessions
  Running Jobs
  Managing Files
  Running Subsystems and Programs
  System Management, Status, and Accounting
  Utility Functions

For more information, enter a KEYWORD.  You can also enter any command
name as a keyword.  Enter 'help' for information on help.  Enter 'exit'
to leave help.
KEYWORDS: SESSIONS,JOBS,PROGRAMS,FILES,MANAGE,UTILITY
>SESSIONS
Running Sessions.  Following are the commands used:
   ( ) COMMAND LOG ON
   ABORT
   BYE
   DSLINE
   EOD
   EOF
   HELLO
   HELP
   REMOTE HELLO
   RESUME

You can use any command as a keyword.
KEYWORDS: SESSIONS,JOBS,PROGRAMS,FILES,MANAGE,UTILITY
>EXIT
:      You EXIT the HELP Subsystem
  MPE Prompt


:HELP REMOTE,EXAMPLE

EXAMPLE

To establish the communications line HDS2 and log on to System B from
System A, you could enter:

    :HELLO USER.X
    :REMOTE HELLO USER.X;DSLINE=HDS2
KEYWORDS: PARMS,OPERATION,EXAMPLE
:  MPE Prompt.  You are given a new prompt automatically
   following the display of information
```

Figure 2-5. Using the On-line HELP Facility

experienced programmers and designers to create and control the command language environment for other, less experienced users of the HP 3000. It also makes it easy and inexpensive to extend and tailor the MPE command set to fit specific needs.

As an example, it is possible for a clerk to perform a data entry program using only the HELLO command at log-on, if you have previously created a user-defined command (UDC) which is automatically executed at log on. This UDC could cause a menu or other information to be displayed at the terminal to prompt the clerk for input. When the last item has been satisfied, the UDC could generate an automatic log-off.

When the clerk logs on as:

```
:HELLO SANDY.ACCTNG
```

the following commands could be automatically executed in sequence, as defined by a UDC:

```
:RUN DATAIN
:BYE
```

Here it is assumed that the data entry program, DATAIN, displays a menu or some other information to prompt the clerk to enter the required input data.

Job control: MPE contains two special facilities, job control words (JCWs) and conditional execution functions, which allow you to design job streams whose execution can be dynamically altered based on the results of previous job steps.

Both system and user-defined job control words are used to store job status information and to pass such information between programs and between a program and the command interpreter. Defined and accessed by commands from the command interpreter and by intrinsics from user programs, JCWs can be used in conjunction with the IF, ELSE, and ENDIF commands of a conditional IF block to alter the path of job execution based on the current JCW value. The IF command contains a logical expression (which may involve a JCW). This expression is evaluated at execution. If its value is true, the remaining IF block statements are executed. If its value is false, any existing ELSE block statements are executed instead.

The following example illustrates the use of JCWs and conditional IF blocks.

```
:RUN P108X1                 (Edit and verify transaction cards)
:RUN P108X2                 (Count valid transactions)
:IF (JCW < FATAL) THEN      (If no fatal errors, schedule shipments)
:  IF (JCW < 5000) THEN     (Number of shipments to schedule)
:     RUN P108X3            (Schedule low priority shipments)
:  ENDIF
:  RUN P108X4               (Schedule high priority shipments)
:ELSE
:  RUN P108X5               (Produce error report and fix JCW)
:ENDIF
:RUN P108X6                 (Produce final report)
```

The sample job runs a program to edit and verify transaction cards and count valid transactions. If no fatal errors are encountered, the job schedules shipments (either all shipments or only high priority shipments depending on the value of JCW) and produces a final report. If fatal errors do occur, the job does no shipment scheduling. Instead, it produces only an error report and a final report.

MPE intrinsics: Most system functions are also available in the form of special MPE procedures, called intrinsics, which may be invoked by external calls from your program. These external links from user programs to operating system intrinsics are satisfied when the MPE segmenter prepares the program for execution. You may also establish your own "intrinsics" within the system.

System intrinsics are written in the HP 3000 Systems Programming Language (SPL) and follow the rules and constraints of that language. They may be called from BASIC, FORTRAN, or SPL programs, and from COBOL and RPG programs by way of an SPL subroutine.

There are MPE intrinsics for:

- Opening and closing files
- Reading from, writing to, and managing files
- Controlling devices (such as rewinding magnetic tapes)
- Obtaining file information
- Obtaining user information
- Obtaining detailed error information
- Performing data translation
- Obtaining date and time
- Process handling
- Resource handling
- Data segment handling

Appendix C of this manual gives a complete list of all MPE intrinsics with a brief description of each.

When a system intrinsic is invoked to perform a system function, two types of error conditions may occur. MPE informs the calling program of a recoverable error by setting the condition code bits of the HP 3000 status register when the intrinsic is exited. The condition code indicates whether or not the request was granted and what conditions existed pertinent to the request. A request to an intrinsic which requires a special capability class not possessed by the calling program, or which passes illegal parameters to an intrinsic, is considered an irrecoverable error and causes the system to abort the program. In such a case, if you have not specified an appropriate "trap procedure," a batch job is usually removed from the system; an interactive session resumes control at your terminal with a message and a prompt for another command. The CONTINUE command can, however, be used to continue execution of a batch job despite an irrecoverable intrinsic error. Also, by initially calling system trap intrinsics, you may specify special actions to be taken in the event of an irrecoverable error.

Input/output

A major objective of MPE is to simplify as much as possible the software overhead in dealing with physical devices. This is accomplished through a set of MPE system intrinsics and commands which form the file system. The file system allows user programs to perform I/O by interacting with logical "files" rather than physical devices. Since physical devices such as a line printer typically have complex and inconsistent software interfaces, the MPE file system isolates you from them by transforming your logical file request into the actual commands and device addresses necessary to interface to a particular I/O device. Within a program, files are addressed by name and accessed through a variety of intrinsics available in the file system. These intrinsics may be called directly from any language or may be made invisible by the file capabilities built into standard programming languages such as COBOL, RPG, BASIC, FORTRAN, APL, and SPL. In addition, there are various file management commands that are provided for file specification at execution time.

Input/output conveniences: Because MPE treats all input/output devices as files (or groups of files in the case of mass storage devices), you may access these devices by file names rather than by device types or logical unit numbers. The file names used in programs are independent of the devices used for file input and output, and need only be associated with these devices at the time the programs are run. This means that you can write programs without immediate concern for the physical source of input or destination of output and run them in either batch or interactive mode without changing the names of the files they reference.

Files on disc can be structured for either direct or sequential access, on an exclusive or shared basis. Direct-access files contain fixed-length records; sequential files, however, can contain fixed-length, variable-length, or undefined-length records. For files on disc, storage space is automatically allocated as it is needed. MPE permits simultaneous access of shareable disc files by many users.

Device specification: When you specify a "device" to the file system, you may use either a device class name or a logical device number. The device class name is related at system generation time to one or more logical device numbers in the HP 3000. This device specification is used by programs that wish to make a generic reference to a device, and thus be independent of the logical device number assignments in a given HP 3000 installation.

The logical device number is related at system generation to a specific physical hardware I/O address. This number is only used when a particular program operating environment makes it necessary to allocate a specific device for the running of that program.

Spooling: The MPE spooling facility allows concurrent usage of devices which would otherwise be non-shareable, such as card readers, magnetic tape drives, or line printers. This is accomplished by copying the input or output to a disc where it is later processed. For example, if six users need to produce output on a line printer at approximately the same time, their output is directed instead to spoolfiles on disc from which it is printed on a priority basis as the line printer becomes free. In this way each user can immediately proceed with other processing activities without having to wait for the line printer. Similarly, if there are ten jobs to be read from a card reader or a magnetic tape unit, they are all read immediately and are directed to spoolfiles on disc where they wait to be executed. Thus, the card reader or magnetic tape unit is not tied up by one job which must be executed before the others can be read.

This process of copying input or output to disc as intermediate storage is called spooling [SPOOL is an acronym for Simultaneous Peripheral Operations On-Line]. The spooling of batch job input can be initiated not only from a card reader or a magnetic tape unit, but also from within an interactive session at a terminal by way of the STREAM command. The STREAM command temporarily transforms your session into an input spooler. Typically, you enter one or more job streams (each beginning with a JOB command and ending with EOJ command) and save them on disc using the HP 3000 text editor. You then issue a STREAM command whenever you wish the particular job streams to be executed. In response to the STREAM command, MPE copies the job streams to spoolfiles and notifies you of the job numbers assigned to each job stream. The session then continues independent of the spooled jobs.

Private disc volumes: MPE offers a private disc volumes facility which allows users who have the private volume capability to create and access files on removable disc volumes. These private volumes consist of removable disc packs which, when mounted on a disc drive, can be accessed by MPE through the file system.

Private disc volumes greatly increase the storage capability of an HP 3000 without the usual storage media such as punched cards, paper tape, or even magnetic tape. Under private volumes, the disc packs mounted on the drives during a cold load are dynamically allocated to the system domain for normal use or to the non-system domain for private use. Non-system domain packs can be both physically and logically mounted and dismounted during normal system operation. This has the advantage of increased data security, since the disc pack can be kept off the system until it is actually needed. Disc-to-disc backup for on-line applications saves valuable time in the event of a system failure, since restoring from tape is unnecessary. Where multiple HP 3000 systems are in use in one location, an application can be moved from one CPU to another simply by moving its disc volume.

Serial disc volumes: The MPE operating system includes a serial disc interface which allows non-system domain drives to be used as non-shareable serial devices. To MPE, the discs appear to be magnetic tape drives. They provide a fast backup and recovery capability when used as an alternative to magnetic tape in backing up the MPE system and storing and restoring files and/or data bases.

Tape labels: MPE offers you the ability to read and write labels on magnetic tapes. These labels can be used to:

- **Identify magnetic tape volumes (reels)**
- **Protect tape volumes from being inadvertently over written**
- **Protect private information**
- **Facilitate information interchange between computer systems**

You can read, but not write, IBM-standard tape labels; read and write ANSI-standard (American Standard Magnetic Tape Labels for Information Interchange) labels; and read and write user-defined labels on previously labeled magnetic tapes.

Processes and their execution

Processes: In an MPE environment, programs are run on the basis of processes created and handled by the operating system. The process is the basic executable entity in MPE. It is not a program itself, but the unique execution of a program by a particular user at a particular time. When a program's execution is requested, a private, hardware-protected data segment called a stack is created for that particular execution. The stack and the program's code segments together constitute the process. Thus, for example, each user working on-line with the BASIC interpreter is running the interpreter under a separate process. All use the same code (the only copy of the interpreter in the system) but each has his own data stack.

Invisible to you, MPE automatically creates, handles, and deletes all processes. However, optional capabilities are also available so you can create and manipulate processes directly.

Privileged mode: Privileged mode gives you the ability to access all MPE resources including intrinsics, system tables and CPU instructions. In essence, you can use the computer as if you were the operating system itself. Programs executed in an unprivileged (or user) mode cannot affect the operation of other users or the operating system. Privileged mode, however, bypasses all the normal checks and limitations that apply to standard users. Therefore, it is possible for a privileged mode program to destroy file integrity, including the MPE operating software itself. Hewlett-Packard cannot assume the responsibility for system integrity when user programs operate in the privileged mode.

Virtual memory: The virtual memory scheme employed by the MPE operating system utilizes both main (semiconductor) memory and disc storage to greatly expand the total amount of memory space available. MPE logically divides programs into variable length segments of code and data which reside in disc memory and are brought into main memory only when required for program execution. (See Figure 2-6.)

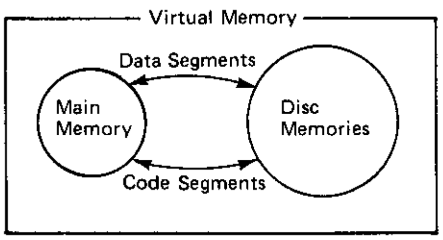


Figure 2-6. MPE Virtual Memory

When a code segment is no longer needed, it is overwritten by a new segment. If the segment is needed again, it is simply copied again from the disc on which the program resides. Data segments, however, are dynamic and their contents can change during execution. Therefore, when a particular data segment is no longer needed, it is automatically copied back to the system disc, replacing the previous version of that segment. This process of transferring segments between disc memory and main memory requires storage for local data to be allocated only as needed. Since this area is then automatically freed when the segment is no longer required, programs require less storage than when executed under conventional systems. The segmentation also allows one copy of a program to be shared by many users, while each one still operates in his own environment, free from interference by the others.

Automatic scheduling: MPE automatically schedules all jobs and sessions according to their priorities. When the execution of a running program is interrupted for any reason (such as input/output or an internal interrupt), MPE passes control to the program of highest priority awaiting execution.

All jobs/sessions running under MPE are scheduled by means of a single master queue, ordered by priority (see Figure 2-7). This master queue is divided into areas called priority classes. Each area is bounded by two priority numbers known to the system manager, but not to an individual user. You are aware of priority classes, but not priority numbers.

The processes which constitute the currently running jobs and sessions are granted control of the CPU by the MPE dispatcher. You may select a general category of process dispatching priority by including the PRI=parameter in your JOB or HELLO command. The four process dispatching priority types available to users are:

AS–	system processing only
BS–	very high priority
CS–	timeshare
DS–	batch
ES–	very low priority (background)

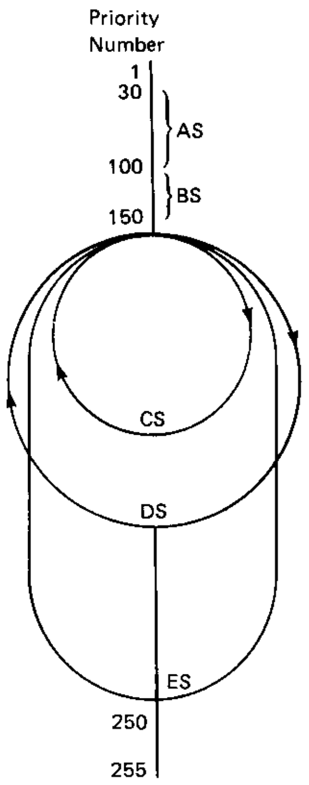


Figure 2-7. MPE Master Queue Structure

The MPE dispatcher translates these priority types into numerical ranges within one overall scheduling queue. The numerical range of each priority type can be changed while the system is running by entering an MPE command. This powerful capability makes it possible for the console operator or the system administrators to dynamically tune the system so that an optimal balance of service is maintained between jobs and sessions at any given time.

Batch job input spooled on disc is selected for execution according to the value of the job input priority parameter (INPRI=x) specified on the JOB command of each job. The console operator specifies the maximum number of batch jobs that can be executed concurrently by entering a command through the system console. Whenever the system is able to execute another batch job, the input spoolfile with the highest INPRI value is selected for execution. If there are two or more input spoolfiles with the same INPRI value, the oldest one is selected first.

Spooled output on disc is selected for processing according to the value of the output priority parameter (OUTPRI=x) specified on the JOB or HELLO command of the associated job or session. When the system is able to process another output spoolfile, the one with the highest OUTPRI value is selected. If there are two or more output spoolfiles with the same OUTPRI value, the oldest one is selected first.

Memory management: MPE divides main memory into two areas. The first, fixed memory, contains only the tables and code that the operating system requires to be memory resident. These include the interrupt handlers, the memory manager, and the scheduler. The remainder of memory, linked memory, contains all other code and data. User and operating system segments are brought into this area by the memory manager as they are required. The architecture allows most of the operating system, including the file system, the command interpreter, the spooler, and even much of the I/O system, to be shared by all users even though they are brought into main memory only when needed.

The MPE memory management system is responsible for the allocation of main memory to the executing processes. Program and library segments which are needed for execution are automatically swapped into main memory from disc. A segment is the basic entity for swapping transfers. When a segment in main memory is no longer needed, this is detected and such an unused segment becomes a candidate for automatic overlay by the system. An attempt by an executing process to access code or data not present in main memory creates an absence trap which causes the memory management software to allocate main memory space for the missing segment. When the absent segment is swapped from the disc memory to the main memory (is said to become present in main memory), the executing process continues. Frequently used segments may never be swapped, but will remain in main memory, while rarely used segments will be in disc memory most of the time. This results in higher efficiency and faster overall execution time. It also creates a dynamic situation in which segments are being swapped rapidly between main memory and disc memory, according to the demands of the executing programs, with many users executing the same or different programs at any given time.

Process execution: The dispatcher is the module of MPE that schedules processes for execution. Each process has a dynamically changing priority number, and the dispatcher keeps a list of active processes (those requesting execution) ordered by priority. This is called the ready list. The dispatcher manipulates the priority number so a process gets service appropriate to its creation parameters, attempting to run the highest priority active process. If that process is not in memory, the dispatcher requests the memory manager to make enough of that process's segments present in memory to allow it to continue.

As a process runs it may require another code or data segment. If the segment is not present in main memory, the hardware requests the memory manager to bring in that segment before that process is allowed to continue. While waiting for the completion of that transfer, some other process in memory may be run. It is clear that a process will run best when all the segments it references are in main memory. However, all the segments for all the processes usually will not fit in main memory. It is also unnecessary because most of the code in any program is executed infrequently. Thus, it is more efficient for the memory manager to bring in segments as they are required on an exception basis.

If the memory manager can ensure that the process has enough segments in main memory so that it only infrequently segment faults (stops execution to request additional segments), then the process will run efficiently and the overhead for virtual memory will be low. This gives rise to the concept of a working set, which is the set of segments required to be in memory for a process to run well. The problem is to determine what that working set is for each process. Very often even the programmer cannot guess what it is likely to be, because it changes dynamically during execution. MPE uses a "segment trap frequency" algorithm to determine which segments belong to each process's working set. This algorithm is highly efficient. A working set list is maintained for each program and the size of this working set is expanded or contracted in an attempt to arrive at a constant interfault time for each program. This is in effect a negative feedback control mechanism. MPE keeps track of the interfault time very accurately on a per-program basis with the help of a special process timer built into the CPU.

When a process implicitly requests an extra segment, the memory manager brings it into main memory after it has found space for it. At this time an important decision must be made—should this process have its working set expanded to include this segment or should one of its older segments be removed from its working set? This decision is based on the CPU time used by that process since the previous segment request. The decision is important because if the working set is expanded too much this process will tend to control more than its share of main memory, thus degrading the performance of other processes. On the other hand, if the working set is too small the process itself will run inefficiently even though it has little effect on any other processes.

When segments are to be removed from the working set, they are merely delinked from the list associated with that process and linked onto a single system-wide overlay selection list. Although this list is used later to find segments to be overlayed, at this point the segments remain in main memory. For a data segment an anticipatory write operation is initiated to update the virtual memory image of that segment. If there is more main memory on the system than is required to support all working sets of currently active processes, it is especially advantageous to leave the segments in main memory at this point. For example, if only one process is active it is conceivable that all the segments it has ever referenced are actually in main memory (because no other process has requested any memory) even though only a small percentage of them are in its working set. In this way a process can use main memory far in excess of its working set, but only to the extent that there is extra unused memory available at that time.

Another important memory manager decision is which segments to remove from main memory (i.e., overlay) when space is required to satisfy a new request for a segment. The memory manager looks for a segment to overlay if there is no free area of the required size. If there are any segments on the overlay selection list these will be overlayed one at a time until a space has been created that is large enough to satisfy the request. Overlaying a segment involves ensuring that the segment has been copied back to virtual memory if necessary, releasing the space it occupied in main memory, and coallescing the free space created with any adjacent free spaces that might exist. If another free area is found to be separated from the newly created one by one small movable segment then that segment will be physically moved in main memory to allow for combination of the two free areas. If the overlay selection list becomes exhausted before a large enough free space is found, the memory manager must turn elsewhere for help in predicting which segments in main memory will not be used in the near future. At this point it is known that all segments in memory are actually required by some process for it to run well. The memory manager now has no choice but to remove one of the processes from main memory temporarily. A communication mechanism has been set up with the dispatcher to assist in predicting which process is least likely to run in the near future.

When the dispatcher puts a process to sleep (temporarily suspends a process) it decides, knowing the reason for suspension of the process, whether that process is likely to wait for a long period before reactivation. If a long wait is likely, the dispatcher links that process to the end of a list called the discard list. The memory manager knows that a process on this list is taking up main memory but is unlikely to be needed soon. Processes are discarded by selecting them from this list one at a time and overlaying each segment in their working sets starting with the least recently used. This procedure is carried out until enough space has been released to satisfy the request on which the memory manager is working. If the discard list is exhausted before enough space is found, the memory manager can, as a last resort, scan the dispatcher's ready list and discard processes starting with the one having the lowest priority. In this way working sets of the processes highest on the ready list will remain in memory; these are precisely the processes the dispatcher will schedule next, and thus the memory manager is actually using knowledge of the near future to assist in its predictions.

Application program management
In the HP 3000 computer system program code is grouped into logical entities consisting of one or more procedures (subroutines). Each code segment may be up to 32k bytes long (k=1024). Programs may optionally be broken into multiple segments with procedures or subroutines fully contained within one segment.

Code segmentation: Three steps are involved in taking a program from source form to an executable state. The first step is to compile the source program into relocatable binary modules (RBMs). This is done by the various MPE language compilers which automatically store the RBMs in a specially-formatted file called the user subprogram library (USL). The second step is to take the USL file and prepare it into a program file. Program preparation resolves external references and results in loadable code segments. This is done by the MPE segmenter. The final step is for the MPE loader to allocate entries in the HP 3000 code segment table for all the segments in the program file and to allocate an entry in the HP 3000 data segment table for this process's data stack. At this point the segments are part of active virtual memory, all external references have been satisfied, and the program is scheduled for execution. Figure 2-8 illustrates the MPE code segment evaluation. Often all of these steps are initiated by a single MPE command (as an example, refer to the session illustrated in Figure 2-2). When necessary, however, you can initiate each step individually, controlling what happens along the way.

A code segment consists entirely of information that is not subject to change during program execution. This includes the instructions of the program itself, constants, and an area for interprocedure links. No modifiable data may be interspersed with the instructions in a code segment, and in no way is it possible to add onto a code segment once it is being executed from a program file or segmented library. It is this feature which ensures that all system code is re-entrant, meaning that any sequence of instructions can be in simultaneous execution by multiple users. All HP 3000 computer system procedures are potentially recursive.

The fact that system code segments are not modified during execution has specific advantages for the memory management system. Since all code segments are re-entrant, all are potentially shared by more than one user when present in main memory. For example, operating system services are provided in part by procedures (segments) contained in a system segmented library (SSL) shared by all programs requesting those services. Similarly, all users executing the same program file share the program's segments. In fact, only one copy of a segment needs to be in main memory, no matter how many users may be executing it concurrently. The result is an ability to handle more users in a given main memory capacity.

The second major advantage of code segment re-entrancy is that code segments are read only. Thus, they never need to be swapped from main memory back to disc, even when overlayed, because there is always an identical copy of the segment in the program file or library, from which it may be fetched whenever needed. Hence, code is only swapped into main memory,

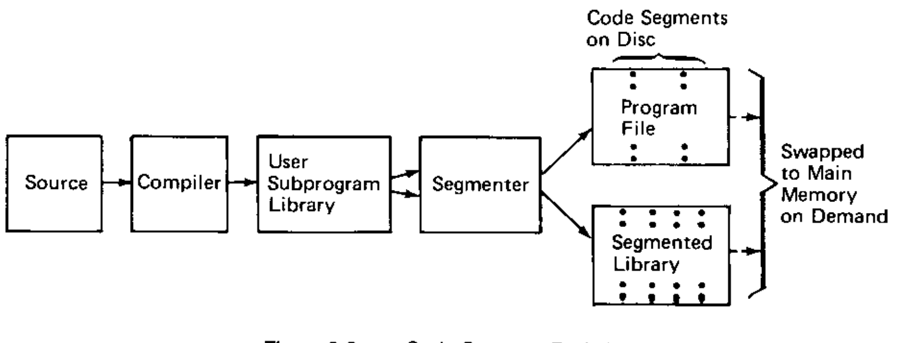


Figure 2-8. Code Segment Evolution

never out, as is shown in Figure 2-5. The resulting reduction in swap traffic is conducive to efficient management of main memory.

You initiate compilation, preparation, and allocation/execution of programs with the following set of MPE commands.

Prepare [PREP]–prepares a program from a single USL file.

Execute [RUN]–causes a prepared program file to be allocated and scheduled for execution.

Prepare and Execute [PREPRUN]–combines the PREP and RUN commands.

The subsystem commands which communicate with the language translator are consistent among the compilers. For instance, the commands to name segments or to produce symbol maps are identical for all of the HP 3000 compilers. This means that when you change languages you do not have to learn a new set of procedures or commands for interfacing with this language. They are identical to those for the language that you were previously using.

Accessing compilers: Three commands are available for accessing the HP 3000 compilers (except APL). These commands allow you to 1) compile only, which stores RBMs in a permanent USL file; 2) compile and prepare, which creates a permanent program file containing prepared segments—the temporary USL file created by the compilation process is lost; 3) compile, prepare and execute—the temporary USL and program files created during the process are lost.

For COBOL, these commands are:

To compile only,	**COBOL**
To compile and prepare,	**COBOLPREP**
To compile, prepare, and run,	**COBOLGO**

Access to the other compilers is identical except that "RPG", "FORTRAN", "BASIC", or "SPL" replaces "COBOL" in each of the three commands. [Note, however, that the command BASIC invokes the BASIC interpreter, whereas BASICOMP invokes the BASIC compiler. APL invokes the incremental APL compiler.]

It is important to note that the data files created by these languages are all generated by the same file system, the MPE file system. Therefore, these data files are shareable among the various languages. For instance, a file created by a BASIC program can be read by a FORTRAN program. Also, this file-sharing characteristic carries down to many HP 3000 subsystems, such as KSAM and IMAGE.

The segmenter: Program preparation is performed by the segmenter in response to a PREP command or one of the combining forms which includes PREP or GO. Segmenter commands may also be used to manage USL files by adding, deleting, activating, or deactivating RBMs within them and to build and manage segmented libraries (SLs) which are used to resolve external references from user programs.

Occasionally you may wish to alter the segmentation of a program to improve its run time efficiency. In many systems you would have to recompile your program in order to do this. With MPE, however, the segmentation can be modified by simply using the segmenter to rearrange the RBMs and then preparing the USL file into a new program file.

A convenience feature of the segmenter is the ability to have different versions of the same subprogram within a single USL file. The segmenter commands have an optional index parameter which specifies the "i'th" most recent definition of an entry point within a particular subprogram. By using this optional index parameter you have complete control over the activating and deactivating of entry points within various versions of your subprograms.

Procedure libraries: When a program is allocated and scheduled for execution, MPE searches up to three libraries for unresolved external references. In order of search, these are: 1) the library of your log-on group, 2) the library of the public group of the log-on account, and, 3) the library of the public group of the system account. Each of these three libraries can possess one or two types of library files, categorized as that group's library programs. These files are named SL (a segmented library containing procedures in segmented form) and RL (a relocatable library containing procedures in RBM form). When a particular library is searched, the RL file is searched first, then the SL file.

Procedures contained in the SL file are in prepared form, that is, they are segmented. When a particular procedure is needed, the segment containing the procedure is loaded, as are all external references from that segment. Because the segmentation has been pre-defined in this manner, these procedures may be shared between programs, and only one copy will exist at any time in virtual memory even though several users may require a particular procedure concurrently.

Procedures contained in the RL file are not in segmented form; rather they are in RBM form and must be segmented before they can be loaded with your program. The procedures that come from an RL are placed into a new segment that is linked to the program. Since this segment is specifically constructed for a given program, procedures loaded from the RL may not be shared between different programs and may indeed be duplicated for each user requesting them.

The combination of segmented libraries and relocatable libraries gives great flexibility for storing often used subroutines or procedures. Procedures used system wide are normally stored in the segmented library at the system level. Procedures used by only a few users or a single group are stored in RLs at the group, account or system level.

Special segmenter commands are available to build an SL or RL within a particular group. The default name of SL will be used for the segmented library and RL for the relocatable library. In addition, commands are available to add routines to the SL or RL, purge routines, and list the procedures contained in either the SL or RL.

System management

This section deals with the day to day operational aspects of running an HP 3000 installation. For all its features and power, MPE is surprisingly easy to work with. A short overview of the basic steps involved in preparing a new HP 3000 to execute application programs precedes details on the tools MPE provides for easy and successful system management.

When you receive your system, your first task is to configure MPE for that particular system configuration. This is an interactive process that takes place on the system console and requires your intervention only if you choose options that are not system defaults. The configuration process on MPE is very simple and straightforward, completed in a matter of minutes. Following the initial configuration it is possible for you to configure MPE systems from your terminal while MPE itself is in operation. In other words, you can build MPE systems for use on other HP 3000 systems or for later use on your current HP 3000 from any terminal on the system.

The next step after the system is configured is to build the various accounts and users that are to run on the system (see Figure 2-9). This is done by first assigning the system manager capability to a specific user. He has the overall control of the system in the form of being responsible for allocating various accounts within the system. Once these accounts are defined, it is the account manager of each account who is responsible for defining the users that will be able to log onto the system. With the information that is provided by this accounting process, it is possible to determine what portion of the system resources has been used by various individuals. This is important for the billing process and for general knowledge of system utilization. The system

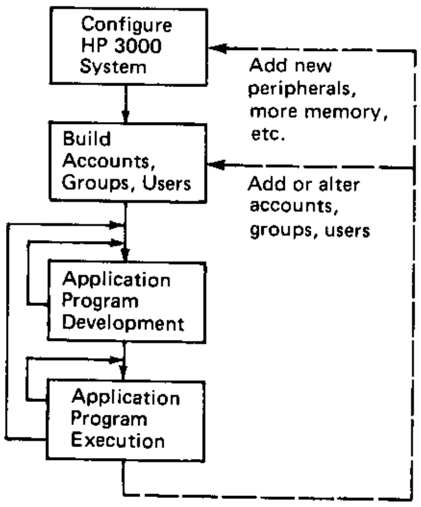

Configure HP 3000 System
Add new peripherals, more memory, etc.
Build Accounts, Groups, Users
Add or alter accounts, groups, users
Application Program Development
Application Program Execution

Figure 2-9. HP 3000 Operations

manager can control the amount of disc space that is available to the accounts and the amount of CPU and connect time they can use. If they exceed these values, it is up to the account manager to ask the system manager for larger quantities. The system manager has the option of putting no limit on these resources for the accounts. Likewise, once the account manager has been given his share of disc space, CPU time, and connect time, he can allocate it among his own users.

Once the system is configured and the account and user structure has been defined, the next step is program development. This is performed interactively on the HP 3000 using terminals and a text editor for the development of source programs. You also have the ability to develop programs in the conventional batch processing mode. Once programs are developed, they are executed. Therefore, the primary task of the HP 3000 operating system from this point on is controlling the development of programs and their execution.

Occasionally the account manager will want to add new users to his account, or it may be necessary to reconfigure the system as new peripherals or more memory is added. These are straightforward processes and involve a minimal amount of time.

The system administrators: Throughout this manual you will find references to several types of system administrators such as the system manager, the system supervisor, and the account manager. These administrator types are each associated with a particular set of functional capabilities. For example, there is a unique set of MPE commands that can be used only by someone who has been designated as the system manager. (See Table 2-2.)

There is no formal requirement as to how these sets of functional capabilities are divided among the users of a particular HP 3000 system. It is only required that at least one person be designated as each type of administrator. The responsibilities can be divided or shared in virtually any combination desired. In a small installation a single person may be designated as the system manager, the system supervisor, and the account manager for all accounts in the system. In larger installations the responsibilities may be divided among several individuals—one or more system managers, one or more system supervisors, and one or more account managers for each account in the system.

It is important to bear in mind that these capabilities are granted to users by the assignment of special attributes, and in no way imply formal responsibilities or duties. Thus, a regular programmer may have the system manager capability or the system supervisor capability, or both capabilities. For this reason, it is best to speak of "a user with system manager capability" rather than a formal "system manager."

Table 2-2.
System Administrator Responsibilities

Administrator	**Responsibility**
System Manager	Manages the overall system by creating accounts and defining resource-use limits (if any) for each account.
System Supervisor	Manages the system on a day-to-day basis. Can manage the scheduling queues, alter the system configuration, maintain the system library and display various items of system information.
Account Manager	Manages one or more accounts by defining the valid users and file groups and specifying resource-use limits (if any) for them.

The console operator function is available to anyone with access to the HP 3000 console; the person using this function need not be an HP 3000 user logged onto the system under an account. He simply alerts MPE to a request through the console. The console prompts with a special character "=" and expects a single command to be entered. The commands entered in this fashion are special and deal primarily with peripherals and other hardware system concerns. System start-up and shut-down are also performed at the HP 3000 console.

Start-up and modification of MPE: MPE is initially brought up on an HP 3000 by the console operator at the system console, through a simple restart operation. There are several restart modes:

- **WARMSTART–The system is restarted from the disc. Spoolfiles are recovered and incompletely processed spooled jobs can be automatically restarted.**
- **COOLSTART–The system is restarted from the disc. The spoolfiles in existence at SHUTDOWN time are not saved, but all resident user files are saved.**
- **COLDSTART–The system is read from a serial storage device (magnetic tape or a serial disc). The I/O configuration and system configuration from the tape are used to define the system. These are merged with the directory and files present on the disc.**
- **UPDATE–Similar to cold start, except that the I/O configuration and system configuration on the disc are used to define the system. This mode is used when starting the system with an updated version of MPE from Hewlett-Packard.**
- **RELOAD–The entire system, directory, files, and configuration information are read from a serial storage device.**

The restart operation can include an interactive dialogue between the console operator and the MPE initiator program. This optional dialogue permits the operator to change the system configuration. Upon completion, MPE is operational.

All HP 3000 computer systems operate under a single operating system—MPE. This means that programs prepared on one HP 3000 operating under MPE III can be run on any other HP 3000 running under MPE III without modification (provided that all devices required by those programs are connected on-line). It also means that if you move from one installation to another you don't need to undergo additional training or read additional documentation to prepare yourself for the new environment.

System backup and recovery: The SYSDUMP command, which is available to the system supervisor, is used to back-up the MPE system and user files on a serial storage device (magnetic tape or a serial disc). This serial storage device media may then be used to reload the system following a hardware or software failure or to transfer the system to another hardware installation.

If the system supervisor is using SYSDUMP to create media for reloading on a different or changed HP 3000, he may specify a new I/O configuration and system table configuration. In this case SYSDUMP initiates an interactive dialogue which allows him to specify the desired changes. There are eight possible areas of change, any or all of which may be skipped. These areas are:

- I/O devices
- System table sizes
- System disc allocation
- Logging facility
- Miscellaneous changes
- Scheduling changes
- Segment size limits
- System modules to be included

The dialogue also allows the system supervisor to define a back-up date. All disc files altered on or after this date will be copied to the back-up media. This facility can be used to back-up specifically those disc files which change each day.

Accounting facility: The MPE accounting facility provides you with a flexible and powerful means of coordinating access to the system and disc file usage.

Coordinating system access: The system administrators of an HP 3000 installation can devise a structure of accounts and users which reflects the functional organization of the people who use the system. Access to the system is partitioned into accounts, each having a unique account name. Individuals who are permitted to initiate jobs and sessions under a particular account name are assigned user names within that account. People who work together on a project or who do the same type of work (such as entering incoming orders through terminals) are frequently assigned the same user name within an account. (See Figure 2-3.)

When initiating a job or session, you supply (as part of the JOB or HELLO command, respectively) your user name and account name. To prevent unauthorized access to the system, a password can be associated with each account and user name. Attempts to initiate a job or session which do not include all defined passwords will fail.

MPE maintains running totals on the amounts of system resources that each account consumes, including disc space used, cumulative CPU time consumed, and (for sessions) cumulative terminal connection time. The current totals can be displayed at any time and can, of course, be used for billing purposes.

The system manager may establish limits on the amounts of system resources that a particular account can consume. If one of the cumulative time limits is exceeded, no more jobs or sessions can be initiated under that account name until the system manager either adjusts the particular limit or clears the offending total. If the account's disc space limit is exceeded, jobs or sessions running under that account name may not acquire more permanent disc space until either the system manager adjusts the limit or someone within the account frees some disc space.

The system manager can also allocate certain special capabilities to accounts which he feels are qualified to use them. These include the ability to manipulate the accounting structure, to handle system log files, to create, manipulate, and delete processes, to create extra data segments, and to operate in privileged mode (which allows the account users to access all MPE resources, including intrinsics which are otherwise uncallable).

Coordinating file usage: The overall permanent disc file domain of the HP 3000 is partitioned among the various accounts. Each account's file domain is further partitioned into groups.

If a request to save a file would result in exceeding the permanent file space limit at either the account or group level, the request is denied.

The fully-qualified name of any file includes the file's name, the name of the group in which it resides, and the name of the account in which it resides. In actual practice, however, you often do not have to supply all three names to access a file.

When initiating a job or session, you may specify a group name in addition to your user name and account name. (Each user has a home group assigned to him by the account manager. This home group is the default if he does not specify a group when initiating a job or session.) If you subsequently reference a file which resides in both the account and group under which you are running, you may omit the account and group names from the file reference. For example, assume you are running a session under the account name ACCT1 and the group name GROUP 1. In this case you can refer to the fully-qualified name FILE1.GROUP1. ACCT1 as merely FILE1.

The primary advantage of the use of fully-qualified file names (whether specified explicitly or implicitly) is that different files with the same file name can exist in two or more accounts, or in two or more groups within an account, with no conflict. In such a case, the account and group names differentiate the files from one another.

You may create files only in the account and group under which you are currently running. By using fully-qualified file names, however, you have access to any file present in the system (provided that existing security provisions allow that access).

Operational management: In addition to all commands available to the standard user, the user with system supervisor capability can enter various commands which enable him to have supervisory operational control of the system. These commands can be entered in either session or batch mode. The session mode gives the system supervisor the option of requesting functions from a remote terminal.

The system supervisor can control the master scheduling queue in two ways. First, by entering the QUANTUM command, he can change the time quantum of any subqueue. Second, he can display information about the scheduling of processes and the contents of various subqueues or the master queue, by issuing the SHOWQ command.

Two important commands allow him to tune the system by permanently allocating a procedure or program in the HP 3000 virtual memory. For large, frequently-used routines, this may considerably enhance system performance.

ALLOCATE–permanently allocates the named program file or procedure in the HP 3000 virtual memory.

DEALLOCATE–removes a program file or procedure and its external segments from the system's virtual memory after all processes currently running have finished accessing the named procedure.

In addition to the above operations, the system supervisor can also exercise control over process scheduling through certain intrinsics and can manage the logging facility (as noted in the next section).

The logging facility: The MPE logging facility optionally monitors the use of system resources by recording details of system resource requests in a series of log files on disc. The system supervisor of each installation selects those system and user events that are to be recorded in the log files.

The log files are ideal sources of information for billing purposes, since each log record is correlated to the job or session that caused the event. There are log records for job or session initiation or termination, program termination, file closing, file spooling completion, and system shut-down. The contents of the log files—the individual log records—are not displayed or otherwise used by MPE; however, the installation may write various analysis routines for this purpose. Details contained in each log record permit an installation's accounting programs to bill accurately for system usage. This is a problem solver in any installation where resources are shared by fiscally independent users.

I/O device failures on any device are also recorded in the log files, which are then available for detecting problems with devices before they begin to interfere significantly with the overall system operation. Note that I/O failures for disc and magnetic tape are automatically retried several times by the operating system and recorded in the log files.

Power fail/auto restart: When an AC power failure is detected by the HP 3000, a power fail/auto restart routine is automatically invoked. This routine preserves the operating environment prior to complete loss of power. Normal system operation resumes as soon as power is restored. Jobs and sessions in progress on the system continue where they were interrupted, with programs unaware of the interruption (except for magnetic tape units and dial-up terminals actually in use when the power failure occurred).

Documentation

There are thirteen manuals that together introduce the HP 3000 computer system and document the Multiprogramming Executive operating system. Figure 2-10, which is repeated in the preface of each operating system manual, shows the relationship of each manual to others in the set. As indicated by the diagram, the thirteen manuals in the set are organized by function into four levels:

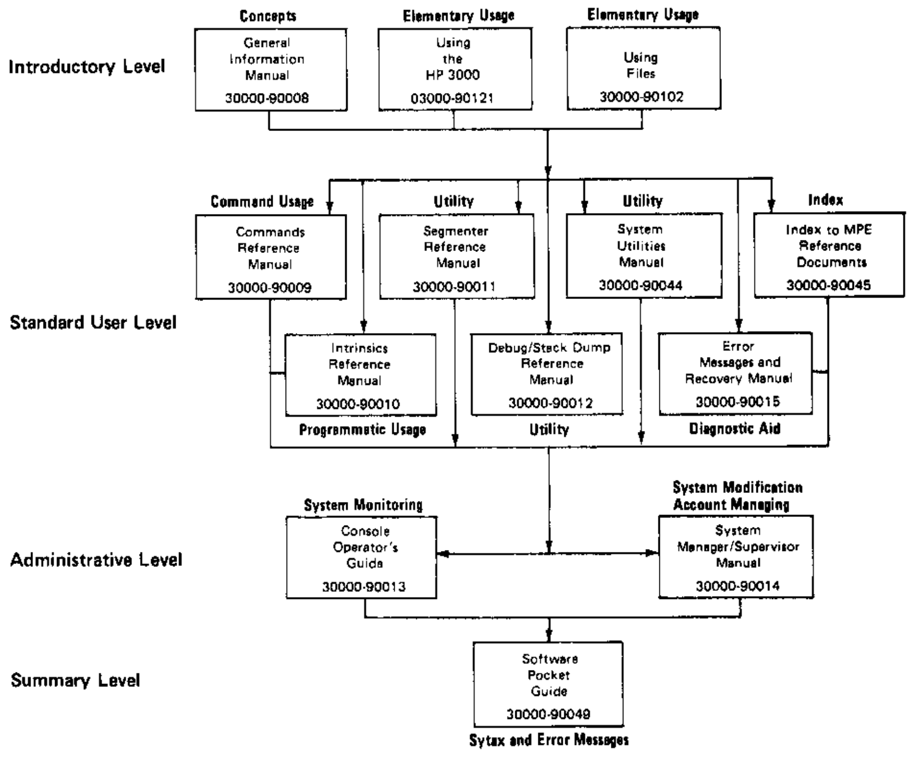

Introductory Level
Concepts
General Information Manual 30000-90008
Elementary Usage
Using the HP 3000 03000-90121
Elementary Usage
Using Files 30000-90102
Standard User Level
Command Usage
Commands Reference Manual 30000-90009
Utility
Segmenter Reference Manual 30000-90011
Utility
System Utilities Manual 30000-90044
Index
Index to MPE Reference Documents 30000-90045
Intrinsics Reference Manual 30000-90010
Programmatic Usage
Debug/Stack Dump Reference Manual 30000-90012
Utility
Error Messages and Recovery Manual 30000-90015
Diagnostic Aid
Administrative Level
System Monitoring
Console Operator's Guide 30000-90013
System Modification Account Managing
System Manager/Supervisor Manual 30000-90014
Summary Level
Software Pocket Guide 30000-90049
Sytax and Error Messages

Figure 2-10. MPE Documentation

- INTRODUCTORY LEVEL manuals
  These manuals introduce all the major concepts of the HP 3000 computer system and provide an introduction to the use of the system.
- STANDARD USER LEVEL manuals
  These manuals provide the detailed reference documentation for the entire operating system—excluding only the system administration function. Of these manuals, the Commands Reference Manual is essential for using the HP 3000. The Error Messages and Recovery Manual summarizes the error messages and recovery actions for operating system and language/utility subsystem errors. The Index to MPE Reference Documents is a master index for the entire set of MPE operating system manuals. The other four manuals cover more advanced programmatic features, going still more deeply into the operating system itself.
- ADMINISTRATIVE LEVEL manuals
  These manuals provide a guide for day-to-day monitoring of the system by a computer operator, and, on a higher level, provide the more complex information required for system management—such as setting up an accounting system and modifying the operating system when necessary.
- SUMMARY LEVEL manual
  The Software Pocket Guide summarizes the MPE commands and intrinsics (system procedures), the MPE file system error messages, and the language and utility subsystem commands, in a convenient pocket size. This guide is primarily a memory aid and assumes a familiarity with the effects of the commands, intrinsics, and their parameters.

All of the available manuals documenting the operating system, utilities, languages, data entry, data management, data communications, hardware, and conversion for the HP 3000 computer system are summarized in Appendix A.

Chapter 3: Processing environments

Transaction processing
In a very broad sense, the expression "transaction processing" can be applied to virtually any interaction between a computer system and its users. This definition holds true regardless of whether the transaction originates from keyboard input or a series of cards read by a card reader. The term "transaction" implies that there is some sort of exchange taking place—and this is indeed the case. It is a series of steps in a predefined logical sequence which accepts input, accomplishes some sort of data processing or manipulation, and generates output which demonstrates that some identifiable action has taken place.

Today, many businesses are moving away from large batch processing computer systems toward networks of smaller computer systems running innovative on-line transaction processing applications. These installations are typified by a relatively large number of users who communicate with the computer system through terminal devices and present the computer with an uneven processing load, large amounts of terminal and disc I/O and a high degree of code and data sharing among all the users on the system. These transaction processing systems (TP systems) are designed to provide each terminal user (on-line to a computer system and communicating with an application program) access through transaction activity to the information stored in the computer's data bases and files.

The major advantages of an interactive transaction processing system are as follows:

- It allows the entry, manipulation and retrieval of data at various dispersed locations—usually the locations where the data is actually generated and used for business decisions. This allows the people who are most familiar with the data to enter and interact with it.
- It speeds up the business cycle. Rapid and accurate data input and retrieval can be performed by relatively inexperienced personnel to make up-to-date information constantly available. This results in faster response to customer demands, thereby providing a major competitive edge.

Views of transaction processing: Based on the concept that each user interacts with the computer, the logical first step is to determine what the users need. This means that you must look not only at the needs of the data entry clerk, but also at the day to day needs of the business operation, management, and corporate staff.

This investigation of the end users' needs is the responsibility of the system designer or analyst. He must translate the needs of the user and the business into some sort of logical specification which the transaction processing system programmer can interpret in a language the computer can understand. The programmer must create the particular set of programs, subroutines and procedure calls that actually result in an implementation of the transaction processing system.

In addition, there is one more viewpoint that must be examined, that of the computer system on which the transaction processing application is intended to run.

In summary, there are actually four sets of requirements which must be examined, those of the user, the system designer, the programmer, and the computer system itself.

The user
- wants a conversational interface to the application program.
- wants the ability to provide one-time reports without additional programming effort.
- doesn't want to get involved in learning the computer system software.

The system designer
- needs to interview end users to determine system requirements.
- needs to keep in mind the important aspects of a transaction processing system—fast response time, rapid information access, and high transaction throughput.
- should be provided with design aids to help his analysis of the application needs.

The programmer
- needs to be familiar enough with the computer to translate the system designer's specifications into an actual application program.
- should be provided with system characterizations which inform him about performance considerations and design trade-offs when using the computer system.
- should be provided with programming tools which increase his efficiency in writing, debugging and maintaining the application program.

The computer system and its manufacturer
- must provide hardware and software which operate well in a transaction processing environment.
- must provide all of the design aids, system characterizations, software development tools and system software which allow the system designer and programmer to design and implement a successful transaction processing system.

Transaction processing on the HP 3000: There are several approaches that provide the designer of an on-line transaction processing system great flexibility in implementing his application to prompt terminal users to perform a specific task. In the most commonly used approach each user logs on as a separate session and invokes an application program with a RUN command, as illustrated in Figure 3-1.

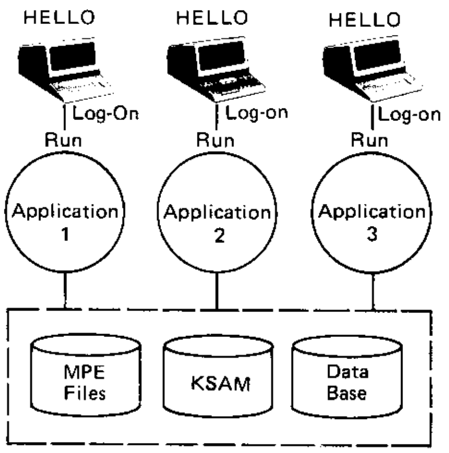


Figure 3-1. On-line Transaction Processing

An alternative approach, which also runs one process per terminal, can completely isolate the end-user from the HP 3000 command language. As illustrated in Figure 3-2, a single process (father) manipulates multiple processes (sons), which have no connection with the MPE command language during their execution. Each son process opens a terminal and prompts the user to perform a specific transaction via a menu or screen display. The terminal user merely enters the required data, without ever having to issue a RUN command. When the user is finished with the application, the son process can relinquish control of the terminal, which can then be used in normal interactive mode.

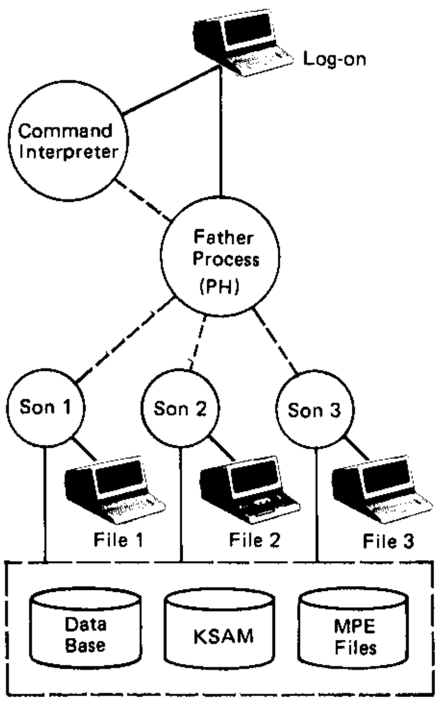


Figure 3-2. Alternative Approach to On-line Transaction Processing

Resources: A transaction processing system must achieve good system throughput and resource utilization while it simplifies the design, development, and maintenance of the programming involved. The HP 3000 is equipped with a complete set of tools which fill the needs of a transaction processing system and make the implementation of that system an easy task. (See Figure 3-3.)

- Multiprogramming Executive operating system (Chapter 2)
- File system (Chapter 2)
- HP-provided security (Chapter 2)
  –system security
  –data (file) security
- IMAGE: data base management system
- QUERY: English-like inquiry facility for IMAGE data bases
- KSAM: Keyed Sequential Access Method
- DEL: Data Entry Library
- High-level programming languages
- Distributed Systems Network: powerful tool for collecting, reporting and sharing data among physically remote locations (Chapter 4)

IMAGE/3000

As data handling requirements become more complex in terms of the number of files needed and the inter-relationships of data within the files, it makes sense to consider a data base management system. IMAGE is Hewlett-Packard's powerful data base management system which handles multiple files and helps you define and create a data base tailored to your special requirements.

What is a data base and why use one: A data base is a collection of logically-related files containing both data and structural information. Pointers within the data base allow you to gain access to related data and to index data across files.

The primary benefit of the IMAGE data base management system is the time savings which result primarily from file consolidation and the ease of program development and maintenance.

File consolidation: Most information processing systems that serve more than one application area contain duplicate data. Consolidation of multiple files into a data base eliminates most of this data redundancy. Through the use of pointers, logically related items of information are chained together, even if they are physically separated, allowing the data to be used by any program needing it. Since there is only one record type to retrieve and modify, the work required for data maintenance is greatly reduced, and all reports drawn from those items of information are consistent.

Program development and maintenance: IMAGE allows the data base to be structured independently of the application programs that use it, so that programs only need to define those data items actually used, rather than the entire file. As a result, changes can be made to other parts of the data base without requiring program modification. Coding can begin before the data base structure is finalized, as long as the format of the individual data items accessed by the program is known. Future enhancements to the data base can be implemented without impacting existing programs. The logical separation of the structure of the data base from the applications results in greater flexibility and substantial time savings for both the data base designer and the programming staff.

IMAGE data base structure: IMAGE is primarily a path oriented or chained approach to data retrieval. Pointers are maintained which logically connect those records with common attributes into chained lists. This allows cross-referenced access to collections of data down to the smallest unit and makes it possible to access related data very quickly.

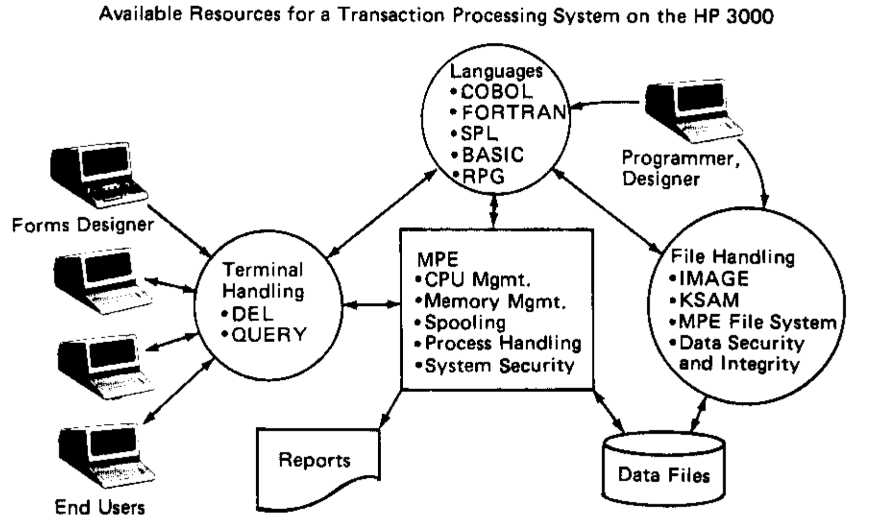


Figure 3-3. HP 3000 System Resources

An IMAGE data base consists of data items, data entries, and data sets. A data item (field) is the smallest accessible data element. It consists of a value referenced by a data item name. In general, many data item values are referenced by the same data item name.

A data entry (record) is an ordered set of related data items. The order of the data items within a particular data entry is specified when the data base is designed. The length of a data entry is the combined lengths of all data items within it.

A data set (file) is a collection of data entries. Each data set is referenced by a unique data set name.

A data base is a named collection of related data sets. It is defined in terms of data items and data sets. A data base may be defined with up to 255 data item names and 99 data set names.

Types of data sets: An IMAGE data set is either a master or a detail data set.

Master data sets

- are used to store information relating to a uniquely identifiable entity, for example, information relating to an employee (such as his name, identification number, address, and salary).
- allow for rapid retrieval of a data entry. One of the data items in the entry, called the search item, determines the location of the data entry.
- can be related to detail data sets containing similar search items and thus serve as indexes to the detail data set.

Detail data sets

- are used to record information about related events, for example, information relating to each pay period for each employee in the master data set.
- allow retrieval of all entries pertaining to a uniquely identifiable entity. Thus, an employee identification number may be used to retrieve all data entries containing information about previous pay periods for the individual with that number.

As illustrated in Figure 3-4, each master data set may serve as an index to a maximum of 16 detail data sets, and each detail data set may be indexed by up to 16 master data sets (or none).

Accessing the data base: You access a data base through call statements to IMAGE Data Base Management Subsystem (DBMS) procedures in COBOL, FORTRAN, BASIC, or SPL programs, or through compiler generated calls in RPG programs. These options give you the ability to select the language most appropriate for each application. The IMAGE procedures locate data, maintain pointer information, manage the allocated file space, and return status information to you. The tasks performed by the IMAGE procedures relieve you of the "bookkeeping" normally associated with file management and allow you to concentrate on applying the processing power of whatever programming language you are using.

Some of the functions performed by IMAGE procedures are:

- Adding a new entry to a data set
- Deleting an entry from a data set
- Reading some or all of the data items of an entry
- Changing the values of data items of an entry

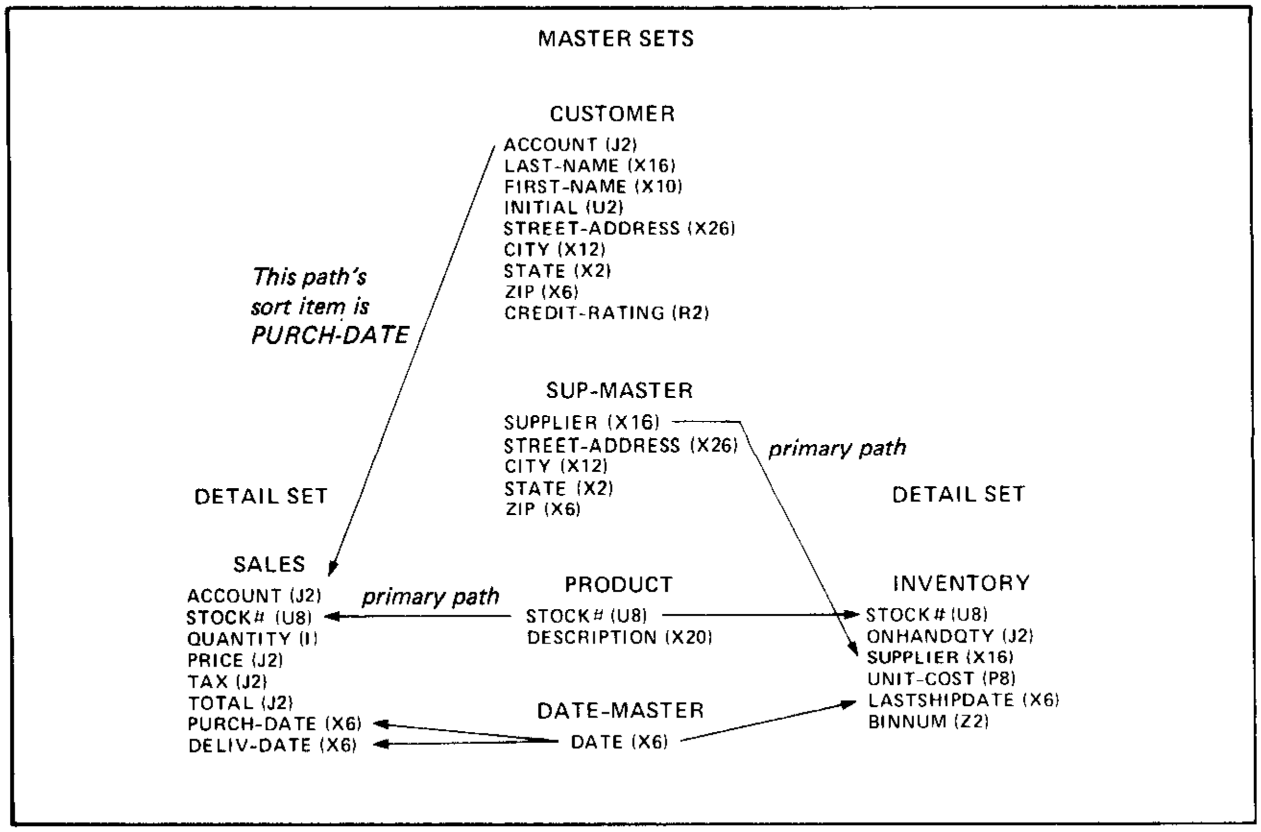


Figure 3-4. Example of Master-Detail Data Set Relationships

There are four modes of access available:

Serial–In this mode IMAGE starts at the most recently accessed record and sequentially examines successive records without regard to their key value. Forward and backward serial access is available.

Directed–In this mode the calling program specifies the record address of the data entry from which the desired data items are to be retrieved.

Calculated–This type of access involves retrieval of master data set entries based upon the search item (key) value.

Chained–This type of access consists of successive retrieval of all entries in a detail data set which contain the same search item value.

Data base security: In addition to the security safeguards of MPE, IMAGE provides the data base administrator with further protection for the data base. The key to the strength of IMAGE/3000 security is the ability to control access not only to specific data sets (files), but also to each data item (field). (See Figure 3-5.) The data base designer can define up to 63 user classes, each with a related password, and associate each class with either read or write access to specific data sets and items.

Therefore, the data base is protected down to the smallest unit of information, for example, a data item specifying an employee's pay rate. This ensures that the only users who access and/or change specific items are those whose job functions require it.

Because multiple users can simultaneously access and update an IMAGE data base, data integrity must be assured. You may specify the type of access you need—e.g. read, modify, update, etc. directly from the terminal or through a program. IMAGE will allow that access unless it conflicts with the functions already being performed by the other concurrent users. This approach results in compatible sharing of the data base and ensures data integrity.

QUERY/3000

QUERY is a companion data base inquiry facility for IMAGE. A self-contained language, QUERY provides a simple means to access the data base through the use of English language key words and other character strings. You communicate with QUERY through 17 unique commands to store, modify, retrieve, and report on data in an IMAGE data base. Commands can be entered either from an interactive terminal or a batch input device such as a card reader.

Applications: Since QUERY is designed for the non-programmer, it can be employed in a variety of applications after only minimal training. Both novices and experienced programmers find it extremely valuable. Some of the major application areas include:

- Casual inquiry of the data base—facilitates searching a data base for information without writing a program or waiting for a periodic batch run.
- On-line data updates—permit modification or deletion of data on-line, directly from a terminal to the IMAGE data sets (files).
- Report generation—formats reports with header and column labels, page numbers, group labels and summary statistics. Predefined report formats can be catalogued in the system to aid inexperienced users.
- Application program debugging—aids in program development. QUERY can be used to build test data as well as to interrogate the results of program and system tests. This feature eliminates the requirement that file-related programs be completed before meaningful functional programs can be written.

Features: In addition to the applications listed, QUERY performs several other valuable functions.

- Selection of data through logical comparisons—locates specific entries for processing based on logical criteria specified in a FIND command.
- Creation and storage of procedures—stores frequently used or lengthy commands as individual procedures in a command file, avoiding the necessity of retyping them when needed.
- Display of the data base structure—displays structural information about the data base and shows the relationship between data items and the data sets they are located within.

Security provisions: QUERY adheres to all of the security provisions included in the IMAGE data base. After QUERY is invoked, a security password must be entered which determines which data entries and data items may be accessed.

Data types: QUERY manipulates many of the various IMAGE data types. Each specific data type has a length ranging from 2 digits plus a sign to 132 characters. The types of data are:

- One-word integer numbers
- Two-word integer numbers
- Two-word real numbers
- Four-word real numbers
- One-word logical values as absolute numbers
- Zoned decimal numbers
- Packed decimal numbers
- ASCII character strings
- ASCII character strings containing no lowercase alphabetics
- One-word integer numbers corresponding to COBOL computational data
- Two-word integer numbers corresponding to COBOL computational data

Inquiries: The FIND command locates entries in a data base. Logical selection criteria, included in the command, allow only the entries pertaining to a given inquiry to be returned. Up to 50 logical relationships can be specified in one command.

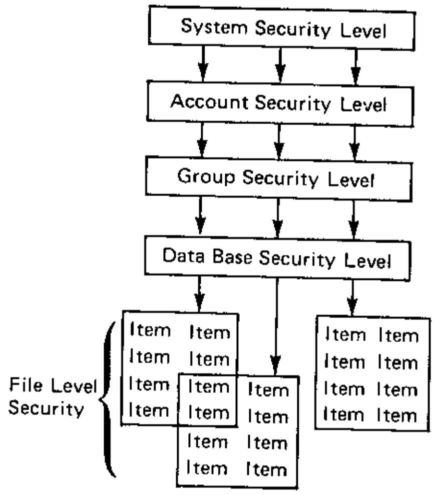


Figure 3-5. Total System and Data Base Security

Reporting: Data entries can be displayed or reported according to a user-specified format. After the data records have been located by the FIND command, a REPORT command is used to specify which items within those records QUERY is to display. REPORT also specifies such items as titles, column headings, page numbers, and line spacing. Figure 3-6 illustrates a QUERY report.

Updating: A data base may be updated by adding, deleting, or modifying an entry using the UPDATE command, which operates within the limits established for a password. QUERY prompts for the values needed to complete a transaction.

KSAM/3000

KSAM/3000 (Keyed Sequential Access Method) provides file access by key values within the data record. Each data record contains one primary key field and may include up to 15 alternative ones. Access of these records can be sequential or random by either primary or alternative key value. KSAM/3000 also supports key access by physical or logical record number, or by chronological order.

KSAM/3000 users may also retrieve data by using part of a key rather than the entire key. Called partial or generic key search, this approach is ideal for values that share a common beginning. Suppose you wish to find all the customers in a certain geographic area. By specifying only the common first three characters of the zip code—like 950, it is possible to read quickly 95050, 95060, 95065, etc. Plus, generic key searches can often help refresh your memory. "I think the brand name begins with Ba. Let's see, is it Bates, or Battes...?"

For those users with existing indexed sequential files, KSAM/3000 facilitates their conversion to the HP 3000. For example, RPG programs on an IBM System/3 using indexed access method require no coding changes to run on the HP 3000 using KSAM.

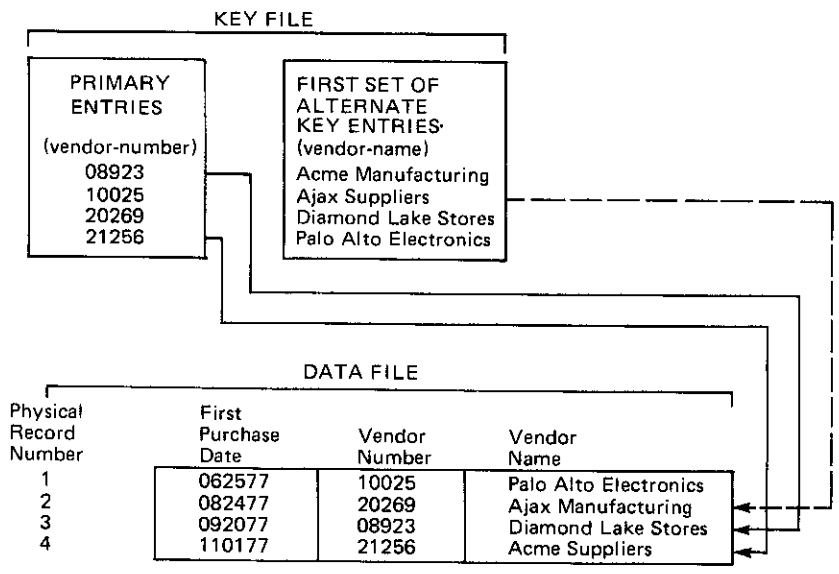
KSAM file structure: A KSAM file consists of two physical disc files—one for data and one for keys. The data file contains all the data records, while the associated key file holds one or more sets of entries that maintain the primary and alternate logical sequences of the data records. When constructing a file, key entries are dynamically added in ascending order. Any key file restructuring required to accommodate additional data file records is performed automatically by KSAM in an operation that is invisible to you. Figure 3-7 illustrates a KSAM file with a primary key and one alternate key.

```
>FIND VENDOR IS "AJAX" AND ORDER-STATUS IS "O"

3  ENTRIES QUALIFIED

>LIST ORDER-NUMBER,QTY-LEFT-TO-REC FOR VENDOR IS "AJAX"


ORDER-NU  QTY-LEFT

TESTPO          10
PO-995         875
PO-995        5000

>REPORT
>>H1,"OPEN PURCHASE ORDERS FOR AJAX",60
>>H3,"PART-NUMBER",14
>>H3,"ORDER-NUMBER",32
>>H3,"QUANTITY",45
>>H3,"DUE-DATE",58
>>H3,"CONTROLLER",70
>>S1,DATE-DUE
>>D1,PART-NUMBER,18
>>D1,ORDER-NUMBER,30
>>D1,QTY-LEFT-TO-REC,44
>>D1,DATE-DUE,57
>>D1,CTLR,65
>>END
                                   OPEN PURCHASE ORDERS FOR AJAX

    PART-NUMBER         ORDER-NUMBER        QUANTITY        DUE-DATE   CONTROLLER
200000                    TESTPO               10           042878        99
AN63                      PO-995              875           071578        40
300001                    PO-995             5000           072678        99
```

You can obtain a simple report by merely using a LIST command.

To request a more complex, formatted report, you can issue a REPORT command and specify headings (H1, H3), detail lines (D1), and sort sequence (S1).

Figure 3-6. A Typical QUERY Report Procedure



Chronological Order 1, 2, 3, 4
PRIMARY KEY (vendor-number) order 3, 1, 2, 4
FIRST ALTERNATE KEY (vendor-name) 4, 2, 3, 1

Figure 3-7. KSAM Example

Data fields, including key fields, may contain signed binary integer, double-word signed binary integer, real, four word long real, ASCII character string, packed decimal, and zoned decimal data.

KSAM features: KSAM/3000 provides a number of features, including:

- Multiple keys. One primary key and up to 15 alternate keys may be specified for any KSAM data file. Each key is ordered in sequence by its value with no relation to other keys.
- Duplicate key values. While KSAM/3000 normally expects unique values for keys, some key types (such as zip codes) may logically contain duplicate values. You may optionally define these keys to allow non-unique data.
- Retrieval by generic key. You may elect to retrieve records based on a match with only the first part of the key. For example, all stock numbers for similar parts can be read if they share a common prefix.
- Retrieval by approximate match. Records may be retrieved based on their relation to a specified key value, such as all dates greater than 780615.
- Fixed or variable length data records. For added flexibility, both fixed and variable length record formats are allowed for KSAM files.

These features make KSAM an appropriate tool for applications where the variety of file access options is more important than flexibility in data base design.

DEL/3000

DEL/3000, the Data Entry Library, is a software package that simplifies the interface from the HP 2640 series of interactive display terminals to the HP 3000. It provides the capabilities for designing forms and maintaining files of formatted data on the HP 3000 computer system. DEL assists you in creating forms that are stored in a form file residing on the system and in writing programs that access both the predefined forms and terminals. These programs can be written in COBOL, FORTRAN, BASIC, and SPL. Typically, a program displays a form selected from a form file onto the terminal screen and allows you to enter data on the form image. There are also DEL procedures to edit the data, which you may optionally store on a data file in a format defined by the program. As shown in Figure 3-8, the form image serves as a visible guide or template for entering data. The form image is moved onto the screen through a program buffer (work area), and the data entered is transferred to the data file via this or another buffer.

DEL consists of an interactive form maintenance program and procedures for form access, terminal access, editing, and high level interface functions.

The interactive form maintenance program (FORMAINT)

- creates the forms and stores them in the form files
- displays, modifies, or deletes the forms
- lists all forms in a form file
- deletes the entire form file

Each form file contains one or more forms that are usually related in some way. Forms of 24 lines or less, when brought into terminal memory, are stored in one memory page and can be completely displayed on the terminal screen. Larger forms occupy additional memory pages that are automatically displayed in sequence as the forms data is entered or transmitted. In addition, sets of related forms can be chained together so they can be displayed in a given order.

Form-access procedures allow the program to

- open a form file
- locate any form in that file
- close the form file

These procedures allow only read access to the form in the form file.

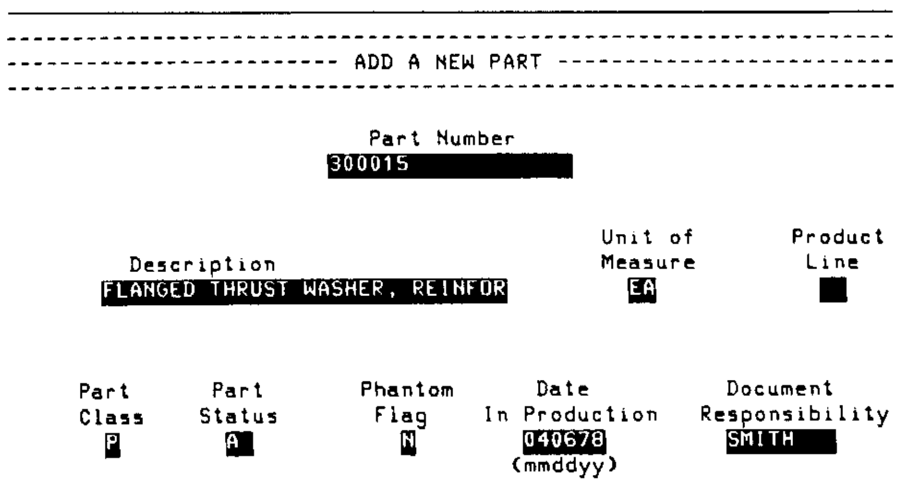
Terminal-access procedures provide read/write access to a terminal as an HP 3000 file. They permit the program to

- open a terminal as a file
- write output from the program buffer to the terminal
- read input from the terminal to this or another buffer
- request status information about the terminal
- close the terminal file

These functions allow the form to be called programmatically onto the terminal screen.

Editing procedures validate the general content of input to alphabetic, alphanumeric, and numeric fields on the forms. A program can call these procedures which check the appropriate input after it is entered and then indicate to the program whether the input data passed the edit check. For instance, if the program calls a procedure to verify that input to a field is alphabetic, but someone running that program enters numeric data, the editing procedure returns an error indication to the program. In addition to these procedures supplied with DEL, user-written editing procedures may be provided to interface between a program, the forms it displays, and the user at the terminal.

High-level interface procedures combine some of the form access and terminal-access operations described above.



```
----------------------------------------------------------------
------------------------ ADD A NEW PART ------------------------
----------------------------------------------------------------

                        Part Number
                        300015

                                              Unit of     Product
           Description                        Measure      Line
FLANGED THRUST WASHER, REINFOR                  EA

Part       Part       Phantom       Date            Document
Class      Status      Flag     In Production    Responsibility
P          A            N          040678          SMITH
                                  (mmddyy)
```

Figure 3-8. Sample DEL Formatted Screen

Interactive program development
In addition to its data management capabilities, the HP 3000 also possesses other powerful tools for application development. These include six programming languages, utilities, and a wide variety of debugging aids. The MPE segmenter, another important facility, is described in Chapter 2 under "Application Program Management."

Languages: Since different kinds of applications may require different approaches, the HP 3000 provides six programming languages to meet a wide range of needs:

- **COBOL–A business programming language whose syntax is quite similar to ordinary English.**
- **RPG–A business programming language designed primarily for complex report generation.**
- **FORTRAN–A programming language originally designed for execution of scientific formulas. It is now used extensively in business as well as the scientific community.**
- **BASIC (both an interpreter and a compiler) –A programming language designed for ease of learning and use by beginners. With the additional capabilities added since BASIC was first introduced, it is now used extensively in on-line business, education, and scientific environments.**
- **SPL–A programming language unique to the HP 3000 which combines the best features of high level and assembly languages. It is a block-structured language similar to ALGOL.**
- **APL–A terminal-oriented, high level language used primarily for numeric manipulations, free of detail and with a large set of functions and operators.**

These languages may be used concurrently in any combination by many individuals or, for multiple compilations, several languages may be incorporated within a single job or session. For a complete description of the features of these languages consult Section II of this manual, where each one is detailed in a separate reference sheet.

All of the compilers on an HP 3000 are invoked in the same general way. This reduces the necessary training, since learning to use one compiler is essentially learning to use them all. The HP 3000 also preserves your programming investment by incorporating subroutine call conventions in the machine's microcode. This enables programs in all languages, except APL, to call subroutines written earlier in other languages. A program written in COBOL, RPG, FORTRAN, BASIC (interpreted or compiled), or SPL can call subroutines written in COBOL, FORTRAN, BASIC (compiled), or SPL.

A program written in any of the available programming languages (except APL) may also manipulate an IMAGE data base, making the HP 3000 data base management capabilities accessible wherever they are needed.

All programs which run on an HP 3000 access data through the MPE file system. This makes the physical type of device on which the data resides transparent to the programmer. Therefore, no changes need to be made to a program to switch input or output from one device to another, e.g. from cards to an interactive terminal or from magnetic tape to disc. Programs used in batch jobs can also be used without modification in interactive sessions, and vice versa. In addition, files created by one language can be accessed by any other language. See Chapter 2 for more details on MPE and the file system.

Utilities: Program development on the HP 3000 is made easy through a powerful editor and other commonly-used utility routines.

EDIT, the HP 3000 text editor, allows you to create and manipulate files of upper- and lower-case ASCII characters with great ease. Lines, strings, and individual characters can be searched for, inserted, deleted, and replaced. The files to be edited may contain source language programs or text material such as reports.

You interact with EDIT through a set of editor commands that includes commands common to many other text editors throughout the computer industry. As illustrated in Figures 3-9 and 3-10, experienced users can write programs directly on the terminal, bypassing both coding sheets and punched cards.

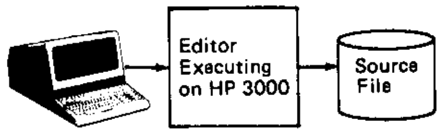


Figure 3-9. Creating a Source File On Your Terminal

After initiating a session with the HELLO command, you can use the editor to create a source file. Here is an example of the simplest method:

```
:EDITOR                                              ◀ Initiate Editor Execution

HP32201A.6.01 EDIT/3000 WED,DEC 7,1977, 2:08 PM      Put Editor in FORMAT=COBOL mode.
(C) HEWLETT-PACKARD CO. 1976                       ◀ Enter an ADD command to add lines of
/SET FORMAT=COBOL                                    source code.
/ADD
    1      $CONTROL USLINIT,MAP
    1.1     IDENTIFICATION DIVISION.                ◀ Space to COBOL column 8
                     .                                (which is Editor column 2)
                     .                                to enter COBOL statements.
     6.7         STOP RUN.
     6.8    //                                      ◀ Enter // or Control Y (Yᶜ)to stop adding lines.
 ...                                                  List the code and check it.
 /LIST ALL
     1      $CONTROL USLINIT,MAP
     1.1     IDENTIFICATION DIVISION.
                     .
                     .
/KEEP SOURCEX                                       ◀ In this case source code is saved in a file
/LIST ALL, OFFLINE                                    named SOURCEX. If you want a line
*** OFF LINE LISTING BEGUN. ***                       printer listing use the LIST command.
/EXIT
```

Figure 3-10. Using the HP 3000 Text Editor

Basic editing functions such as adding, locating, changing, deleting and listing lines are provided using single-word commands (which can also be abbreviated to a single, unique letter), whether the edits are applied to one line or many lines. Most functions operate either on specified line numbers or on lines containing specified character strings. Groups of lines may be moved or copied within a file or from one file to another.

EDIT also lets you write complex edit command sequences where editing is based upon conditions found within the text itself. In such command sequences the editor language assumes an ALGOL-like structure with the commands WHILE, NOT, and OR acting upon statements that can be simple or compound in nature.

Through the use of the Z::= and USE commands, you can store sequences of editor commands in the system and then invoke them at will to act upon text data in the editor work file. In this manner commonly-used sequences of editor commands, such as complex WHILE sequences, can be activated by way of a single USE command rather than by reentering the entire sequence.

A very powerful feature of EDIT/3000 is that it allows you to write procedures in COBOL, FORTRAN, or SPL to perform editing functions not otherwise possible with the standard repertoire of editor commands. You write, prepare, and store your procedures on disc and then invoke them with the PROCEDURE editor command. The editor passes the lines of text specified in the PROCEDURE command, one at a time, from the work file to the designated procedure which then manipulates them and passes them back to the editor. EDIT substitutes the modified lines for the original ones in the work file. After the procedure is finished, the editor resumes processing standard editor commands. This feature allows you to perform any editing function that can be done a line at a time.

SORT, the HP 3000 sort/merge program, allows you to sort records in a file into a prescribed order and to merge records by combining two or more previously sorted files into one sorted file. SORT can be used as a free-standing subsystem through a few simple commands in either an interactive session or a batch job. It can also be accessed by programs written in COBOL, FORTRAN, or SPL.

FCOPY, the HP 3000 file copier, may be used for all file copying operations on the HP 3000, including KSAM files. With FCOPY you can transfer files from disc, magnetic tape or any other valid input device to disc, magnetic tape or any other valid output device.

The Scientific Library and Compiler Library provide subroutines that perform the functions most frequently required in scientific applications.

For a more detailed explanation of all of the HP 3000 utilities, turn to Section II of this manual, where each product is described in a product reference sheet.

Debugging aids: The HP 3000 provides a wide variety of aids for debugging programs at many different levels. MPE commands such as SHOWJOB, LISTF, or SHOWOUT are available to inquire about the status of a job or a file. Compiler subsystem $CONTROL commands may be used to specify whether to print source code, object code, warning messages, or a symbol map. The $IF compiler subsystem command permits conditional compilation of portions of the program, allowing debugging statements to be inserted permanently in the code and compiled only when needed.

Two powerful capabilities available during execution of a program are Debug and Stack Dump.

The Debug facility is an intrinsic procedure which provides an interactive debugging facility to enable you to set breakpoints, display the contents of memory locations, modify memory locations containing data, display and/or modify the contents of registers, and so forth.

The Stack Dump facility is composed of an intrinsic that enables a program to selectively dump any part of the stack and a stack analysis mechanism that is activated when a program aborts. If the program is running interactively, an automatic call to the debug procedure is generated. If the program is running in a batch environment, part or all of the stack is dumped.

Batch processing
While on-line transaction processing is the design focus for the HP 3000, batch processing is just as easy and convenient. For a complete discussion of batch operation, see Chapter 2.

Chapter 4 Data communications

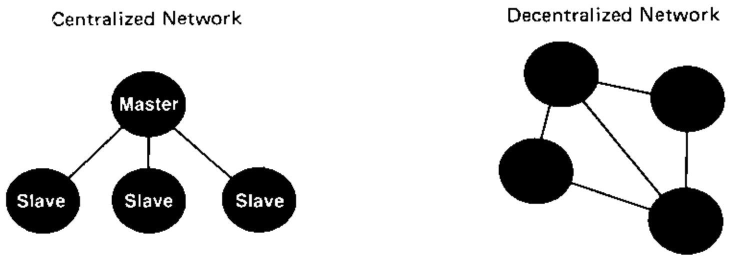
Distributed Systems Network
Fundamental to an understanding of the data communications capabilities of the HP 3000 is a basic knowledge of the Hewlett-Packard Distributed Systems Network. HP DSN consists of:

- An overall set of design objectives for the interconnection of various computer systems to form a network
- A basic philosophy of distributed data processing
- A set of products that are available today (Distributed Systems/3000, Distributed Systems/2026, and Distributed Systems/1000)

The architecture of HP DSN distributes both processing and control, while retaining sufficient flexibility to support hierarchical configurations and satellite processing for large, central mainframes via IBM-compatible remote job entry services (see Figures 4-1 and 4-2). All HP computer systems communicate with IBM 360 or 370 computers or any other mainframe computer supporting IBM remote job entry services.

Distributed Systems Network objectives: The objectives of HP DSN may be summarized as follows:

- System and terminal capabilities that free you from having to know communications protocols and network topology (configuration)
  - Remote file access
  - Remote device (peripheral or CPU) access
  - Remote data base access
  - Remote program to program communications
  - User access to any terminal in the network
  - Terminal access to any system in the network
- Interconnection of the Hewlett-Packard family of computing products
- Connection with non-HP equipment
- Distributed computing with local control and processing
- Sharing of data and system resources
- Network diagnosis and recovery
- Data and communication security



Centralized Network	Decentralized Network
▲ Economics of Scale	▲ Improved Response Time
▲ Corporate Standards/Control	▲ Local Access to Information
▲ Responsibility for Data Processing Costs	▲ Local Access to Computer Services
	▲ Lower Incremental Cost for System Growth

Figure 4-1: The Hewlett-Packard Distributed Systems Network Design Objectives: provide a framework for connecting HP computer systems with each other and with larger central host systems. The framework of HP DSN accepts a wide spectrum of network types, allowing you to optimize the trade-off between total centralization and total decentralization.

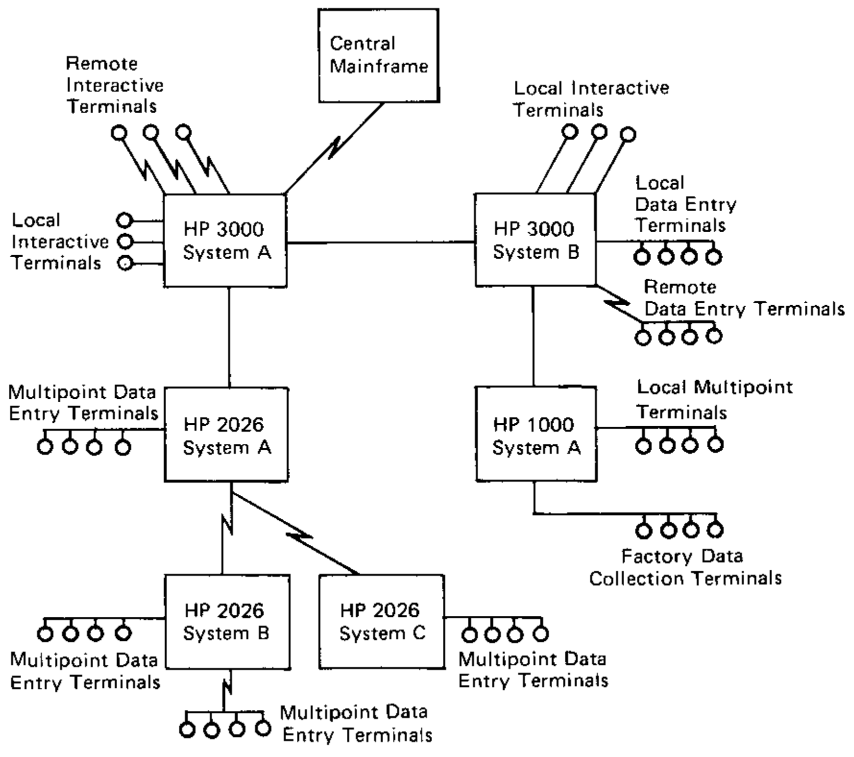

Central Mainframe
Remote Interactive Terminals
Local Interactive Terminals
Local Data Entry Terminals
Local Interactive Terminals
HP 3000 System A
HP 3000 System B
Remote Data Entry Terminals
Multipoint Data Entry Terminals
HP 2026 System A
HP 1000 System A
Local Multipoint Terminals
Factory Data Collection Terminals
HP 2026 System B
HP 2026 System C
Multipoint Data Entry Terminals
Multipoint Data Entry Terminals
Multipoint Data Entry Terminals

Figure 4-2: Sample Distributed Processing Network: HP DSN supports distributed processing and control as well as hierarchical configurations and satellite processing for large central mainframes.

Distributed Systems Network description: The Hewlett-Packard Distributed Systems Network is implemented as a number of functional layers in software (see Figure 4-3) with only the high level system services visible to you. The layered approach provides flexibility for additions and enhancements at each layer. Advantages of this approach include both stability and flexibility—stability because changes can be made to the internal layers without requiring you to alter application programs, and flexibility because the modular design of the network easily accommodates changes as a result of technological improvements.

The specific functions assigned to each layer are as follows:

Network access method: The network access method gives you the network capabilities at the application level via callable MPE intrinsics, system services, and specific language I/O constructs.

The services include:

- **Remote file access**
- **Remote data base access**
- **Remote command processing (virtual terminal access)**
- **Remote device access**
- **Remote program management**

Network manager: The network manager is the layer that is aware of the network topology and is responsible for managing network error recovery and the various topology dependent functions such as polling.

Message control protocol: This layer provides control functions, addressing information, message type, and other requirements to effect end-to-end transmission.

Communication line protocol: The communication line protocol is the "grammar" or protocol by which two or more systems can exchange information in an efficient and reliable manner.

Communication line controller: This consists of the hardware that connects to the communications line, and the software driver for this hardware interface.

Terminal communications
The terminal handling capabilities of the HP 3000 provide you with a number of alternatives to design the most cost-effective and functional communication links to terminals. Asynchronous or synchronous communication may take place over point-to-point or multipoint links. Links may use modems, or be hardwired, with several transmission speeds possible.

Point-to-point terminal communications: The Asynchronous Terminal Controller (ATC) allows the attachment of up to 16 asynchronous terminals (speeds to 2400 bps) to the HP 3000. Multiple controllers may be attached via the IOP bus. Terminal connections to the ATC are point-to-point, and may be hardwired or (optionally) modem links. Modems supported are Bell types 103A2, 103A3, 103J, 113B, 202S/T and Vadic VA 3400. The ATC also performs automatic speed sensing for dial-up modem connections. Terminals interfaced through the ATC can be configured to the operating system (MPE) as data entry terminals under your program control, or as log-on terminals accessing all the capabilities of the HP 3000. On the first ATC, the system console occupies one of the sixteen ports.

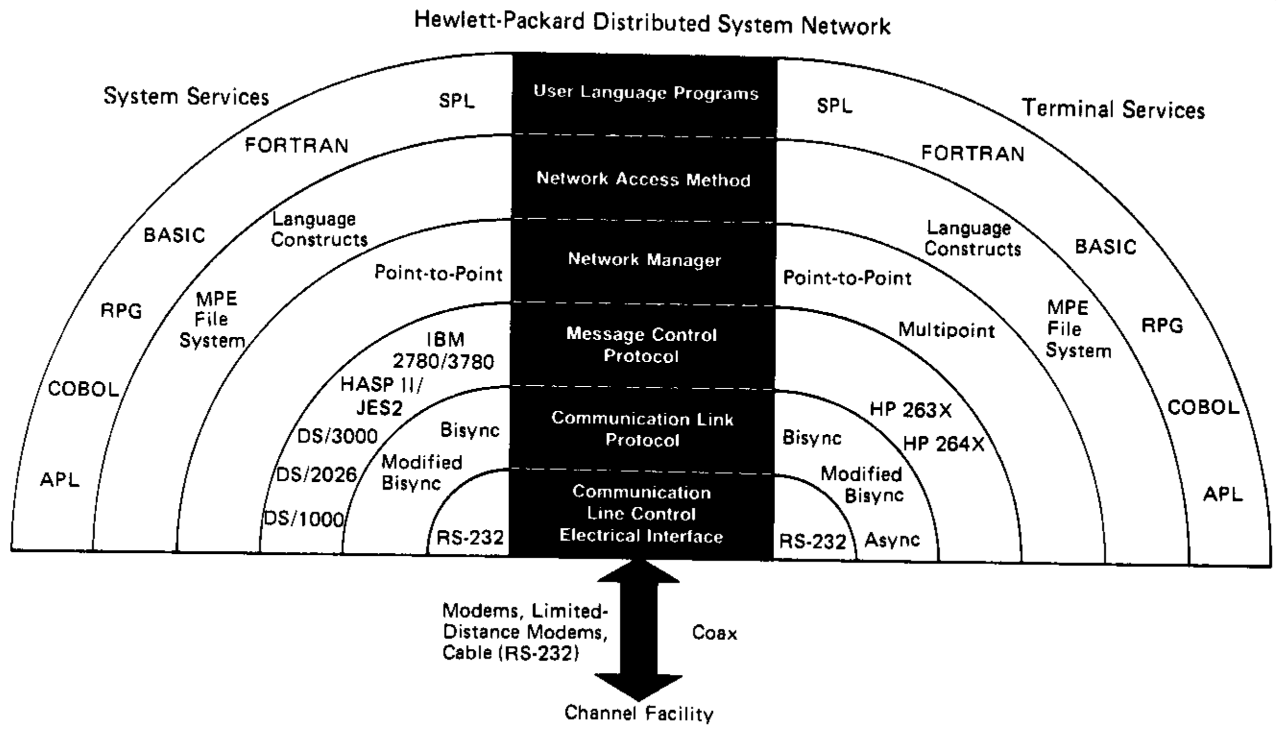


Figure 4-3: HP DSN Implementation: HP DSN is implemented as a series of software layers. Improvements can be made in any layer without affecting the highest level, your application programs.

Terminals on the ATC normally operate in character mode, except when accessed via the Data Entry Library (DEL/3000) when block mode is employed. Users who wish to access terminals in block mode directly (i.e., not using DEL/3000) must provide their own detection and correction facilities for transmission errors by calling operating system routines.

Multipoint terminal communications: Multipoint Terminal Software (MTS/3000) permits two-way, half duplex data transmission between the HP 3000 and multiple HP 2640 series multipoint terminals via a single communications line using binary synchronous protocol. The terminals may be connected to the computer by means of a modem (remote access) or may be hardwired to the computer (local access). With MTS/3000 you use MPE commands and file system intrinsics to communicate with the terminals. You may also initiate sessions from the terminals and thus access all the resources of the MPE operating system.

Terminal networks: Figure 4-4 is an illustration of a network of multipoint terminals. When terminals are connected by modems (remote access) communication is synchronous; hardwired terminals (local access) may be either synchronous or asynchronous. Up to 2000 feet (609.6 meters) of cable may separate individual hardwired terminals, however the first terminal must be within 50 feet (15.24 meters) of the system unless using the Asynchronous Repeater to extend the cable distance. Multiple lines of terminals may be attached to an HP 3000 computer system, with a maximum addressing limit of 255 terminals multidropped on these lines.

Data transmission: Terminals may transfer data at speeds up to 9600 bps. They may operate in either log-on (interactive) or data entry (page) mode. In either mode, you enter data into the terminal's memory using the cursor positioning capabilities, TAB key, and RETURN key. This data can then be edited as much as desired until the ENTER key is pressed, transferring the data to the computer. Variable length blocks of data may be transferred in this manner.

Power down bypass cable: A power down bypass cable is available for use with the MTS/3000. The cable, ordered with the multipoint terminal, enables the system to bypass a terminal which has no power, so that the remaining terminals in the daisy chain group are not affected.

Longer distance direct connections: The Asynchronous Repeater (AR) is a stand-alone device which converts standard RS232C communication signals to levels compatible with the HP 2640 series of terminals. It is useful in either an asynchronous multipoint or point-to-point environment.

By adding an AR in a multipoint environment, the first directly connected terminal on a line can be located up to 2000 feet (609.6 meters) from the computer. This removes the 50 foot (15.24 meters) limitation imposed when an RS232C direct connect interface is used. In addition, each AR can extend the maximum cable distance between individual multipoint terminals (or a group drop of terminals) by an additional 2000 feet (609.6 meters). Figure 4-5 shows examples of the possible AR uses in a multipoint terminal environment. Note that additional ARs may be added to further extend the maximum distance allowed.

One or more ARs may also be used in the point-to-point environment where terminals are attached to the system via the Asynchronous Terminal Controller.

Terminal peripheral devices: Terminals can be viewed as processors with several peripheral devices:

- **Keyboard**
- **Display**
- **Two cartridge tape units (optional)**
- **Printer (optional)**

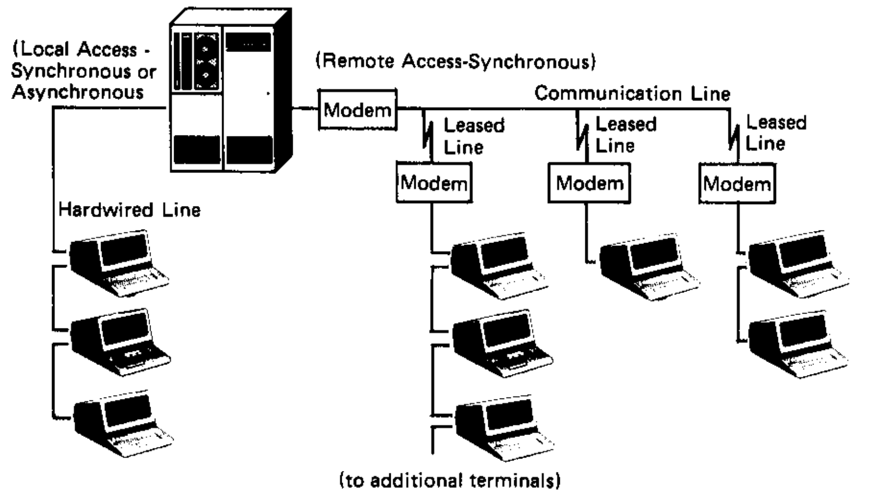

(Local Access - Synchronous or Asynchronous
(Remote Access-Synchronous)
Communication Line
Modem
Leased Line
Leased Line
Leased Line
Modem
Modem
Modem
Hardwired Line
(to additional terminals)

Figure 4-4. A Typical MTS/3000 Network

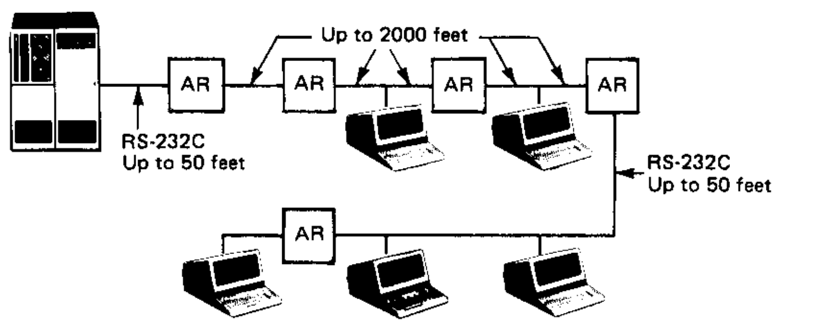


Figure 4-5. Use of the Asynchronous Repeater in Multipoint Terminal Environment

These peripheral devices can be controlled by transmitting the appropriate escape sequences to the terminal from your application program. The capabilities provided by the escape sequences include data transfers from the HP 3000 to a peripheral device, from a peripheral device to the HP 3000, and from one peripheral device to another within the same terminal. An escape sequence also exists to retrieve status information for a peripheral device.

These capabilities can be used with both point-to-point and multipoint communications. In multipoint transmission the peripherals share the terminal's multidrop line. In a point-to-point environment using the Asynchronous Terminal Controller (ATC) the peripheral shares the point-to-point line.

Computer-to-computer communications

Distributed Systems/3000: Distributed Systems/3000 (DS/3000), consisting of both hardware and software, is the HP 3000 implementation of HP DSN for communicating with Hewlett-Packard systems. It provides a complete set of network communications services so that programs, files, peripheral devices, and processing can be shared in a network. No knowledge of communications programming is required to use DS/3000. In fact, most DS/3000 capabilities are available to you with no programming effort once the network hardware and software are installed.

DS/3000 allows an HP 3000 to communicate with other HP 3000 systems and with HP 1000 and HP 2026 computer systems. Communication between these systems occurs in a bidirectional interleaved fashion using hardwired coaxial cables for HP 3000 to HP 1000 communications and modem lines for HP 3000 to HP 2026 communications. Communication between two HP 3000 systems may use either the hardwired or modem links with optional compression of data.

The HP 1000, with its Real-Time Executive (RTE) operating system, is a high-performance computer system designed for real-time computation and instrumentation applications. HP 1000s with RTE-III, RTE-IV, RTE-MII and RTE-MIII operating systems can be nodes of a DS/3000 network.

The HP 2026 is designed to meet the data entry, local file retrieval and data communications needs of companies with geographically dispersed locations. Data can be entered, stored, and retrieved locally and then communicated using batch transfer techniques to the larger, centrally located HP 3000 or IBM 360/370 compatible host system for further processing.

The HP 3000 to HP 3000 and HP 3000 to HP 1000 communications links give you the opportunity to select the appropriate blend of computing power for your specific requirements. After either of these communications links has been established, two commands enable you to make use of DS/3000 network services to other network nodes. The command DSLINE identifies the remote computer. Telephone number, identification list, maximum transmission buffer size, and the activation of data compression may also be specified. The other command, REMOTE, is used to direct locally entered commands to a remote computer identified in the DSLINE command. With the new commands, the following capabilities are available:

- **Remote command processing (RCP) allows users of a local system to employ exactly the same set of operating system commands as are available to a user at the remote system. All of the capabilities of both the local and the remote systems for program preparation, program management, subsystem access, and utility program use are available to you locally. In this mode, your local terminal becomes a virtual terminal to the remote system.**
- **Remote file access (RFA) gives you full access to the data files and peripheral devices on a remote system. To access a remote file or device, you need only note in your file declaration (to the operating system) that the resource is at the remote site.**
- **Remote data base access (RDBA) lets you access a remote IMAGE/3000 data base on a remote computer system on a transaction-by-transaction basis.**
- **Program-to-program communications allows your application programs being run on separate systems to interactively exchange data and control information. This capability has provisions for remote program initiation and termination, data exchange coordination, and dynamically variable master/slave relationships.**

3000/3000 and 3000/1000 communication links

Remote command processing: With remote command processing you can execute the entire set of MPE commands on a remote HP 3000 while connected to a local HP 3000 or a local HP 1000. You can also execute system level RTE operator commands on a remote HP 1000 while connected to a local HP 3000.

To execute a remote MPE or system level RTE operator command from a local HP 3000, enter the following at the local terminal:

```
:REMOTE command
```

where command is the desired MPE or RTE operator command in its format.

HP 3000 to HP 3000 communication: The following example shows how you issue commands to a remote system (also see Figure 4-6). The underlined text in this example is all you need to access the remote system.

```
:HELLO USER.ACCT          Local log on
:DSLINE RCPU              Designates the remote system
                          for remote processing
:REMOTE
#HELLO RUSER.RACCT        Log on to the remote system. You
#LISTF                    can now issue any HP 3000 command
#EDITOR                   to be executed on the remote computer.
#SHOWJOB
   •
   •
   •
#BYE                      Log off the remote HP 3000
```

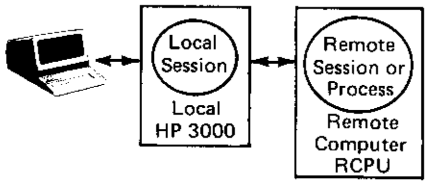


Figure 4-6. Remote Command Processing: allows you to issue commands to a remote system as if the local terminal were connected directly to the remote system.

Issuable HP 3000 to HP 3000 commands include all user commands, system supervisor commands, account manager commands, and system manager commands.

HP 3000 to HP 1000 communication: You can use the same REMOTE command from the HP 3000 to issue standard HP 1000 commands to be executed on a remote HP 1000 computer system. Issuable HP 3000 to HP 1000 commands include:

AB	Abort current batch program
BL	Set buffer limits
BR	Set break flag in named program's ID segment
DN	Declare I/O device unavailable
EQ	Examine status of I/O device
GO	Restart programs out of suspension
IT	Set time intervals for programs
LU	Examine/alter device logical unit assignments
OF	Turn programs off (abort)
ON	Turn programs on
PR	Change priority of programs
RU	Start a program immediately
RT	Release a program's disc tracks
SS	Suspend programs
ST	Examine the status of programs
TI	Display the current time
TM	Set the RTE real-time clock
TO	Examine/alter an I/O device's time-out parameter
UP	Declare I/O device available

To display the description and status of a particular I/O device on a remote HP 1000, for example, you could issue a remote EQ command from a local HP 3000.

```
:REMOTE EQ,2
```

In response to this command, the system would display information on the availability and current site of the specified device.

HP 1000 to HP 3000 communication The procedure for issuing a remote MPE command from a local HP 1000 has a slightly different format. After the remote HP 3000 is accessed and the communications link is established, you need only enter the following at the local HP 1000:

```
#COMMAND
```

where *command* is the desired MPE command in its normal format. For example, if you want to display the status of an input/output device on a remote HP 3000, you can issue a remote SHOWDEV command as follows:

```
#SHOWDEV 10
```

Replies generated at the remote HP 3000 are returned to you at your terminal on the local HP 1000.

Remote file access (RFA): With DS/3000 linking an HP 3000 with either another HP 3000 or an HP 1000, you have access to the files of the remote system, under control of the usual system file security. Peripheral devices connected to the remote system are also readily accessible, since they are treated by the system as files. Three methods exist by which you can access remote files—terminal access (3000/3000 link only), standard language input/output statements (3000/3000 link only), and remote file access intrinsics (3000/3000 and 3000/1000 links).

HP 3000 to HP 3000 RFA: With the terminal access capability, you merely issue a local MPE FILE command on your terminal, defining the desired device or file residing at the remote HP 3000 site. Included in the FILE command must be a remote device specification denoting the location of the desired file. The remote file may then be utilized on a record by record basis or as a complete file as though it resided on your local computer.

Access using standard language input or output statements permits local programs written in any language to define files and manipulate complete files, file records or devices on another remote HP 3000 as if the data or device were on the local system. All that is required is a FILE command which may be made external to the program or may be included in the program. This command specifies the location of the target device or file. Subsequently, the remote file or device may be utilized as if it resided on your local computer.

For example, you can locally run your application program (MYPROG) which accesses the remote file SOURCE1 with the following command sequence:

```
:HELLO USER.ACCT                              (log onto local HP 3000)

HP3000 / MPE III B.00.00.  TUE, MAY 23, 1978, 11:15 AM

:REMOTE HELLO RUSER,RACCT;DSLINE=CPUC         (log onto remote HP 3000)

DS LINE NUMBER=3

HP3000 / MPE III B.00.00.  TUE, MAY 23, 1978, 11:16 AM

:FILE SOURCE1;DEV=CPUC#DISC                   (define file SOURCE1 as being on
                                              remote system's disc)
:RUN MYPROG                                   (application program using file
                                              SOURCE1)
```

Local HP 3000 application programs may utilize the remote file access intrinsics shown in Table 4-1 to access standard MPE files and peripherals which reside on a remote HP 3000. In addition to accessing standard MPE files on a remote HP 3000, DS/3000 also allows you to access remote KSAM/3000 (Keyed Sequential Access Method) files using standard KSAM intrinsics from your local HP 3000.

HP 3000 to HP 1000 RFA/DEXEC: To access files on an HP 1000's moving-head disc, flexible disc, or tape mini-cartridge from an HP 3000, you must write an HP 3000 program that makes use of a set of intrinsics, called remote FMP, that allow access to the remote HP 1000 file system. These intrinsics have a direct correspondence to the standard HP 1000 RTE file management package (FMP). The remote FMP intrinsics to access HP 1000 files are listed in the first column of Table 4-1.

To access remote HP 1000 peripherals, such as line printers, magnetic tape units, and scientific instrumentation, you must write an HP 3000 program using remote Distributed Executive (DEXEC) calls. The remote DEXEC intrinsics to access HP 1000 I/O devices are:

DEXEC (1)	**Read a record**
DEXEC (2)	**Write a record**
DEXEC (3)	**I/O control**
DEXEC (10)	**Program schedule**
DEXEC (11)	**Time request**
DEXEC (12)	**Execution time**
DEXEC (13)	**I/O status**

These DEXEC calls may also be used to schedule or terminate programs, request system time, or inquire about the status of a program or an I/O device. These intrinsics have a direct correspondence to standard RTE EXEC calls.

To access HP 1000 files or peripherals, the local HP 3000 enters:

:HELLO USER.ACCT	**Local log on**
:DSLINE RCPU	**Specifies the remote computer (HP 1000)**
:RUN PROG	**This is a program that executes on the HP 3000 to access an HP 1000 disc and/or instrument using the intrinsics described above.**

Table 4-1. HP 3000/HP 1000 Remote File Access Intrinsics

INTRINSICS			Requested Action
1000/3000		3000/3000	
Local HP 3000 Call to Remote HP 1000	Local HP 1000 Call to Remote HP 3000	Local HP 3000 Call to Remote HP 3000	
DCRET			Creates a file.
DNAME	FRNAM	FRENAME	Renames a specified file.
DPURG			Removes a file and directory entry.
DOPEN	FOPEN	FOPEN	Opens a specified file.
DCLOS	FCLOS	FCLOSE	Closes a specified file.
DREAD	FREAD	FREAD	Transfers one record from a file (sequential file on HP 3000).
	FRDIR	FREADDIR	Reads a record from a direct access file.
	FRLAB	FREADLABEL	Reads a user file label.
DWRIT	FWRIT	FWRITE	Transfers one record to a file (sequential file on HP 3000).
	FWDIR	FWRITEDIR	Writes a record to a direct access file.
	FWLAB	FWRITELABEL	Writes a user's file label.
	FSTMD	FSETMODE	Changes file access mode.
	FLOCK	FLOCK	Dynamically locks a file.
	FUNLK	FUNLOCK	Dynamically unlocks a file.
	FUPDT	FUPDATE	Updates a record in a file.
DPOSN	FSPAC	FSPACE	Positions a file.
DAPOS			Positions a file to a known record.
	FPOIN	FPOINT	Resets pointer for sequential file.
	FRDSK	FREADSEEK	Prepares for reading a direct access file.
DCONT			Sends control request to peripheral device identified as a type 0 file.
	FCNTL	FCONTROL	Performs control of file or terminal device.
DWIND			Resets file to first record.
DLOCF	FINFO	FGETINFO	Returns file status.
	FCHEK	FCHECK	Requests details on file I/O errors.
DSTAT			Returns 250 bytes (125 words) of disc directory.
	FRLAT	FRELATE	Determines if file pair is interactive (requires human intervention for all input operations), or duplicative (echoes all input operations to a display device without intervention by the operating system software), or both.

HP 1000 to HP 3000 RFA: Local HP 1000 application programs may use the remote file access intrinsics in Table 4-1 to define, control, and access disc files and/or devices on a remote HP 3000.

Remote Data Base Access (RDBA): The remote data base access feature of DS/3000 gives you the capability of direct and indirect access of data bases on remote computer systems. Using the IMAGE and QUERY data base inquiry facilities of the remote system you can locate, report, and update data values directly in remote IMAGE/3000 data bases and indirectly in remote IMAGE/1000 data bases.

Direct data base access is a feature of the HP 3000 to HP 3000 communications link only. Using this method, you can automatically interrogate data bases on remote HP 3000 computer systems using remote QUERY interactive commands. Your local application programs can also contain standard IMAGE calls which retrieve and manipulate information in the remote data base. Intervention from a remote program is not required.

There are two methods by which you can directly access a remote data base. The first method requires you to establish a communications line and a remote session and enter a FILE equation for each remote data base. The FILE equation specifies which data base is to be accessed on which remote system and device. A local IMAGE application program can then be run to access the remote data base. The second method requires that a special file called the data base access file (DBA file) be created. This file provides IMAGE with the necessary information to establish a communications link and a remote session. It also specifies the remote data base file name so that the necessary IMAGE intrinsics can be executed on the remote computer. The following example illustrates this method (also see Figure 4-7). The DBA file is built by the HP 3000 text editor, EDIT/3000. It is named DBAX and contains:

```
REC1 FILE DBX;DEV=RCPU #DISC     Record 1 specifies the location of the remote data base.
REC2 DSLINE RCPU                 Record 2 specifies the remote computer on which data base resides.
REC3 USER.ACCT,GROUP=HELLO RUSER.ACCT
 •
 •                               Records 3 through n specify which local user, group and account may
 •                               access which user, group, and account on the remote computer.
RECN
```

As a local HP 3000 user, you simply type the following to access the remote data base DBX:

```
:HELLO USER.ACCT   Local log on
:RUN PROGX         PROGX is a user-written IMAGE application that opens DBAX,
                   automatically establishes communication with the remote HP 3000,
                   and accesses the remote data base.
```

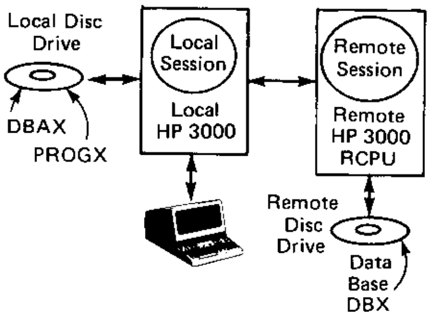


Figure 4-7. Direct Remote Data Base Access Between HP 3000s: can be implemented either by entering FILE equations or creating data base access files. Here a data base access file DBAX has been created and is used by PROGX, an IMAGE program, to access the remote data base DBX.

Indirect data base access is available on both the 3000/3000 and the 3000/1000 communications links, using the program-to-program capability. By first initiating a program on your local system, you can then run a remote program which accesses and/or updates the remote data base. The remote program then passes the requested information back to your local program.

Program-to-program communications (PTOPC): The DS/3000 program-to-program communications capability gives you the ability to write application programs using a set of nine intrinsics. These intrinsics make it possible for two or more user programs residing on different computer systems to exchange data and control information directly and efficiently with one another. The intrinsics, called PTOPC intrinsics, are directly callable by SPL, FORTRAN, BASIC, and COBOL programs. The nature of any two programs communicating with one another in this manner is not symmetrical. One of them (referred to as the master program) is always in control and is the one that initiates all activity between the two programs. The other (referred to as the slave program) always responds to requests received from the master (see Figure 4-8). Those intrinsics used within a master program are summarized in Table 4-2 and those used in a slave program are summarized in Table 4-3. The master program opens the data link, initiates the slave program, and is always in control. The slave program merely responds to requests received from the master program, either to accept or reject the master program requests. This master-slave relationship is dynamic in that it is determined solely by the design of your application program. Each computer may have master and slave programs active simultaneously, depending on the needs of the specific application.

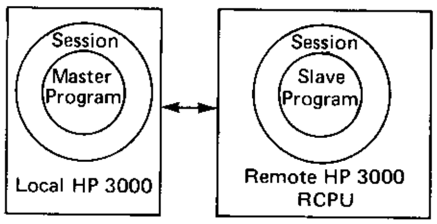


Figure 4-8. Program-to-Program Communication: allows user programs on different systems to execute and exchange data in a coordinated manner. One of the programs is the master and is always in control. The slave program responds to the master's requests.

Table 4-2. DS/3000 Master Program-to-Program Intrinsics

Intrinsic Name	Function
POPEN	Initiates and activates a slave process in a remote HP 3000 and initiated program-to-program communication with the slave program.
PREAD	Sends a read request to the remote slave program asking the slave to send a block of data back to the master.
PWRITE	Sends a block of data to the remote slave program.
PCONTROL	Transmits a tag field (containing user-defined control information) to the remote slave program and receives a tag field back from the slave.
PCLOSE	Terminates (kills) the remote slave process.
PCHECK	Returns an integer code specifying the completion status of the most recently executed master program-to-program intrinsic. (HP 3000 only.)

Table 4-3. DS/3000 Slave Program-to-Program Intrinsics

Intrinsic Name	Function
GET	Receives the next request from the remote master program.
ACCEPT	Accepts (and completes) the request received by the preceding GET intrinsic call.
REJECT	Rejects the request received by the preceding GET in-intrinsic call.
PCHECK	Returns an integer code specifying the completion status of the most recently executed slave program-to-program intrinsic. (HP 3000 only.)

HP 2026 to HP 3000 communications link

To enhance the remote processing functions and communication between a remote HP 2026 Data Entry/Communications system and an HP 3000 host computer, two basic capabilities are provided by DS/3000:

- Remote command processing from the HP 2026 console
- Point-to-point file transfer

The DS/3000 operation in no way alters the standard HP 2026 to HP 2026 communications features or the HP 2026 remote job entry capabilities to large central computers.

Remote command processing: All HP 3000 commands can be executed remotely from the system console of the HP 2026 in an interactive mode. After the remote data link is established by the HP 2026, the procedure for entering the MPE commands at the HP 2026 system console is as follows:

```
#command
```

where *command* is the desired MPE command in its normal form. For example, to display the names of all disc files residing in the remote log-on group and account of the remote HP 3000, you could issue a remote LISTF command from the local HP 2026 system console as follows:

```
#LISTF
```

All HP 3000 terminal capabilities available to a DS/3000 remote session are available from the HP 2026 console, including all user commands, system supervisor commands, account manager commands, and system manager commands.

Point-to-point file transfer: HP 2026 users located at remote terminals can create files to be sent to the host HP 3000 and place their names in a queue file until batch transmission to the HP 3000 is initiated from the HP 2026 system console. The system operator at the HP 2026 system console can specify additional files for transfer after the 2026/3000 link is established.

Files residing on the HP 3000 can be transmitted to the HP 2026 at the request of the system operator at the HP 2026 system console. Requests can be made for specific files, or all files named in the HP 3000 transfer queue may be sent to the HP 2026 after the HP 2026 to HP 3000 link is established. The queued files could, for example, be report listings from various operations performed on the HP 3000. Thus, multiple applications systems on the HP 3000 can send information to the HP 2026 by placing their report listings in disc files and these disc file names in the queue for transfer to the HP 2026.

Procedures may also be initiated in advance by the HP 2026 system operator to permit the HP 3000 to retrieve all of the HP 2026 queued files and send any queued HP 3000 files to the HP 2026 on request.

Figure 4-9 gives a complete list of HP 2026 to HP 3000 functions. All of the commands that initiate file transfer may specify that interactive mode is to be entered when the transfer is completed instead of dropping the communications link.

RECV	Receive an operator-specified file from an HP 3000
ARECV	Receive files queued on the HP 3000. Drop link when done, then set up for another call
QSEND	Send queued files to an HP 3000. Wait for more to be queued when done
MSEND	Send operator specified and/or queued files to HP 3000. Drop link when done
MS2AR	Send operator-specified and/or queued files to HP 3000. Receive queued files from the HP 3000. Drop link when done, then revert to ARECV
INTER	Use HP 2026 console as a remote virtual HP 3000 terminal

Figure 4-9. DS/2026 Functions: Distributed Systems/2026 provides for file transfer between HP 2026 and HP 3000 systems, and for the HP 2026 console to act as a virtual HP 3000 terminal for remote command processing.

So that the HP 2026 operator does not have to repeatedly type the (generally constant) log-on information, the HP 3000 HELLO command can be stored in a disc file and used automatically. This default can be overridden on any use of the system.

Summary of DS/3000 communications link features

With the powerful remote communications capabilities provided by these three HP computer systems, you can configure your computer systems network to coordinate and maximize the efforts and resources of the total organization. With HP 3000 computer systems available for batch and transaction-oriented data processing at remote or central sites, HP 1000 systems for dedicated applications on the factory floor or in the lab, and HP 2026 systems for remote data entry to a central EDP installation, DS/3000 provides you with all of the features necessary to make this distributed processing concept a reality.

- Virtual terminal capability to remote HP 3000 systems
- Simultaneous local and remote processing with HP 1000 and HP 3000 systems
- Intersystem data transfer
- Bidirectional interleaved communications
- Switched, leased, and hardwired lines
- High data throughput
- Data compression option
- Peripheral sharing
- Network design flexibility
- Network control and accounting
- Network security
- Network reliability

Virtual terminal capability to remote HP 3000 systems: This feature allows terminals physically connected to one computer system to operate logically as though they were connected to a remote HP 3000. Virtual terminal capability to remote HP 3000 systems is available to HP 2026 system consoles and to all terminals connected to HP 1000 or HP 3000 systems in an HP Distributed Systems Network. It provides access to all the hardware and software resources of any HP 3000 in the entire network.

Simultaneous local and remote processing: Within each HP 3000, HP 2026 or HP 1000 system in an HP Distributed Systems Network, a wide variety of processing can be in progress simultaneously, including local and remote batch jobs, local and remote transaction processing, local and remote interactive problem solving, intersystem program-to-program communication, and job entry.

Intersystem data transfer: Within an HP distributed Systems Network you can transfer data from one system to another. The remote file access capability of the 3000/3000 link makes it easy for a user on one HP 3000 system to access the files and devices of any other directly connected HP 3000 system in the network. The DS/3000 program-to-program communications capability of both the 1000/3000 and 3000/3000 links can also be used for performing high-speed intersystem block data transfers. In the 2026/3000 communications link, the point-to-point file transfer capability allows the HP 2026 and HP 3000 to send files to and accept files from each other.

Bidirectional interleaving: Each physical communications line in an HP Distributed Systems Network can be used concurrently by many different batch and interactive applications originating from either end of the line. You can also obtain exclusive access to a specific line, if necessary, given the proper security codes. DS/3000 automatically multiplexes and demultiplexes all data being transferred over each line, so that in interleaved usage, each application is essentially unaware that anyone else is using the same line.

Switched, leased, and hardwired lines: DS/3000 offers a choice of switched, leased, or hardwired lines in its 3000/3000 communications link and allows you to mix these line types freely throughout your network. You need not differentiate between links in your programs. This is handled automatically and transparently by DS/3000, so you can choose the most economical type of line for each particular link. The HP 2026 communicates with the HP 3000 via modems over switched or leased lines. The HP 1000 and HP 3000 communicate over a hardwired connection consisting of a coaxial cable available in lengths up to 2000 feet. Additional information on switched networks and leased line service can be found in the HP 3000 Guide to Synchronous Modems. (See Appendix F.)

High data throughput: Using a hardwired line (coaxial cable) on either the 1000/3000 or 3000/3000 link, hardware line speeds up to 2.5 million bits per second can be achieved on lines up to 1000 feet in length. Using common carrier facilities for 3000/3000 and 2026/3000 communication, data can be transferred at a rate of up to 9600 bits per second depending upon line conditioning and choice of modem. With the 2026/3000 link, each HP 2026 system console has exclusive access to a switched or leased line to the HP 3000. In the 3000/3000 link, when an application requires the maximum possible data throughput over a line, DS/3000 allows you to obtain exclusive access to a line, given the proper security codes. In all cases, the effective data transfer rate is a function of the application and total system/network workload.

Peripheral sharing: With DS/3000 several systems can share the use of expensive peripherals via communications lines, thus achieving a significant cost savings without sacrificing capability. The peripherals appear to you just as if they were on your local system.

Network design flexibility: DS/3000 offers you a full choice of network configurations with the ability to connect individual systems together in rings, stars, strings, or combinations of all three. DS/3000 has also been designed so that the master/slave relationships between any two HP 3000 computers and between HP 1000 and HP 3000 systems in the network are truly dynamic. Both of the systems in a link can alternate freely from master to slave status or operate in both modes simultaneously.

Network control and accounting: With DS/3000 you can have a computer network that distributes computer power to the locations where it is needed while, at the same time, maintaining control at a central location. The remote command processing feature allows a single system manager to control all satellite HP 3000 systems from the central EDP facility. The system manager may assign different capabilities on various systems to any individual user. Resource usage is automatically logged for CPU time, disc space, and connect time for all users on a system, whether remote or local.

Network security: Within an HP Distributed Systems Network the accounting structure and file security features of the operating systems provide protection against unauthorized use of HP 3000 and HP 1000 systems and their disc files. For example, to initiate a remote HP 3000 session from an HP 1000, HP 2026, or another HP 3000, you must supply the proper ID sequences to log on to the remote system, as well as on to the local system. Passwords can be attached to user names, account names, group names, and disc file names to provide a multilevel security scheme within each HP 3000 in the network. In addition, the file systems can be used to attach access levels to individual disc files, thus strictly limiting read, write, or read/write access to a particular subset of users. The HP 1000 and HP 2026 operating systems provide similar security code protection of disc files from unauthorized users.

Network reliability: DS/3000 uses cyclic redundancy check (CRC) characters to ensure that each particular block of data is transferred successfully from one system to another. If this CRC character indicates that an error has occurred, the block is automatically retransmitted. If an error persists through the maximum number of retransmissions, DS/3000 considers the error to be irrecoverable, discontinues communication, and reports the condition. The HP 3000's fault control memory provides an added measure of reliability, detecting and automatically correcting single-bit errors. In addition, DS/3000 contains specific procedures that you can execute to determine if your network configuration is working properly.

HP 3000 communications with IBM compatible systems

Distributed networks of HP computer systems can communicate with IBM host computers or other computer systems supporting IBM remote job entry services. The HP 3000 emulates a full function HASP II or JES2 work station with its Multileaving Remote Job Entry (MRJE/3000) program. For central systems whose operating systems do not support the more powerful multileaving packages, the HP 3000 may emulate an IBM 2780 or 3780 terminal with the RJE/3000 program. RJE takes place concurrently with other HP 3000 processing. This allows you to select the appropriate blend of remote job entry and local computing power to achieve your information processing goals, while making best use of the strengths of the HP 3000 and the host system. For example, you can maintain locally used files (data bases) on the HP 3000 and periodically transmit them to an IBM 370 for batch processing.

MRJE/3000: The Multileaving Remote Job Entry software (MRJE/3000), which runs under the control of MPE, gives all terminal users on a local HP 3000 simultaneous batch access to any remotely connected host computer system utilizing a HASP II or JES2 job entry system. Input can be entered from and output directed to any and all HP 3000 supported peripheral devices or files.

MRJE features: MRJE/3000 provides a number of significant features:

- **Full function HASP II or JES2 work station emulation**
- **Runs in a full multiprogramming environment**
- **Flexible, easy to use commands for job submission and status inquiry**
- **Accessible from both interactive terminals and traditional batch devices**
- **User specified job input and output from or to any peripheral device or file**
- **Local job entry capability available to multiple users simultaneously**
- **Supports an operator console, as well as up to 7 printers, 7 card readers, and 7 card punches interleaved on the same communications line**
- **Supports multiple hosts and/or multiple lines to a single host**
- **Modem communications up to 9600 bits per second for each MRJE line (switched network or leased line)**

Simultaneous transmission: With interactive terminals and batch devices, MRJE job transmission occurs concurrently with other HP 3000 activities. For instance, output peripherals are not idle while the job input phase is being completed. Instead they can be used as soon as the host system generates output. Similarly input peripherals can be used for batch job submission to the host even while the host is outputting processed data.

Submitted jobs on-line or off-line: The MRJE emulation package consists of two functional modules. One module allows you to submit jobs, display job status, and define characteristics of the work station. You may submit MRJE jobs whether or not an actual connection exists between the local and remote systems. Jobs submitted off-line are spooled and automatically transmitted when the connection is made.

The second module permits the MPE console operator to establish the MRJE communications connection with the host system, after which all user jobs (both spooled and newly initiated) are automatically transmitted to the host system for processing. Output is then directed from the host to the proper HP 3000 peripheral device or file without further intervention. Also, while the communications line is connected, the MRJE manager can enter host console commands interactively from any local HP 3000 terminal, permitting him to monitor and control job activity.

Automatic job routing: A significant feature of MRJE/3000 is its automatic job routing capability, which allows you to specify a particular output device or file when you submit the job. Output received from the host is then automatically directed to the specified device. If no output destination has been indicated, job output is routed to the default device designated by the MRJE manager.

RJE/3000: The 2780/3780 Emulator Subsystem (RJE/3000), which runs under the control of MPE, makes the HP 3000 appear to the remote processor as either an IBM 2780 or 3780 data transmission terminal. The emulator is more flexible than the IBM terminals in that it allows you to use a greater variety of input/output devices, including disc and magnetic tape.

The remote processor can be any of the following:

- **Any computer which supports the IBM 2780 or 3780 data transmission terminals**
- **An IBM 2780 or 3780**
- **Another HP 3000 which is also using RJE/3000**

Provided that the HP 3000 has more than one synchronous communications controller, several people may use the emulator concurrently. The number of concurrent users is limited by the number of synchronous communications controllers which are available. Prior to invoking the emulator, you specify which synchronous communications controller you wish to use by issuing an MPE FILE command.

RJE/3000 features:

- **Runs in a full multiprogramming environment**
- **Input/output access to disc and magnetic tape in addition to card reader, card punches, and line printers**
- **Supports multiple hosts and/or multiple lines to a single host**
- **Modem communications up to 9600 bits per second (switched network or leased line)**
- **Provides a set of ten emulator commands to control the sequence of input/output processing**

RJE/3000 capabilities: RJE/3000 provides all of the capabilities of the IBM 2780 or 3780 terminals with the following exceptions:

- IBM 2780 six-bit transcode is not supported.
- When emulating an IBM 2780, RJE/3000 performs short-record truncation without the user having to supply EM control characters in the data.
- When emulating an IBM 2780, RJE/3000 can perform blank field compression.
- When emulating either an IBM 2780 or 3780, RJE/3000 provides input/output access to a greater variety of peripheral devices including disc and magnetic tape.
- When emulating an IBM 2780, RJE/3000 can block more than 7 records.
- Reverse interrupt capability is not supported.

HP 3000 Computer System Reference Sheets

The reference sheets which follow contain the specifications of the software and hardware which comprise the HP 3000 computer systems. To locate a particular topic, just turn to the pages marked with the corresponding tab indicated below.

MPE III

Multiprogramming Executive Operating System

model 32002B

The Multiprogramming Executive, MPE III, is the general purpose, disc-based operating system which supervises the processing of user programs on HP 3000 computer systems. Designed to take advantage of HP 3000 features such as virtual memory and a stack architecture implementing separation of code and data, MPE allows multiple users to concurrently access all of the computer system resources.

Features

- **Multiprogramming: Concurrent transaction processing, timesharing, and batch processing**
- **Virtual memory**
- **Stack architecture: Separation of code and data, variable length segmentation, and data stacks**
- **Concurrent multilingual capability: COBOL, RPG, FORTRAN, BASIC, APL, and SPL**
- **File system with file backup and security**
- **System security and complete accounting of resources**
- **Friendly, powerful command language, including user-defined commands, conditional job control, on-line HELP facility, and meaningful error messages**
- **Device and file independence**
- **Input/output conveniences: Spooling of input and output, private disc volumes, and tape labels**
- **Complete, automatic terminal management, local and remote**
- **Power fail/auto restart**

The HP 3000 operating under MPE is extremely versatile, performing transaction processing, timesharing, and batch processing concurrently. Programs which support multiple terminals are both easy to create and easy to manage using MPE's full set of terminal management functions. Since the system controls device status monitoring, padding characters, buffer management, and error handling, the application program simply makes regular file read/write requests. MPE also offers you two ways to construct a transaction-oriented application: central control, in which an application program manages multiple terminals; and individual control, giving each terminal user the ability to run separate programs.

MPE relieves you of many program control, input/output, and other housekeeping responsibilities by monitoring and controlling program input, compilation, run preparation, loading, execution, and output. MPE also regulates the order in which programs are executed and allocates the hardware and software resources they require. It does this by providing an account/user/group structure, a command interpreter, a file system, a scheduling mechanism for the CPU, and a memory allocation manager.

Total system security allows you to operate in an environment protected from interference or illegal access from other users. Program protection (in memory) is provided by the hardware, with access to the system controlled through a set of passwords on account, user and group names at log-on. File security is based on a series of file passwords (called lockwords in MPE) and hierarchical access restrictions that allow you to specify the degree of security.

HP 3000 system resources such as main memory, the central processor, and peripheral channels and devices are dynamically allocated to each program as needed. Typically, each user is independent of all others on the system. For those cases where you need to interact and cooperate with others on joint efforts, however, MPE provides capabilities for sharing information.

All input/output (I/O) to physical devices (peripherals) is handled by the MPE I/O system. It receives I/O requests from other system software, queues them if necessary, and performs the transfer of data to or from the device. Because all I/O devices are treated by MPE as files, you can write programs without immediate concern for the physical source of input or destination of output and run them in either batch or interactive mode without changing the names of the files they reference.

Asynchronous terminal communications are automatically handled by MPE. Synchronous data communications to terminals, other Hewlett-Packard computer systems (including additional HP 3000s) and non-HP computers is achieved by optional software subsystems which extend the MPE data communications capabilities.

MPE also includes many tools for overall management and control of the system including a power fail/automatic restart routine. In addition, system managers and system supervisors can access an accounting facility, logging facility, and a SYSDUMP program which is used to backup system software and user files.

Specifications:

The HP 3000 Multiprogramming Executive consists of these major components:

- Configurator
- Initiator
- System console manager
- Command interpreter
- File management system
- Input/Output system
- Virtual memory manager
- Disc space manager
- Private volumes facility
- Serial disc interface
- Tape labels facility
- Spooling facility
- Job/session scheduler
- Process dispatcher
- Segmenter
- Loader
- User trap manager
- Utility intrinsics
- Accounting facility
- Logging facility
- Backup/restore facility
- Power fail/auto restart

Real memory supported: 192kb to 2Mb

Virtual memory supported: up to 8Mb

Maximum length of spoolfiles: 262,144kb

Maximum number of spoolfiles: 300

Maximum number of I/O controllers (DRT entries): 125

Maximum number of processes: 255

Maximum data segment length: 64kb

Maximum code segment length: 32kb

Maximum number of code segments per program: 63

Maximum number of files open per program: 253

Maximum file size: 536,871kb

Maximum number of extents per file: 32

Documentation

For further technical information, consult the following Hewlett-Packard manuals:

MPE Commands Reference Manual (30000-90009)

MPE Intrinsics Reference Manual (30000-90010)

MPE Segmenter Reference Manual (30000-90011)

MPE Debug/Stack Dump Reference Manual (30000-90012)

MPE System Utilities Manual (30000-90044)

Error Messages and Recovery Manual (30000-90015)

Index to MPE Reference Documents (30000-90045)

System Manager/System Supervisor Manual (30000-90014)

Console Operator's Guide (30000-90013)

HP 3000 Software Pocket Guide (30000-90049)

Training

Hewlett-Packard offers the following training courses to give you a working knowledge of the MPE operating system:

HP 3000–A Comprehensive Introduction, 5-days, #22801A at HP Technical Center, #22815A on-site.

HP 3000–System Management and Operation, 5-days, #22802A at HP Technical Center, #22816A on-site.

HP 3000–Special Capabilities, 5-days, #22805A at HP Technical Center.

HP 3000–SPL File System Introduction, 5-days, #22804A at HP Technical Center.

IBM System/3–to–HP 3000 Conversion, 3-days, #22817T at HP Technical Center, #22817A on-site.

HP 3000CX or Series I–to–HP 3000 Conversion, 1-day, #22818A on-site.

Also available is HP 3000 Software Consulting, #22825A, an on-site consultation service to help in applying Hewlett-Packard software to your specific applications problems.

The HP 3000 Fundamental Operating Software consists of a set of utility programs (EDIT/3000, FCOPY/3000 and SORT/3000), a subroutine library (Compiler Library) and the Hewlett-Packard Systems Programming Language.

EDIT/3000

The EDIT/3000 text editor permits you to create and manipulate files of any ASCII characters. Lines, strings and characters can be inserted, deleted, replaced, searched for, etc. The files to be edited can be source language programs, such as COBOL, RPG, or SPL, or text material, such as reports.

The command language is designed to include those commands that normally exist in all editors (e.g., DELETE, REPLACE, INSERT), as well as commands to write complex command sequences, where editing is based on conditions found within the text itself. For example, you can:

- Change occurrences of a character string
- Call user-written procedures for modifying or processing text
- Execute pre-stored EDIT/3000 commands
- Use a nested, interactive loop facility for repetitive editing
- Perform multiple-line deletions, insertions, moves and replacements
- Use Boolean logic for conditional editing
- Display before editing, display after editing, do not display
- Set and reset margins during operation
- Store data in a hold file to be duplicated into another portion of the work file
- Selectively concatenate portions of files
- Easily modify complex text using a line by line template display (MODIFY command)

For further technical information, consult the EDIT/3000 Reference Manual (03000-90012).

FCOPY/3000

FCOPY/3000 is a program used for general file copying operations. In addition to this basic capability, it can translate character code, dump files in a user readable form, verify a copy operation, select a subset of a copied file, and ignore a specified number of read errors from a source file. These functions can be performed as a single operation or as multiple operations within a single access to FCOPY.

Character code translation gives you the ability to convert EBCDIC and BCD source files to ASCII and vice versa.

Dump formatting allows for the formatting of octal, hexadecimal, and character dumps. When you specify the dump formats and title, the utility automatically establishes the dump format according to the output device type.

The copy verification capability allows you to compare two files. When a compare error is found, you are given both the record and the word or byte number where it occurred.

Through the subset option, you can select a portion of a file based on field content, or number of records starting with a given record, or all records contained between two record numbers.

FCOPY can copy files from any supported input device to any supported output device. When using this utility to copy files from a tape cassette on one terminal to a tape cassette on another terminal, you must first copy the files to an intermediate I/O device (such as a disc).

For further technical information, consult the FCOPY/3000 Reference Manual (03000-90064).

SORT/3000

SORT/3000 performs two activities:

- Orders records in a file according to a given key sequence
- Merges a number of sequenced files into a single sequenced file

This permits you to arrange large quantities of records (a file) into a prescribed order. Each record consists of a series of data fields which describe one "item" of information. Sorting is based on keys (values of one or more data fields). Merging forms one sorted sequence of records by combining one or more previously sorted sequences of records.

Additional SORT features include the following:

- Ascending and descending sort by keys can both be performed.
- Keys can be contiguous, separated, or overlapping.
- Keys may be of multiple data types.
- Record size is unrestricted and may be fixed or variable length.
- Input and output media may be of various types (e.g., disc files, magnetic tapes, cards, printer output, etc.).
- The sorted output can be chosen from sequenced records, key fields, record numbers or record numbers plus key fields.
- User specified routines may be used for key compare, pre-processing, and post processing.
- Any sorted files can be merged.

For further technical information, consult the SORT/3000 Reference Manual (32214-90001).

Compiler Library

The HP 3000 Compiler Library is a set of subroutines that provides many operations commonly needed when programming in COBOL, RPG, FORTRAN, SPL, and BASIC. These operations include:

- Matrix operations
- Complex arithmetic
- Trigonometric functions
- Mathematical functions
- Numeric conversions
- Utility functions

In addition, the Compiler Library includes a formatter that simplifies input/output operations for FORTRAN programs and makes it unnecessary to specify precise machine operations; you only specify the format of the data, a list of variables, and a device or file.

For further technical information, consult the Compiler Library Reference Manual (30000-90028).

SPL/3000 Compiler

SPL, the Systems Programming Language for the HP 3000 computer systems, is detailed with the other programming languages later in this section.

HEWLETT PACKARD

Scientific Library

model 32205B

The Scientific Library is a collection of procedures that perform the functions required most often in scientific applications. These procedures may be called directly by user programs written in FORTRAN or SPL; most of these functions may also be accessed from BASIC.

Features

- Error function of a single-precision or extended-precision number
- Complementary error function of a single-precision or extended-precision number
- Gamma function of a single-precision or extended-precision number
- Natural logarithm of the gamma function of a single-precision or extended-precision number
- Single-precision or extended-precision elliptic integral function of the first or second kind
- Single-precision or extended-precision complete elliptic integral function of the first or second part
- Special integral functions (exponential, sine-cosine, and Fresnel) of a single-precision number
- Jacobian elliptic functions sn, cn, and dn using single-precision numbers
- Bessel function of the first or second kind of a single-precision number
- Computation of mean, standard deviation, standard error of the mean, variance, kurtosis, skewness, minimum, maximum, and range
- Calculation of product-moment correlation coefficients, means, and standard deviations
- Extraction of a subset of correlation coefficients and a vector of a dependent variable
- Computation of multiple linear regression coefficients and related analyses
- Inversion of a symmetric matrix stored in triangular form

Documentation

For further technical information, consult the Scientific Library Reference Manual (30000-90027)

HEWLETT hp PACKARD

COBOL/3000 Compiler

model 32213C

COBOL provides you with language resembling English as a programming tool. It is self-documenting, easy to learn, and permits fast program development. The language has efficient statements to simplify file descriptions, I/O, table handling, sorting, mass storage manipulation and report generation. The compiler is integrated into the HP 3000 Multiprogramming Executive (MPE) to allow great flexibility in every environment.

Features

- **Packed decimal, binary and display (zoned) data types**
- **Sequential and random files**
- **File lock capability**
- **Interface to KSAM (Keyed Sequential Access Method) files**
- **Multiple entry points for subprograms**
- **Subprogram code segmentation allowed**
- **Data segmentation through dynamic-type subroutines**
- **Object code segmentation controlled by programmer**
- **Compile time editing**
- **Selective compilation**
- **Table handling up to 3 dimensions**
- **Optimal bounds checking for tables at program execution time**
- **Communication with COBOL or non-COBOL subroutines**
- **Direct communication with SORT/3000 via SORT verb**

Implementation level

The major standard describing COBOL compilers is the ANSI (American National Standards Institute) standard. Hewlett-Packard COBOL has fully implemented the ANSI 1968 standard in all categories*.

*Except Report Writer

The following table shows the COBOL/3000 rating.

MODULE	ANSI RATING
Nucleus	High
Table Handling	High
Sequential Access	High
Random Access	High
SORT	High
Report Writer	Null
Segmentation	High
Library	High

COBOL modules

COBOL/3000 is a set of functional processing modules that have the following capabilities:

Nucleus: Provides a basic language capability for the internal processing of data within the basic structure of the four divisions of a COBOL program.

Table Handling: Used to define tables of contiguous data items and access an item relative to its position in the table. Tables may be variable length and may have up to three dimensions.

Sequential Access: Used to access records of a file in an established sequence. Sharing memory area among files is also provided.

Random Access: Used to access records of a mass storage file according to a programmer-supplied key. Sharing memory area among files is also provided.

Sort: Used to order a file of records according to a set of user-specified keys within each record. Special processing of addition, deletion, creating, altering, editing, etc., is provided.

Segmentation: Used to specify object program segmentation requirements.

Library: Used to specify text that is to be copied from a library. Library text is available to a source program at compile time and need not be actually written as part of the source program.

Interprogram Communication: Provides the capability to call (or be called by) a program written in COBOL or other HP 3000 languages.

Language extensions

In addition to the ANSI standard, Hewlett-Packard has implemented a number of extensions which include:

- Interprogram communication
- Packed decimal (Computational-3)
- Note lines (defined by an asterisk in column 7)
- Current-date (MM/DD/YY)
- Time-of-day (HHMMSS)
- THEN optional
- Multiple REDEFINEs of a given location
- Unary+
- GO TO MORE-LABELS EXIT
- Synchronized for index data items
- Forms message for special forms

Data types

COBOL/3000 allows the following data types:

- Binary (Computational)
- Packed decimal (Computational-3)
- Display (Zoned)

Documentation

For further technical information, consult the COBOL Reference Manual (32213-90001).

hp museum

HEWLETT PACKARD

RPG/3000 Compiler

model 32104A

RPG, the Report Program Generator, is a machine-independent, problem-oriented report generating language that is easy to learn, use, and code. It allows you to specify many important operations with a minimum of effort by making simple entries on specially formatted coding sheets. Because RPG is a standard language available on many different machines, programs can be submitted coded in another manufacturer's RPG or RPG II, directly to the Hewlett-Packard RPG compiler with little or no re-coding for conversion. In addition, the RPG compiler helps detect errors at the source language level with extensive diagnostic messages.

Features

- **Automatic program segmentation**
- **Edit codes**
- **Calculation control of I/O**
- **Closed subroutines**
- **Single dimension arrays**
- **Automatic EBCDIC/ASCII file translation and alternate collating sequence**
- **Cross reference**
- **Formatted dump**
- **Preselected or dynamic run time error options**
- **Source level debugging with DEBUG**
- **Spread cards**
- **File error option**

RPG can handle jobs ranging from simple tasks such as printing address labels to very complex ones such as an entire payroll process (printing paychecks, payroll registers, and various allied reports). RPG is ideal for producing such documents as inventory lists, billings, invoices, insurance benefit notices, or summaries of sales, profits and losses, and customer transactions. It is also excellent for updating the master files used in producing these documents.

Extensions to RPG II

Parameters for external subroutine calls: Parameters may be specified after an EXIT (external subroutine call) operation, simplifying interfacing with COBOL, SPL, BASIC or FORTRAN subroutines.

Interface to data base management: Data bases can be accessed through regular I/O reads and writes by specifying the file as being an IMAGE data base in the file specification section. Keyed sequential files with up to 16 keys may be accessed in the same program.

Run-time error options: Three methods are provided for handling run-time errors.

1. Specifying on the Control Record at compile time whether the run-time error should be ignored or the program terminated.
2. Allowing the operator to determine the mode of operation at run time.
3. Testing an error code in RPG calculations and determining the mode of operation programmatically.

Cross reference option: A cross reference may be requested showing all references to file names, indicators and field names.

Automatic program segmentation: RPG will automatically segment code generated for an RPG program in programmer-selectable 2k, 4k, 6k or 8k-byte segments, resulting in a large virtual workspace.

EBCDIC/ASCII automatic translation: You can request RPG to automatically generate file translation tables for EBCDIC to ASCII or ASCII to EBCDIC conversions, or to use an EBCDIC alternate collating sequence.

Combined terminal file: You may define an Input/Output terminal file.

Calculation indicator repetition: Duplicate conditioning indicators need not be repeated line-to-line in calculation.

Data types

RPG allows data to be input or output in the following formats:

- Binary—one or two word binary data
- Packed decimal
- Alphanumeric
- Unpacked decimal
- Unpacked decimal with leading or trailing sign

Specification types

The statements that describe the input, processing, and output to the compiler must be written according to the rules of RPG. The seven specification types are:

- Control record
- File description
- File extension
- Line counter
- Input
- Calculation
- Output

Documentation

For further technical information, consult the RPG Compiler Reference and Application Manual (32104-90001).

HEWLETT PACKARD

FORTRAN/3000 Compiler

model 32102B

FORTRAN/3000 is a full implementation of the ANSI STANDARD FORTRAN (X3.9-1966). It also has many extensions which expand the capabilities and increase the power of the language.

Features

- Seven data types—integer, double integer, logical, real, double precision, complex, and character
- Character variables and character arrays
- Bit extract and deposit capability with partial-word designators
- Arrays with up to 255 dimensions
- Named common blocks initialized by block data subprograms
- Multiple entry points for subprograms
- Support of user-written error handling routines which are called under trap conditions
- Parameter statement for specifying constants with symbolic names
- Dynamic array declaration and allocation in subprograms
- Up to 99 files available during execution of a FORTRAN program
- Functions and subroutines called recursively
- One logical IF statement as the dependent statement of another logical IF
- Parameters to non-FORTRAN subprograms passed by value rather than reference
- ACCEPT and DISPLAY statements for free field input/output
- Compilation time editing
- Symbolic names with up to 15 characters
- Action labels specified in READ/WRITE statements to indicate point of transfer in case of end-of-file or I/O error
- Label used as an argument in subprogram call statements to allow alternate return points
- Mixed mode arithmetic
- Generic functions
- Built-in optional cross reference listing
- Undefined variable detection

Source program format

FORTRAN/3000 was designed with several powerful convenience features for interactive users. The nature of terminal devices makes the historical position-dependent fixed-format program representation inconvenient; however, FORTRAN overcomes these drawbacks by offering both fixed format and free format representation for source language input.

File facility

Uniform access to disc files and standard input/output devices is accomplished through the MPE file system. You access your files using normal READ/WRITE statements. The structure of a file and method of access can be defined via a file statement or left to default values. This provides device independence and easy access to all types of files, including KSAM files and IMAGE data bases.

Device type can be defined at execution time; consequently, the devices used by a program can be readily changed.

Sequential and random access of disc files is supported by FORTRAN/3000.

If you have highly specialized requirements you may communicate directly with the MPE file system. Data file privacy is achieved through the normal MPE protection mechanisms.

Documentation

For further technical information, consult the following Hewlett-Packard manuals:

30000-90040 FORTRAN Reference Manual

32102-90002 FORTRAN Pocket Guide

HEWLETT PACKARD

BASIC/3000 Interpreter

model 32101B

BASIC is an easy-to-learn language designed especially for interactive terminal use. Applied extensively in business, educational, and scientific environments, BASIC/3000 provides all the features commonly found in a BASIC system, as well as numerous extended capabilities. Consisting of both an interpreter and a compiler, BASIC/3000 allows you to create and debug your BASIC programs interactively and then compile them for faster production execution.

Features

- **Programs and data files accessed from either timeshare or batch mode**
- **Conversational program generation with extensive messages**
- **Four numeric data types: real, integer, real extended precision, and complex**
- **Mixed mode arithmetic**
- **All standard functions (SIN, COS, LOG, etc.), plus matrices, strings and files**
- **Program segmentation with common storage**
- **User definable file security including password**
- **Can be used alone or in conjunction with BASIC Compiler**

Environment

Programs and data files can be accessed from either interactive or batch mode.

Interactive mode

Implementation of BASIC in the HP 3000 operating system (MPE) results in a very powerful language which encourages the use of extensive conversational capabilities.

Batch mode

HP BASIC itself is a flexible language that may also be used in batch mode. In batch mode, all input (program statements, commands and data) is read from the batch input device; all output is directed to the batch output device.

User tailored modes

BASIC permits full use of MPE device independence. You can link each type of input (program statements, commands and data) and output (program output, messages and listings) with any available peripheral device. This flexibility within BASIC can be employed to construct end-user packages in which BASIC is invisible to the user. The resultant simplicity of execution is especially important to instructional/ educational applications.

Character string manipulation

The BASIC Interpreter incorporates the ability to define and manipulate ASCII character strings and string arrays. All digits, upper and lower case alphabetic characters, and all other printing and non-printing ASCII characters can be stored in string variables. They can be input and output at the terminal and stored and retrieved from data files. Substrings as small as zero characters and as large as 255 characters in length can be printed, concatenated and compared to other strings. These may be used for branching or sorting.

Data files

BASIC maintains three distinct file types:

- FORMATTED files provide advanced, easy-to-use capabilities that are intended for (but not restricted to) BASIC language use. These enable run-time checking of file data type.
- ASCII and BINARY files are available for communicating data to and from programs written in languages other than BASIC.

BASIC FORMATTED files may have a record size between 4 and 319 (16 bit) words. Data can be accessed either serially or on a record basis with random access to any record in the file. The ADVANCE and UPDATE statements provide the capability to access individual items within a record. Files may be created and purged either by commands or under program control.

Subroutines

BASIC provides two types of functions and two types of subroutines:

- Built-in functions include SIN, TAN, TNH (hyperbolic tangent). Approximately 40 such functions are provided.
- User defined functions are established in the user's program and can be called from within the program. They may consist of multiple statements and local variables and arrays whose scope extends only within the declared function.
- A simple subroutine consists of a set of BASIC statements followed by a return statement. There is no explicit indication in a program as to which statements comprise a subroutine.
- External subroutines which are written in another language, i.e., FORTRAN or SPL can be called by BASIC. BASIC programs may also call external subroutines from the libraries accessible to the user.

Documentation

For further technical information, consult the following Hewlett-Packard manuals:

03000-90025 BASIC for Beginners

30000-90026 BASIC Interpreter Reference Manual

03000-90050 BASIC/3000 Pocket Guide

32103-90001 BASIC Compiler Reference Manual

BASIC/3000 Compiler

model 32103B

The BASIC Compiler provides the means for converting BASIC programs (including those which have been written, debugged and saved via the BASIC Interpreter) into machine code. Compiled BASIC programs exist in the system as actual code segments and can be run directly, rather than through line-by-line interpreting.

Features

- **Supports all HP 3000 BASIC interpreter language extensions**
- **Faster average execution speed than interpreter**
- **Shareable machine code**
- **Can be combined with SPL procedures and FORTRAN subroutines in the same program file**
- **Load-and-go capability**

Since the programs to be compiled are written using the BASIC Interpreter, all of the language features described for the interpreter apply to the BASIC/3000 Compiler as well.

Environment

There are three general phases in the development of a BASIC compiled program—program development, compilation and preparation, and execution.

Program development

In the first phase, a BASIC program is usually written and debugged interactively using the BASIC Interpreter commands and statements. The interpreter constructs the interpretive version of a program. When you are satisfied that the BASIC program runs properly in its interpretive form, you save it (SAVE, FAST) in a file. This fast save file is the "source" input to the BASIC Compiler.

Compile and prepare

The BASIC Compiler is used to compile the fast save file. The program is then prepared in a form that results in an efficient machine code version of the original program.

Execution

The third phase is to execute the program directly under the operating system using the RUN command. BASIC programs may be compiled in batch or interactive mode, and programs may be run in either mode.

Documentation

For further technical information, consult the following manuals:

03000-90025 BASIC for Beginners

30000-90026 BASIC Interpreter Reference Manual

03000-90050 BASIC Pocket Guide

32103-90001 BASIC Compiler Reference Manual

HEWLETT PACKARD

APL\3000 Language Subsystem

model 32105A

APL\3000 is a language subsystem for the Hewlett-Packard 3000 computers consisting of an advanced version of APL (A Programming Language). This language is popular in both business and scientific environments because of its conciseness, power, and exceptional facility for manipulating arrays of data.

Features

- **Patterned after APLSV – shared variables, same standard notations and extensions, many enhancements**
- **Virtual workspaces – size limited only by on-line storage**
- **Friendly, powerful editor**
- **APLGOL, a structured, easy-to-maintain language extension to APL**
- **Full power of the HP 3000 file system**
- **Interactive or batch operation**
- **Dynamic incremental compiler**
- **Extended control functions**
- **Access through terminals with standard ASCII interface**

APL works in an environment known as a workspace in which you can place variables (data) and user-defined functions (algorithms). Workspaces can be saved, loaded and modified. You enter APL in a calculator mode where calculations and function execution can be done directly.

Full APL plus extensions
APL\3000 contains the extensions to IBM APLSV which typically are part of the most recent implementations of the language. These include: format (⍕), execute (⍎), scan(\ ), and matrix divide (⌹); continue workspace; system variables; shared variable capability; system functions such as canonical representation (⎕*CR*), name list (⎕*NL*), and DEBUG and TRACE functions.

In addition to the extensions commonly found in APLSV, APL\3000 is substantially enhanced by the addition of several other powerful extensions.

Virtual workspaces
A firmware-assisted virtual memory scheme is employed in APL\3000 with the result that very large workspaces are available, constrained only by the amount of free on-line storage. In many cases this important feature allows the incorporation of files into the workspace.

Friendly, powerful editor
APL\3000 features a full text editor which may be used in both calculator mode and edit mode. Information in edit mode can be stored as a character matrix or vector matrix if desired. A HELP command lists all commands or individual meanings as requested.

The APL\3000 editor is much more powerful and friendly than the standard APL del (∇) editor. Edit commands, most of which can be abbreviated, are: ADD, BRIEF, CHANGE, COPY, CURSOR, DELETE, DELTA, END, FIND, HELP, EXPLAIN, LIST, LOCK, MATRIX, MODIFY, QUIT, REPLACE, RESEQUENCE, UNDO, VECTOR, VERBOSE.

APLGOL
You have the option of defining functions in either APL or APLGOL, a unique Hewlett-Packard extension of APL. APLGOL uses ALGOL-like key words in conjunction with APL expressions to describe the control flow within a given function. This provides a framework for structured programming in an APL environment.

The APLGOL commands are: ASSERT, BEGIN END, BEGIN + END CASE, CASE OF, EXIT, FOREVER DO, HALT, IF DO, IF THEN ELSE, NULL, REPEAT UNTIL, WHILE DO.

Shared variables
This facility allows you to send and receive information outside of APL workspaces. Communication with other processes can include devices such as printers, plotters, mag tapes, and files. It does not include passing information directly between two APL workspaces.

File system
A link to the MPE file system is provided through the use of shared variables to give APL\3000 the full power and flexibility of the MPE file facility. APL provides a straightforward mechanism for data type conversion to and from APL and the file system. Features include allowing reference to files without specific knowledge of their actual names or characteristics and many classes of file security.

Dynamic incremental compiler
Rather than interpreting function code, as is done in other systems, APL\3000 compiles, runs, then saves the compiled code statement by statement. Subsequent program executions can often use this saved code, allowing them to rerun much faster. Of course, when solving one-line one-time problems compiled code is thrown out just as it would be for an interpreter.

A signature containing characteristics of the statements (such as data types, ranks, & dimensions) is saved in the code for each statement. On subsequent runs this signature is compared with the new input data; if the characteristics have changed, a new compilation is performed. This process typically generates a more flexible code which is able to handle all the data experienced to date. Compiling is done automatically, so that the system retains all the benefits of an interpreter.

Extended control functions
For debugging involving chains or for applications which require returning to previous environments, the APL\3000 extended control functions are extremely useful. Complex problems in such areas as artificial intelligence, pattern recognition, modeling simultaneous processes, and symbol manipulation can be solved through use of the extended control functions. Standard APL implementations limit you to a linear stack, while APL\3000 with its extended control functions offers capabilities such as co-routining and backtracking.

Documentation
For further technical information, consult the following Hewlett-Packard manuals:

32105-90002 APL\3000 Reference Manual

32105-90003 APL\3000 Pocket Guide

HEWLETT PACKARD

SPL/3000 Compiler

included in Fundamental Operating Softwar

SPL is the Systems Programming Language for the HP 3000 computer systems. Because it combines the efficiency of a machine-dependent language with the simple structure of a high level language, SPL was used to write the HP 3000 operating system, language compilers, system utilities, and data base management and data communications subsystems.

Features

- **Self-documenting for ease of readability**
- **Permits access to all hardware features and data types**
- **Dynamic allocation of local storage for working space and local variables in procedures. Memory de-allocated on exit from procedures**
- **Program segmentation feature**
- **Assemble statement permits machine level coding**
- **Six data types–logical, byte, integer, double integer, real, and long real**

If you are used to developing your application software using assembly language, you will find that SPL provides the same efficient and powerful results while greatly reducing the development time. It allows you to write software quickly, easily and efficiently, while producing object programs with good code compression and efficient execution times.

To simplify systems programming, SPL offers a high level language similar to (but not equivalent to) ALGOL, together with features which enable you to easily exert control over machine-dependent functions of the computer system.
You can operate directly on hardware registers, perform branches based on hardware status, extract/deposit/shift bit fields, or generate any sequence of hardware machine instructions. Also, commonly executed routines can be written in SPL and called from COBOL, BASIC, RPG, FORTRAN or APL programs.

The language provides many features normally found only in applications languages such as ALGOL and PL/1, and includes:

- Free-form structure
- Arithmetic and logical expressions
- High level statements with unlimited nesting (IF, FOR, SWITCH, CASE, DO-UNTIL, WHILE-DO, MOVE, SCAN, assignment and compound statements)
- Recursive procedures
- Variables and arrays of many data types

Environment

Programs may be compiled in batch or interactive mode.

Variables

Variables may be either "global" or "local." Global variables are those declared in the main program and are accessible from any part of the program including procedures. Local variables, however, are declared within a procedure and are only accessible from within that procedure.

Programming features

Each SPL statement is either a high-level or machine-dependent feature.

High-level features: In all programming efforts, a need frequently arises for standard program constructs, such as loops, and evaluation of arithmetic expressions. Rather than hand-coding these often-used structures each time, SPL/3000 allows you to write them at a high level. The compiler then provides an efficient, error-free code sequence in each case.

Machine-dependent features: SPL allows the use of machine-level constructs to ensure complete control of the HP 3000 computer systems. These constructs permit the following:

- Direct register references
- Branching based on actual hardware conditions
- Bit extraction, deposit, and shift
- Generation of any sequence of hardware machine instructions (in the midst of high level constructs)

Documentation

For further technical information, consult the SPL Reference Manual (30000-90024) or the SPL Textbook (30000-90025).

Training

Hewlett-Packard offers the following training course for this product:

22804A HP 3000 SPL File System Introduction, 5 days, HP Technical Center.

HEWLETT hp PACKARD

DEL/3000

Data Entry Library

model 32206A

The Data Entry Library is a programming aid designed to simplify terminal oriented data collection. It provides capabilities for designing and displaying forms on the screen of an HP 2640 series terminal, either line by line or with full page mode operation. The DEL procedures accept data input from the terminal and check the data for errors.

Features

- **Forms creation, update, and deletion through a single forms maintenance program**
- **Forms-handling, terminal-handling, and editing procedures callable from COBOL, BASIC, FORTRAN, and SPL**
- **Employs HP 2640 series CRT terminals with display enhancements and line/page block mode operation**
- **Interfaces with hardwired and remote terminals**

DEL Components

Forms maintenance program

You may use the program FORMAINT to interactively create forms and store them in forms files on disc. The menu presented by FORMAINT also allows you to display, modify or delete forms, list all forms contained in a forms file, or delete an entire forms file.

After the form is created, you may specify editing, test, and range-check information which is applied when an application program displays and reads from the form.

User callable procedures

DEL provides a set of procedures which can be used in COBOL, FORTRAN, BASIC, or SPL application programs to display a form image on the terminal screen, accept data entered on the form image by a terminal operator, and edit this data. You may then store this data in a data file on disc in the format established by your program. The form image serves as a visible guide or template to entering the data. These procedures are classified into four groups.

Forms procedures – To aid the application program in manipulating the forms contained in the forms file, including:

OPENFORM Opens a specified form file, verifies the file created by FORMAINT, and initializes user's communications area.

FINDFORM Searches the form file for a specified form and sets current form information in user's communications area.

GETFORM Reads form definition into user-buffer and uses control characters to lock the keyboard, set format mode, etc.

NEXTEDIT Returns edit specifications to user-buffer.

Terminal procedures – The application program can control the CRT terminal and retrieve information with the following procedures:

OPENTERM Identifies the terminal to the computer, verifies it is of the 2640 series, determines the terminal operating mode (line/page) and sets the information in the user's communication area.

WRITETERM Writes the contents of the user-program buffer to the terminal.

READTERM Reads all unprotected fields from the screen (removes field and block separators).

TERMSTATUS Reads the terminal status information and stores its values in the user-buffer area.

Edit procedures – To edit the unprotected fields on the CRT screen the following procedures are available:

ALPHAEDIT Scans input field, indicates an edit failure condition if characters are not the letters A thru Z.

ALPHAFILL Performs like ALPHAEDIT, except there may be any number of spaces to the right of the last alpha character.

ANEDIT Scans the input field, indicates an edit failure if any character is not alphabetic letter, space, or digit 0 thru 9.

NUMRCEDIT Scans the input field, indicates an edit failure if any character is not digit 0 thru 9.

ZEROFILL Works like NUMRCEDIT except there may be any number of spaces before or after the numeric digits, and a plus or minus before them.

NRANGE Compares input data with the contents of a low range and a high range field.

MIICREATE Generates a modulo eleven checkdigit and inserts the digits in the right-most digit-space of the input field.

MIIVERIFY Will generate a modulo eleven checkdigit and compare with the right-most digit-space of the input field.

High-level procedures – These are special combinations of previous procedures to perform specific terminal-handling or editing tasks:

SHOWFORM Combines the FINDFORM, GETFORM, WRITETERM, and READTERM procedures.

EDITFIELD Combines NEXTEDIT with the required edit procedures to perform the appropriate editing.

Documentation

For further technical information, consult the following Hewlett-Packard manual:

30000-90050 Data Entry Library Reference Manual

Training

Hewlett-Packard offers the following training course for this product:

22819A Data Entry Library, one day (On-site only).

HEWLETT hp PACKARD

IMAGE/3000

Data Base Management Subsystem

models 32115A and 32115B

IMAGE is a general purpose data base management system that allows information to be related logically between data sets (files), minimizing data redundancy and facilitating information retrieval. It provides facilities to describe data base structures, create a data base, and access, maintain, restructure and back-up data. IMAGE operates concurrently in both terminal and batch environments within the constraints of an external (MPE) and internal security scheme. Application programs for use with IMAGE may be written in COBOL, RPG, FORTRAN, compiled BASIC, and SPL (Systems Programming Language). When used in conjunction with a computer network featuring Distributed Systems/3000 software (DS/3000), IMAGE allows direct access and modification of remote data bases.

Features

- **Network structure allowing for complex data relationships**
- **Password security at data base, data set, and data item levels**
- **Serial, direct, calculated, and chained access methods**
- **Concurrent terminal and/or batch access to one or more data bases on local or remote HP 3000**
- **Library routines callable from COBOL, FORTRAN, BASIC, and SPL; RPG language interface**

IMAGE components

IMAGE consists of three components:

Data base definition language. The data base designer describes the data items (fields), security, data sets (files), data set relationships, and storage requirements using a **data base definition language;** this description is called a **schema.** Schema statements are then processed by a schema processor (DBSCHEMA), to create a stored data structure known as the **root file.** The location and relationships of information are known to IMAGE through this root file.

Data base management intrinsics. To access and maintain data on both local and remote HP 3000 computer systems, a set of IMAGE library routines is provided. These intrinsics are callable from user written programs, and include the following:

DBOPEN Initiates access to a data base.

DBINFO Returns information about the data base currently being accessed.

DBGET Retrieves items from data entries (records).

DBPUT Adds new data entries to the data base.

DBLOCK Provides temporary exclusive control of a data base.

DBCLOSE Terminates access to a data base.

DBFIND Prepares for chained access to data entries.

DBUPDATE Modifies existing data entries.

DBDELETE Deletes data entries.

DBUNLOCK Relinquishes temporary exclusive control of a data base.

DBEXPLAIN Examines status information and prints an English-language message on the standard listing device.

DBERROR Examines status information and supplies an English-language message in a buffer.

Data base utilities. Stand alone utility programs aid in the creation and the maintenance of the data base.

DBUTIL
(1) Allocates and initializes disc space for a data base
(2) Re-initializes the data sets of a data base back to their empty condition
(3) Purges the root file and all data sets of a data base
(4) Activates or deactivates access to a remote data base.

DBSTORE Produces a physical copy of a data base on a serial storage device.

DBRESTOR Copies a data base from a serial storage device.

DBUNLOAD Produces a logical copy (data only) of a data base on a serial storage device.

DBLOAD Loads data saved by DBUNLOAD into an existing data base on disc.

Data types

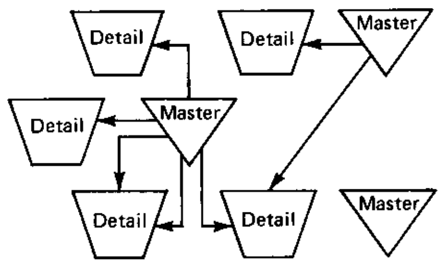
Signed binary integer, logical binary, real, ASCII character string, packed decimal, and zoned decimal.

Data set types

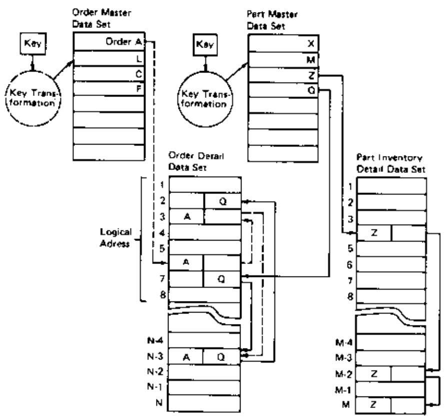
IMAGE supports two types of data sets, master and detail. Access to data entries in a master data set may be calculated, based on the key value of the data entry. Access to a data entry in a detail data set is usually via a particular master entry and a particular relationship between the master and detail data sets.

The logical relationships which may exist are exemplified in the following diagram:

Image Network Structure



An example of data set relationships is shown in the following diagram:



Master Detail Chain Paths:

Order Master entry A links to Order Detail entries 6, 3, and N-3. Parts Master entry Q links to Order Detail entries 7, N-3, and 2. Parts Master Z links to part inventory entries 4, M-2 and M. This data sturcture allows a master entry to be related to many detail entries, and a detail entry related to many master entries.

Security

IMAGE maintains privacy and security down to the item level. Up to sixty-three classes of users can be defined. A password is associated with each class. Sets of user classes can then be permitted 'read' or 'read-and-write' access to any or all data items and data sets, independent of the elements accessible to other user classes.

Restructuring
DBLOAD, DBUTIL, and DBUNLOAD utilities provide for restructuring data bases in a number of ways, including changing data item names or data set names, increasing or decreasing data set capacities, adding or removing data items at the end of a data entry, and changing data set relationships.

Accessing data bases
A data base is accessed through a call to IMAGE intrinsics from COBOL, FORTRAN, SPL, and BASIC, through the chain and read statements of RPG programs after the data has been declared in the RPG file specifications section, or from QUERY as part of the language function.

Data entries may be accessed serially, directly, or by key value.

Each of these data base access methods can be applied to both local and remote IMAGE data bases. To access a data base on a remote HP 3000, both the local and remote computer systems must include DS/3000 and IMAGE/3000.

Additional features
- Multiple master-detail data set relationships
- Shared and exclusive data base access
- Storage and retrieval of related entries in sorted sequence
- Access to multiple data bases
- Automatic linkage management when data is added, modified, or deleted
- Efficient disc utilization (no index or overflow areas and automatic reuse of deleted record space)

Specifications
- Data item names per data base: 255
- Data items per data entry: 127
- Data sets per data base: 99
- Detail data sets per master data set: 16
- Master data sets per detail data set: 16
- Search items (keys) per detail data set: 16
- Maximum entry size: 2047 words (4094 bytes)
- Entries per data set: $2^{23}-1$ (8,388,607)
- Entries per chain: 65,535
- Characters per data base name: 6
- Characters per password: 8
- Characters per data set name: 16
- Characters per data item name: 16

Documentation
For further technical information, consult the following Hewlett-Packard manual:

32215-90003 IMAGE Reference Manual

Training
Hewlett-Packard offers the following training courses for this product:

22956A IMAGE Data Base Management, 5 days (HP Technical Center)

22827A IMAGE Data Base Management, 5 days (On-site)

HEWLETT PACKARD

QUERY/3000

Data Base Inquiry Facility

model 32216A

QUERY is a data base inquiry facility designed to easily locate, report, and update data values within an IMAGE data base. QUERY can be executed from either a terminal or batch device, and reports may be directed to either a terminal or a line printer. You communicate with QUERY through 23 English-like commands, all of which may be used with both local and remote data bases. To use QUERY with a remote data base, both the local and the remote HP 3000 must include Distributed Systems/3000 (DS/3000) and the IMAGE/3000 Data Base Management System.

Features

- **Interactive or batch data base interrogation**
- **English-like commands for simplified use by non-programmers**
- **Data base updating through addition, deletion, and modification of data records**
- **Formatted reporting of retrieved data including page titles, column headings, group subtotals, totals, and averages**
- **Command files to store complex or frequently-used commands for repeated execution**

Security considerations

QUERY adheres to all the security provisions of the IMAGE data base. The security password determines which data elements (i.e., data items and data sets) you are allowed to access. QUERY returns an error whenever your security class does not match the security class of the data element you are attempting to access.

You may display the data base structure and determine which data elements are accessible by entering the FORM command. QUERY responds by listing all the data elements available, based upon the password that was entered.

Locating data

QUERY is capable of retrieving all occurrences of data within a data set which meet user-specified conditions. To accomplish this, you enter a FIND command which includes logical terms that are similar to phrases spoken in English. For example, suppose a production control manager wishes to know which part numbers are in short supply or over supply when compared with outstanding customer orders. Assuming the data base contains a part-number data set, the manager could locate all such parts with one FIND command as follows (note that LT means "is less than" and GT means "is greater than"): FIND QUANTY LT 100 AND CUST-ORD GT 50 OR QUANTY GT 10000 AND CUST-ORD LT 1000

QUERY responds to the FIND command by locating all the data entries (records) which contain the requested data item values.

Reporting data

After the data entries have been located through the FIND command, you may enter a REPORT command to specify which items within those records QUERY is to display. The REPORT command may also specify:

- Top-of-page titles including date and time
- Addition, counting and averaging of selected data items
- Arithmetic operations on data items using registers
- Column headings, group subtotals and totals
- Line spacing
- Up to 5 levels of sorting to produce grouped items
- Edit masks to suppress leading zeroes, insert punctuation characters, etc.
- Page skipping

For quick information, the REPORT command can simply specify that all data item names and their values are to be displayed without formatting. The LIST command will display all or specified data items, automatically formatted and with headings.

Updating a data base

Maintenance of the data base can also be performed using QUERY. The ADD, DELETE, and REPLACE commands, designed for this purpose, allow insertion and deletion of data entries and replacement of data item values. When ADD is entered, QUERY prompts you at the terminal for data item values. You are not required to enter values for all items.

Frequently used commands

Repetitive or complex operations are easily performed through QUERY's ability to execute FIND, REPORT and UPDATE commands from a command file stored on disc. A command stored within a file is referred to as a procedure; a procedure may consist of one or more lines. QUERY provides commands for creating, deleting and listing procedures within a command file. Also, the lines within a procedure may be added, deleted or replaced. In addition, a sequence of commands can be stored in a file and executed at any time by entering a single command, XEQ.

Data types

The following data types are converted and error-checked during QUERY I/O operations:

- One word integer numbers
- Two word integer numbers
- Two word real numbers
- Extended precision real numbers
- One word logical values as absolute numbers
- ASCII character strings containing no lower-case alphabetics
- General ASCII character strings
- Zoned decimal numbers
- Packed decimal numbers

Documentation

For further technical information, consult the following Hewlett-Packard manual:

30000-90041 QUERY Reference Manual

Training

Hewlett-Packard offers the following training courses for this product:

22956A IMAGE Data Base Management, 5 days (HP Technical Center)
22827A IMAGE Data Base Management, 5 days (On-site)

HEWLETT PACKARD

KSAM/3000

Keyed Sequential Access Method

model 32208A

KSAM (Keyed Sequential Access Method) allows you to create and maintain disc files whose records are accessed by the value of key fields within the data records. Each data record contains one primary key field and may include up to 15 alternate key fields. Data records are written to a KSAM file in any order without regard to key sequence, although they may be presorted if desired. Records are accessed sequentially or randomly by primary or alternate key value, by logical record number, or in chronological (physically sequential) order. Duplicate key values are allowed, and records can be accessed by generic keys (partial key values) or by approximate keys.

Features

- **Multiple keys; one primary and up to 15 alternate keys**
- **Duplicate key values allowed**
- **Retrieval by generic key value or by approximate match**
- **Access from COBOL, RPG, FORTRAN, BASIC or SPL**
- **Fixed or variable length data records**

KSAM Components

KSAM consists of the following two logical components:

KSAM procedures and intrinsics. To assist in accessing KSAM data records, special sets of COBOL procedures, BASIC procedures, and MPE file system intrinsics (for FORTRAN and SPL) have been provided.

Note: Most of the standard MPE file system intrinsics can be used with little or no parameter changes to access and manipulate KSAM files.

KSAM utility program (KSAMUTIL). This utility program provides a set of commands designed specifically for creating and manipulating KSAM files.

BUILD Creates a KSAM file consisting of a data file and a key file.

ERASE Clears the contents of a KSAM data file and resets the pointers in the associated key file.

PURGE Deletes a KSAM file from the system.

RENAME Changes the name of a KSAM data file or key file.

SAVE Saves a session/job temporary KSAM file as a permanent file.

VERIFY Displays the current status of a KSAM file.

HELP Displays a description of all the KSAMUTIL commands.

EXIT Terminates KSAMUTIL and passes control back to the MPE command interpreter.

KSAM file structure

A KSAM file consists of two physical disc files: a key file and a data file. A KSAM data file contains all the data records. The associated key file contains one or more sets of entries that maintain the primary and alternate logical sequences of the data records. When a data record is added to the data file, a key entry is added to the associated key file for each defined key field in the new data record. These key entries are dynamically added in ascending order to maintain the sequential nature of the key file. The diagram below illustrates the functional relation between a KSAM key file and data file.

Accessing KSAM files

Data records in a KSAM file can be retrieved, updated, or deleted sequentially or in random order by key value through the key file. They can also be accessed as they were written, in chronological order, or by record number without using the key file.

KSAM files can be accessed:

From RPG: Through file specifications and chaining operations.

From COBOL: Through call statements to a special set of COBOL procedures designed specifically for use with KSAM files. (These procedures correspond to the INDEXED I-O module of ANSI X3.23-1974 COBOL.)

From FORTRAN: Through call statements to MPE file system intrinsics or to the special set of COBOL procedures.

From BASIC: Through call statements to a special set of BASIC procedures designed specifically for use with KSAM files.

From SPL: Through MPE file system intrinsic calls.

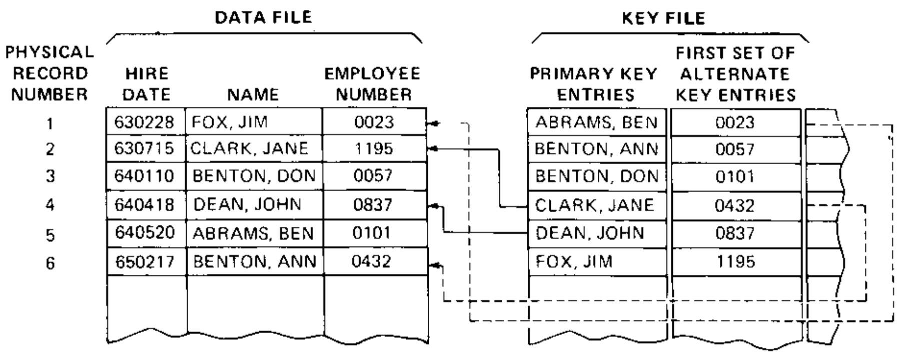
KSAM files can be opened for shared access by more than one user, or access to a KSAM file can be restricted to a single user.

Key file restructuring

KSAM automatically performs any key file restructuring required to accommodate newly added records to the associated data file. This restructuring is invisible to you.

Data types

Data fields, including key fields, may contain the following types of data: integer, double integer, real, four-word long real, byte string, packed decimal, and numerical display.



DATA FILE

PHYSICAL RECORD NUMBER	HIRE DATE	NAME	EMPLOYEE NUMBER
1	630228	FOX, JIM	0023
2	630715	CLARK, JANE	1195
3	640110	BENTON, DON	0057
4	640418	DEAN, JOHN	0837
5	640520	ABRAMS, BEN	0101
6	650217	BENTON, ANN	0432

KEY FILE

PRIMARY KEY ENTRIES	FIRST SET OF ALTERNATE KEY ENTRIES
ABRAMS, BEN	0023
BENTON, ANN	0057
BENTON, DON	0101
CLARK, JANE	0432
DEAN, JOHN	0837
FOX, JIM	1195

CHRONOLOGICAL ORDER: 1, 2, 3, 4, 5, 6

PRIMARY KEY (Last Name) ORDER: 5, 6, 3, 2, 4, 1

FIRST ALTERNATE KEY (Employee Number) ORDER: 1, 3, 5, 6, 4, 2

Security
All MPE file system security provisions also apply to KSAM files.

Compatibility
FCOPY/3000 and the MPE STORE/RESTORE commands can be used with KSAM files. FCOPY can produce copies of files that are sequenced in primary key order, alternate key order, or chronological order. In addition, the NOKSAM option allows you to retrieve records that have been flagged as "deleted." The SUBSET option allows you to select only certain records based on key value or record number.

Documentation
For further technical information, consult the following Hewlett-Packard manual:

30000-90079 KSAM/3000 Reference Manual

Training
Hewlett-Packard offers the following training course for this product:

22828A KSAM/3000, 2-day, on-site training course

HEWLETT hp PACKARD

RJE/3000

2780/3780 Emulation Subsystem Software

model 30130E

Remote Job Entry (RJE/3000), the 2780/3780 Emulation Subsystem Software, allows you to transfer data between an HP 3000 and a variety of remote host processors, in a full multiprogramming environment. It communicates via dial-up telephone or leased lines at speeds up to 9600 bits per second through numerous HP recommended modems and the Synchronous Single Line Controller. The emulator causes the HP 3000 to appear (to a remote processor) as either an IBM 2780 or 3780 data communications terminal. Added flexibility results from the variety of input/output devices, including both interactive terminals and batch peripherals, which are available to you.

Features

- **Accessible from both interactive terminals and traditional batch devices**
- **Unrestricted peripheral and file access for input and output**
- **Provides for multiple users to queue jobs to be transmitted**

The package includes all standard 2780 capabilities with the exception of 6-bit transcode and all standard 3780 capabilities except for reverse interrupt and conversational mode. Also offered are 22 optional capabilities, including:

- Blank compression
- Short record truncation
- EBCDIC and ASCII transparency
- Horizontal tabulation
- All 2780/3780 vertical format control
- Multi-record transmission (can optionally transmit more than seven records per block under user control)
- Print/punch component select

You can interface with the emulator though seven commands which control the sequence of input/output processing. Additionally, the full capabilities of the HP 3000 are available with related service utilities (e.g., EDIT/3000) to expedite job control programming and to facilitate program/data file transfer and maintenance.

Documentation

For further technical information, consult the following Hewlett-Packard manuals:

HP 3000 2780/3780 Emulator Reference Manual (30000-90047)

HP 3000 Guide to Synchronous Modems (See Appendix F)

HEWLETT PACKARD

MRJE/3000

Multileaving Remote Job Entry Software

model 32192A

Multileaving Remote Job Entry (MRJE/3000) software gives multiple users on a local HP 3000 simultaneous batch access to any remotely connected host computer system utilizing a HASP II (version 3.1 or 4.0) or JES2 job entry system. Since the HP 3000 appears to the host as a multileaving remote job entry work station, you can submit any job from your local HP 3000 which could be entered into the host system from a locally attached peripheral device. You can enter input from and direct output to any HP 3000 supported peripheral device or file. You can also communicate with multiple hosts and/or over multiple lines to a single host.

Features

- **Accessible from both interactive terminals and traditional batch devices**
- **Job input from any peripheral device or file**
- **Job output to any peripheral device or file**
- **Available to multiple users simultaneously**
- **Complete interactive HASP II or JES2 work station console support**
- **Supports an operator console, as well as up to 7 printers, 7 card readers, and 7 card punches**
- **Supports multiple hosts and/or multiple lines to a single host**

Available from both interactive terminals and batch devices, MRJE job transmission occurs concurrently with other HP 3000 activities. Communication between the local HP 3000 and the host system takes place over dial-up telephone or over leased (dedicated) lines at speeds up to 9600 bits per second through numerous HP recommended modems and the Synchronous Single Line Controller.

The MRJE emulation package consists of two functional modules. One module allows you to submit jobs, display job status, and define the characteristics of the work station. You may submit MRJE jobs whether or not an actual connection exists between the local and remote systems. Jobs submitted off-line are spooled and automatically transmitted when the connection is made.

The second module permits the MPE console operator to establish the MRJE communications connection with the host system, after which all user jobs (both spooled and newly initiated) are automatically transmitted to the host system for processing. Output is then directed from the host to the proper HP 3000 peripheral device or file without further intervention. Also, while the communications line is connected, the MRJE manager can enter host console commands interactively from any local HP 3000 terminal, permitting him to monitor and control job activity.

Automatic job routing

An additional feature of MRJE/3000 is its automatic job routing capability, which allows you to specify an output device or file when you submit your job. Output received from the host is then automatically directed to the specified device. If no output destination has been indicated, job output is routed to the default device designated by the MRJE manager.

User requirements

Users of MRJE/3000 software must be operationally familiar with Job Control and other procedures associated with the HASP II or JES2 job entry system of their host computer.

Documentation

For further technical information, consult the following Hewlett-Packard manuals:

MRJE/3000 Reference Manual (32192-90001)

HP 3000 Guide to Synchronous Modems (See Appendix F)

HEWLETT PACKARD

MTS/3000

Multipoint Terminal Software

model 32193A

Multipoint Terminal Software (MTS/3000) permits two-way, half duplex data transmission between an HP 3000 computer system and multiple multipoint terminals in the HP 2640 series via a single communications line. The terminals may be connected to the computer by means of a modem (remote access) or may be hardwired to the computer (local access) through the Synchronous Single Line Controller.

Features

- **Up to 32 terminals can share a line**
- **Transmission speed of up to 9600 bps**
- **Full page mode operation**

Functional capabilities

In both the log-on (interactive) mode and the data entry page mode, you can enter data into the terminal's memory using the cursor positioning capabilities, TAB key, and RETURN key. This data can then be edited as much as desired until the ENTER key is pressed, transferring the data to the computer.

Each of the multipoint terminals can be viewed as a processor with several peripheral devices—keyboard, display, two cartridge tape units, and printer. MTS/3000 allows the computer system to control these peripheral devices by transmitting the appropriate escape sequences to the terminal. The capabilities provided by the escape sequences include data transfers from the HP 3000 to a peripheral device, from a peripheral device to the HP 3000, and from one peripheral to another within the same terminal. An escape sequence also exists to retrieve status information for a peripheral device.

Additional capabilities include:

- Automatic error detection and retransmission of data
- Remote synchronous communication over either switched or leased lines
- Log-on (interactive) operation
- Transmission of variable length blocks of data
- Hardwired terminal communication in either synchronous or asynchronous mode

Specifications

- Data transfer at up to 9600 bps
- Two-way, half duplex operation
- Line protocol is similar to IBM Binary Synchronous Communication
- 255 maximum terminal identification numbers
  —26 group identifiers
  —10 terminal identifiers per group
- Terminals supported
  HP2641A, HP2645A, HP2648A

Documentation

For further technical information, consult the following Hewlett-Packard manuals:

MTS/3000 Reference Manual (32193-90002)

HP2641A, 2645A, 2645S, Display Station Reference Manual (02645-90005)

HP2648A Graphics Terminal Reference Manual (02848-90002)

DS/3000 Distributed Systems Software

model 32190A

The Distributed Systems Software (DS/3000) provides the capability to establish interactive communications links between different types of Hewlett-Packard computer systems in geographically dispersed locations. The systems may be linked using coaxial cables with the Hardwired Serial Interface or via modems using the Synchronous Single Line Controller. Part of the Hewlett-Packard Distributed Systems Network, these links permit an HP 3000 computer system to communicate with another HP 3000 (modem or coaxial cable), with an HP 1000 system (coaxial cable), or with an HP 2026 system (modem link). This flexibility allows you to design a network configuration capable of performing your specific data processing tasks.

Features

- Remote command processing
- Remote file access
- Remote data base access
- Program to program communication

Designed to be both easy to use and versatile enough to adapt to a broad spectrum of commercial/industrial applications, most of the powerful DS/3000 capabilities can be used with little or no programming effort. As soon as the system hardware and software are installed, you can begin to make use of the expanded communications features.

Remote command processing

From a local HP 3000 or HP 1000, you can execute the entire set of MPE commands on a remote HP 3000 using DS/3000 remote command processing. You can also execute system level RTE operator commands on a remote HP 1000 while connected to a local HP 3000. All HP 3000 MPE commands can be executed remotely from the system console of the HP 2026 in an interactive mode.

Remote file access

With DS/3000 linking an HP 3000 to either another HP 3000 or an HP 1000, you have access to the files of the remote system, under control of the usual system file security.

When transferring data between two HP 3000 computers, you can select an optional data compression feature. By merely including a single parameter in the DSLINE command which opens the remote communications line, you enable the system to compress blanks and strings of repeated characters before transmission and then expand the code to the form of the original data at the destination. This data compression capability can save transmission time, decrease response time, and increase throughput, especially when moving large amounts of data on slower, modem linked lines.

Peripheral devices connected to the remote system are also readily accessible, since they are treated by the system as files. Three methods exist by which you can access remote files--terminal access, standard language input/output statements and file manager calls.

Terminal access is available only for the 3000/3000 communications link. With this capability, you merely issue a local MPE FILE command on your terminal defining the desired device or file residing at the remote HP 3000 site. Included in the FILE command must be a remote device specification denoting the location of the desired file. The remote file may then be utilized on a record by record basis or as a complete file as though it resided on your local computer.

The same remote file access capabilities are also available in HP 3000 to HP 3000 communications using standard language input/output statements in application programs. This permits local programs written in COBOL, FORTRAN, BASIC, and SPL to define files and manipulate complete files, file records or devices on a remote HP 3000 as if the data or device were on the local system. All that is required is a FILE command which may be made external to the program or may be included in the program. This command specifies the location of the target device or file. Subsequently, the remote file or device may be utilized as if it resided on your local computer. RPG programs may also use this capability by accessing an SPL subroutine with the appropriate commands.

Access via File Manager Calls is a feature of both the 3000/3000 and 1000/3000 communications links. With this capability your applications programs can define, control, and access disc files and/or devices on the remote system by means of the following intrinsics. The intrinsics may appear in COBOL, FORTRAN, BASIC, and SPL programs when accessing remote HP 3000 processors. When accessing a remote HP 1000, they can be used only in FORTRAN or BASIC programs.

INTRINSICS			Requested Action
1000/3000		3000/3000	
Local HP 3000 Call to Remote HP 1000	Local HP 1000 Call to Remote HP 3000	Local HP 3000 Call to Remote HP 3000	
DCRET			Creates a file.
DNAME	FRNAM	FRENAME	Renames a specified file.
DPURG			Removes a file and directory entry
DOPEN	FOPEN	FOPEN	Opens a specified file.
DCLOS	FCLOS	FCLOSE	Closes a specified file
DREAD	FREAD	FREAD	Transfers one record from a file (sequential file on HP 3000)
	FRDIR	FREADDIR	Reads a record from a direct access file.
	FRLAB	FREADLABEL	Reads a user file label.
DWRIT	FWRIT	FWRITE	Transfers one record to a file (sequential file on HP 3000).
	FWDIR	FWRITEDIR	Writes a record to a direct access file.
	FWLAB	FWRITELABEL	Writes a user's file label.
	FSTMD	FSETMODE	Changes file access mode.
	FLOCK	FLOCK	Dynamically locks a file
	FUNLK	FUNLOCK	Dynamically unlocks a file.
	FUPDT	FUPDATE	Updates a record in a file.
DPOSN	FSPAC	FSPACE	Positions a file.
DAPOS			Positions a file to a known record.
	FPOIN	FPOINT	Resets pointer for sequential file.
	FRDSK	FREADSEEK	Prepares for reading a direct access file.
DCONT			Sends control request to peripheral device identified as a type 0 file.
	FCNTL	FCONTROL	Performs control of file or terminal device.
DWIND			Resets file to first record.
DLOCF	FINFO	FGETINFO	Returns file status.
	FCHEK	FCHECK	Requests details on file I/O errors.
DSTAT			Returns 250 bytes (125 words) of disc directory.
	FRLAT	FRELATE	Determines if file pair is interactive (requires human intervention for all input operations), or duplicative (echoes all input operations to a display device without intervention by the operating system software), or both.

In addition to the remote program access of standard MPE files, DS/3000 also allows you to access remote KSAM/3000 (Keyed Sequential Access Method) files using standard KSAM intrinsics.

Queued file transfer

HP 2026 terminal users can create files to be sent automatically to the host HP 3000. These files are placed in a queue until transmission to the HP 3000 is initiated from the HP 2026 system console. The system operator at the HP 2026 system console can specify additional files for transfer after the 2026/3000 link is established.

Files residing on the HP 3000 can be transmitted to the HP 2026 at the request of the system operator at the HP 2026 system console. Requests can be made for specific files, or all files named in the HP 3000 transfer queue may be sent to the HP 2026 after the 2026/3000 link is established. The queued files could, for example, be report listings from various operations performed on the HP 3000. Thus, multiple application programs on the HP 3000 can send information to the HP 2026 by placing their report listings in disc files and these disc file names in the queue for transfer to the HP 2026. Procedures may also be initiated in advance by the HP 2026 system operator to permit the HP 3000 to retrieve all of the HP 2026 queue files and send any queued HP 3000 files to the HP 2026 on request.

Remote data base access
The remote data base access feature of DS/3000 gives you the capability to access data bases on remote HP 3000 computer systems directly from other HP 3000 processors using the optional data compression feature. You can locate, report, and update data values in remote IMAGE/3000 data bases using local standard IMAGE/3000 subroutine calls or the QUERY/3000 data base inquiry facility.

Data base access using program-to-program communications is available with the 1000/3000 communications link. By first initiating a program on your local system, you can then run a remote program which contains standard IMAGE calls accessing and/or updating the remote IMAGE/1000 or IMAGE/3000 data base. The remote program then passes the requested information back to your local program.

Program-to-program communication
A set of procedures (called intrinsics) makes it possible for two or more programs residing on remote computer systems to interactively exchange information with one another. For a given data transfer between any two programs, one of the programs is the master, and the other is the slave. The master program opens the data link, initiates the slave program, and is always in control. The slave program merely responds to requests received from the master. In the 1000/3000 and 3000/3000 communication links this master/slave relationship is dynamic in that it is determined solely by the design of the applications program. Each HP 3000 or HP 1000 computer in the network can be used either as a master or as a slave or may operate in both modes simultaneously, depending on what you require for your specific application.

These procedure calls (intrinsics) can be used directly from a FORTRAN or Assembly language program on an HP 1000, and from a COBOL, BASIC, FORTRAN or SPL (System Programming Language) program on an HP 3000 for program-to-program communication.

User-created interface routines can be developed for the HP 3000 to allow this capability for other programming languages.

Program-to-program communication between two HP 3000 systems can also utilize the optional data compression feature.

Remote calls to the RTE system executive
FORTRAN and SPL programs on an HP 3000 can make calls to the system executive of a remote HP 1000 system to write to, read from, control, or get the status of I/O devices. Other calls can be used to schedule programs without wait, request system clock time, and set the execution interval or start time of a program.

Documentation
For further technical information, consult the following Hewlett-Packard manuals:

DS/3000 Reference Manual (32190-90001)

DS/3000 to DS/1000 Reference Manual for HP 3000 Users (32190-90005)

HP 2026 Data Entry/Communications System Reference Manual (22704-90001)

Training
Hewlett-Packard offers the following training course:

HP 36900E—This three day course provides the student with a working knowledge of the capabilities and features of DS/3000.

Program-to-Program Intrinsics

Intrinsic	Requested Action	Intrinsic	Requested Action
POPEN	Initiates and activates a slave process in a remote HP 1000 and initiates program-to-program communications with the slave program.	GET	Receives the next request from the remote master program.
PREAD	Sends a read request to the remote slave program asking the slave to send a block of data back to the master.	ACCEPT	Accepts (and completes) the request received by the preceding GET intrinsic call.
PWRITE	Sends a block of data to the remote slave program.	REJECT	Rejects the request received by the preceding GET intrinsic call.
PCONTROL	Transmits a tag field (containing user-defined control information) to the remote slave program and receives a tag back from the slave.	PCHECK	Returns an integer code specifying the completion status of the most recently executed slave program-to-program intrinsic. (HP 3000 only.)
PCLOSE	Terminates (kills) the remote slave process.	FINIS	Terminates communication with the master program. (HP 1000 only.)

HEWLETT hp PACKARD

Asynchronous Terminal Controller

model 30032B

The Asynchronous Terminal Controller (ATC) is designed to interface up to 16 terminals to the HP 3000 via the IOP bus. Since the system console occupies one terminal port, fifteen additional terminals may be hardwired or connected through modems before expansion to another controller is necessary. Terminals interfaced through the ATC can be configured to the Multiprogramming Executive (MPE) as on-line transaction processing terminals under user program control, or as log-on terminals accessing all the capabilities of the computer system.

Features

- **EIA RS232 compatibility to ensure industry wide acceptability**
- **Speed detection for up to six standard rates**
- **Automatic break detection**
- **Automatic answering**
- **Character assembly performed by hardware**
- **Full duplex (Echoplex), or half duplex transmission mode**
- **Parity generation and checking**

Documentation

For further technical information, consult the following Hewlett-Packard Manuals:

Asynchronous Terminal Controller Installation and Service Manual (30032-90004)

HEWLETT hp PACKARD

Asynchronous Repeater

model 30037A

The HP 30037A Asynchronous Repeater (AR) is a stand-alone device which converts standard RS232C communication signals to levels compatible with the HP family of terminals which operate in the asynchronous multipoint communication mode, such as the HP 2641A, HP 2645A, and HP 2648A display terminals.

Features

- **Operates in either asynchronous or point-to-point environment**
- **Extends maximum cable distance between the computer and the first asynchronous multipoint terminal to 2000 feet**
- **Extends maximum cable distance for asynchronous point-to-point communication to 4000 feet**

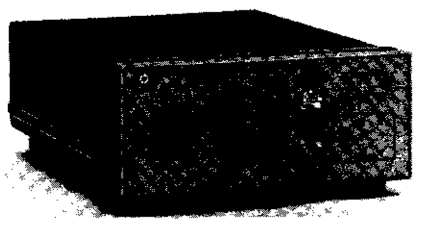
The addition of the AR permits the first directly connected asynchronous multipoint terminal to be located up to 2000 feet (609.6 meters) from the computer. This removes the 50 foot (15.24 meters) limitation imposed when an RS232C direct connect interface is used. In addition, each AR used can extend the distance between individual asynchronous multipoint terminals by 2000 feet (609.6 meters). The AR also permits extended asynchronous point-to-point communication with a maximum cable distance of 4000 feet (1219.2 meters). Two built-in diagnostic facilities, Self test and Loop test, are available for daisy chain verification.

Specifications

Communications facility: Twisted pair (22 AWG)
Data transmission rate: Up to 20k bps
Modulation: Baseband
Line conditioning: None required
Synchronization: Asynchronous
Data format: Serial
Operational mode: Full duplex
Terminal interface: RS232C
Line interface: Four/two twisted pair (22 AWG)
Diagnostic facilities: Self test and Loop test
Power requirements: 88 to 126 Vac
198 to 252 Vac (optional)
Line frequency: 47 - 66 Hz
Line current: .17 A RMS, .5 A Turn-on surge

Physical characteristics

Width: 8.3 inches (21.08 cm)
Height: 3.5 inches (8.89 cm)
Depth: 10.5 inches (26.27 cm)
Weight: 7 lb. 4 oz. (3.289 kg)



Environmental conditions

Since the Asynchronous Repeater may be used outside the immediate system area, it is not subject to the environmental specs of the system itself.

Temperature, free space ambient

Non-operating: −40° to +75°C (−40° to +167°F)
Operating: 0° to +55°C (32° to 131°F)
Humidity: 5% to 95% (non-condensing)

Documentation

For further technical information, consult the following Hewlett-Packard manual:

HP30037A Asynchronous Repeater Installation and Service Manual (30037-90003)

Synchronous Single Line Controller

for HP Distributed Systems Networks

model 30055A

The HP 30055A Synchronous Single Line Controller provides the hardware to link the HP 3000 computer systems to other computers in an HP Distributed Systems Network by means of modems. Also used in communications subsystems such as DS/3000, MRJE/3000, MTS/3000, and RJE/3000, the printed circuit card fits into a single I/O slot of the computer. The interface connects to a modem to facilitate data transfers at up to 9600 bits per second.

Features

- EIA RS232C and CCITT v.24 compatibility
- Compatible with Bell 201, 208, 209 and Milgo Modem 96-compatible synchronous modems using the same control protocol
- Half or full duplex operation
- Hardware data transfer'rates to 9,600 bits per second
- Double character buffering
- Special character recognition
- Compatible with IBM Binary Synchronous Communications protocol

Data set signals provided:

	PIN	RS232	v.24
Transmitted data	2	(BA)	103
Received data	3	(BB)	104
Request to send	4	(CA)	105
Clear to send	5	(CB)	106
Data set ready	6	(CC)	107
Signal ground	7	(AB)	102
Received line signal detector	8	(CF)	109
Transmitter signal element timing (DTE) (not RS232 Pin Out)	13	(DA)	113
*Secondary transmitted data	14	(SBA)	118
Transmit clock (DCE)	15	(DB)	114
*Secondary receive data	16	(SBB)	119
Receive clock	17	(DD)	115
Data terminal ready	20	(CD)	108.2
*Ring indicator	22	(CE)	125
Data signal rate selector (DTE)	23	(CH)	111

*Not supported by software

Specifications

Model: 30055A
Interface Level: EIA/CCITT
Operation Mode: Half or full duplex modems
Data trans. rate: Up to 9600 bps/ hardware rate
Char. buffering: 2
Prog. parity generation/checking: None, odd, even
Spec. char. recog.: Program selectable
Sync. char.: Program selectable
Power requirements: 3.5A (+5.0V), 0.095A (+12V), 0.007A (−12V)
Modem interface capability: Bell 201C, 208A and B, 209A, and Milgo Modem 96-compatible
(Purchaser must supply the modem.)

HEWLETT PACKARD

Hardwired Serial Interface

and Cabling for HP Distributed Systems Networks

model 30360A

The HP 30360A Hardwired Serial Interface card provides the hardware to link the HP 3000 computer systems to other computers in an HP Distributed Systems Network by means of a coaxial cable.

Features

- 2.5 M bits per second transfer rate up to 1000 ft./hardware data rate
- 4 software selectable channels
- Hardware transmission of an acknowledge word without program interruption
- Optically isolated reception
- Immediate line turnaround

Used for high-speed point-to-point data transfer, the interface fits into a single I/O slot of the computer. The maximum hardware data rate is 2.5M bits per second at up to 1000 feet. The card may also be used with cables up to 2000 feet long at 1.25 million bits per second (hardware rate).

The cable is isolated at the receiver end by an optically isolated gate. This enables long distance data transmission with high reliability against errors due to common mode noise or ground level shifting.

The HP 30360A has several other features including 4 selectable channels under system operator control, automatic data acknowledgement or handshaking via hardware, and CRC generation, transmission, and processing through hardware.

The CRC accumulation is a 16-bit feedback shift register that implements a 15th degree CRC polynomial.

The Hardwired Serial Interface must be inter-connected with 30220A cable kits. Several optional lengths are available in lengths from 25 to 2000 feet.

Disc Drives

models 7920, 7925

The HP 7920 family of disc drives provides exceptionally high performance, high reliability mass storage of formatted information. Their extremely fast data access rate using track-follower technology (25 ms average random seek time, 5 ms average track-to-track seek time) is among the fastest in the industry.

Features

- Fast-access, track-following technology
- Advanced serviceability features
- Constant spindle speed, independent of line frequency
- Extended maintenance periods through use of prefilter
- Exceptionally fast stop/startup speed
- Microprocessor-based controller with hardware error correction for enhanced data reliability
- Up to eight drives connected to the HP 3000 system controller
- Exceptional reliability over a wide range of environmental conditions

Designed for low noise operation and ease of service, the drives are all equipped with a compact, removable disc pack which consists of multiple platters. The upper and lower platters are protective. Interior platters have two surfaces. One interior surface on the disc is used for servo information to control the rapid and precise positioning of the actuator mechanism. Incorporation of positioning information on the disc pack provides for accurate track positioning over wide temperature variations. All other interior surfaces are used for formatted data storage.

Since up to eight drives in any combination may be connected to a single disc controller, a total of 960 megabytes of formatted storage capacity may be accessed per controller.

Operator switches and indicators
The outside operator's panel has five backlit indicators: Unit Select, Drive Ready, Read Only, Door Unlocked, and Drive Fault. There is one switch: Run/Stop.

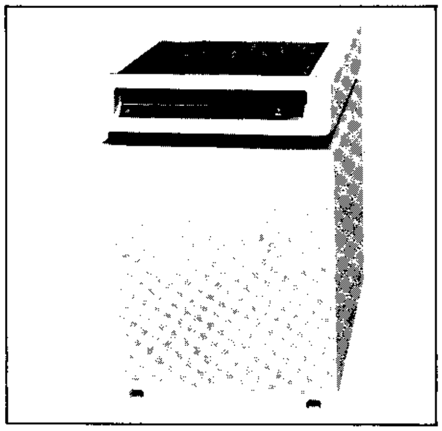
On the inside operator's panel a group of eight LED indicators light if the internal fault detection circuitry detects a fault condition. This advanced serviceability feature facilitates trouble shooting and reduces the time to diagnose and repair failures.

In the event of power failure, heads are retracted and carriage locked using energy from the filter capacitors, supplemented by the spindle motor acting as a generator.

Automatic error correction and detection
The error correction code (ECC) hardware increases data reliability and system availability by reducing the effects of media errors.

The ECC hardware and algorithm are together capable of correcting one single burst data error per sector, if the error is less than or equal to 32 bits. Every single burst data error greater than 32 bits but less than or equal to 48 bits will be detected without being miscorrected. For burst errors greater than 48 bits, 99.9999% are detected.

If the controller detects an error within a sector, it notifies the CPU at the end of that sector. The CPU then requests the location (displacement) of the error within the sector and three words of mask which are used to correct the record now in CPU memory. The controller calculates these masks from information accumulated in special registers. The three words are merged with data in CPU main memory to obtain a corrected record.



Specifications

	7920	7925
Dimensions		
Height:	32.5 in (82.5 cm)	32.5 in (82.5 cm)
Width:	19.65 in (50 cm)	19.65 in (50 cm)
Depth:	32 in (81.3 cm)	32 in (81.3 cm)
Shipping weight:	415 lb (188 kg)	420 lb (191 kg)
Seek time		
Track-to-Track:	5ms max	5ms max
Average random:	25ms	25ms
Maximum stroke:	45ms	45ms
Capacity		
	815 data tracks/surface 48 sectors/track 2 bytes/word 256 bytes/sector 12,288 bytes/track 5 data surfaces 50Mbytes/drive	815 data tracks/surface 64 sectors/track 2 bytes/word 256 bytes/sector 16,384 bytes/track 9 data surfaces 120Mbytes/drive
Rotation		
Speed:	3600rpm	2700rpm
Avg. rotational delay:	8.3ms	11.1ms
Data transfer rate		
Bits/second:	7,500,000	7,500,000
k-bytes/second:	937.5	937.5
Temperature		
Operating:	+10°C to +40°C (50°F to 104°F) rate of change not to exceed 20°C (36°F)/hr Optimum interchange and data transfer require that the disc drive be operated within ±15°C (±27°F) of the temperature at which the heads were aligned.	+10°C to +40°C (50°F to 104°F) rate of change not to exceed 20°C (36°F)/hr Optimum interchange and data transfer require that the disc drive be operated within ±10°C (±18°F) of the temperature at which the heads were aligned.

Interactive Display Terminals

models 2640B, 2640C, 2640N, 2640S, 2641A, 2645A, 2645C, 2645N, 2645R, 2645S

The HP 2640 series of interactive display terminals brings to your HP 3000 an unparaleled set of features designed to complement your system in all areas of your organization.

Features

- Enhanced high-resolution display
- Plug-in character sets
- Dynamically allocated memory
- Character/block mode
- Self-test
- Full editing capability
- Multi-task keyboard
- Off-screen storage with scrolling capability
- Programmable protected fields
- Inverse video for highlighting; and optional blinking, underline, half-bright
- Cursor addressability and positioning control tabulation
- Hard copy interface
- Multipoint asynchronous and synchronous communications capability

Enhanced high-resolution display
The terminals have a 5 inch by 10 inch rectangular display providing a 1,920 character capacity in 24 lines of 80 characters per line. The characters are formed by a 7 x 9 dot matrix generated in a 9 x 15 dot character cell. The high resolution of the 7 x 9 dot matrix is enhanced by dot shifting for precise character definition, and by the use of the enlarged character cell for wide character and line separation, underlining, line descenders, and inverse video. These display features are engineered to increase clarity and ease sessions at the terminal.

Plug-in character sets
Recognizing the demand for terminals that speak many languages and fill diverse sets of needs, the terminals have the capacity to include up to four 128-character sets resident concurrently in the terminal. Adjacent characters on the display may be from any of the four character sets and are available with the optional underline, blinking and half-bright feature.

Dynamically allocated memory
Because of the efficient linking memory organization (transparent to you) spaces to the right of the last character typed on a line are normally not stored in memory. Consequently, the basic terminal equipped with 1024 characters of display memory can store from 8 to 50 lines of information dependent on line length. Optional memory can expand this line capacity to a maximum of over 400 lines of information. Lines are viewed 24 at a time by using the roll up, roll down, next page, and previous page keys.

Pop-in modularity and expandability
The modular computer-like construction is designed for ease of service. Digital electronics are contained on printed-circuit cards that can be exchanged within the terminal; up to 14 cards can be accommodated to allow a choice of options. The combination of microprogramming and modularity means that the terminals can be expanded as new technologies and devices become available.

Microprocessor controlled
The operating characteristics of the terminal are controlled through firmware. The terminal's microprocessor manages memory allocation, data communications, keyboard scanning, and display control. This microprocessor implementation and the use of single bus architecture yield a terminal utilizing electronics and mechanics with a wide range of capabilities and potential for future enhancements.

Character/block mode
The terminals will operate character-by-character as a completely interactive terminal or are capable of operating on a block at a time. Text can be composed and edited locally, thus allowing you to verify and correct data before transmission to the CPU. Editing and CPU connect times are significantly reduced by such features as character or line insert and delete, cursor sensing and positioning control, programmable protected fields, off-screen storage with scrolling and page select capability, tabulation, transparent display control codes, 8 special function keys for user defined routines and a positionable memory protect.

Self-test
The terminals have been engineered for high reliability, ease of maintenance, testing, and rapid repair when needed. By depressing the TEST button on the keyboard you receive a Go/No-Go indi-

cation from results of a memory test, firmware test, and display verification.

Hard-copy interface
The terminals accommodate most RS232C-compatible serial printers or any HP-manufactured printer which uses the HP parallel interface. The serial printers are connected via the 13250A interface card, while the parallel printers use the 13238A duplex register interface card. Commands to print data can be initiated locally from the terminal keyboard or remotely from the CPU under control of a user's program. The MPE operating system does not recognize hardcopy devices connected to the terminals. All control of hardcopy devices is assumed by your program.

Display enhancements
You can highlight specific fields by using any of the 16 possible combinations of inverse video, blinking, underline and half-bright to emphasize areas of the display.

Optional features

Fully integrated mass storage (2641, 45)
Benefits of mass storage: Many operations normally requiring connection to a computer system can now be done offline with the terminal. Optional dual in-board cartridge tape units allow batching of information, and add extensive stand-alone capabilities which can significantly reduce user time; conserve both computer and communications resources; provide a tape backup; and, very importantly, allow the terminal to keep on working even when a computer is unavailable.

Mini-cartridge: Highly reliable, interchangeable mini-cartridges each provide the capacity for many hours of typing, up to 110,000 characters of storage formatted in variable length records and files. The tape units feature rapid data transfer and bi-directional high-speed search to access any file in seconds. The mini-cartridge is ideally suited for storing data, forms, programs or text, and is an excellent substitute for paper tape. The HP 3000 FCOPY utility may be used to transfer files to and from mini-cartridges to other HP 3000 peripherals.

User-defined soft keys (2641, 45)
Each of the eight special function keys can be easily used to issue a user-defined string of up to 80 characters or several control sequences stored in the 2645A. This feature allows the keyboard to adapt to specialized applications, and can considerably simplify use of the keyboard and result in greater efficiency—each soft key performs the operations of several key sequences. For example, the soft keys could issue frequently used programming sequences, search for files, aid forms construction for data entry, dynamically configure the terminal, or issue instructions to the operator, computer or both.

Character sets
APL: With the 2641, the APL basic set and APL overstrike set are both available for development and display of APL programs. The clarity and visual representation of the characters are facilitated by the 9 by 15 dot character cell. The basic set is comprised of the 96 APL graphics, 64 overstrike graphics and includes the ability to display control codes which help in debugging programs.

Also standard in the 2641A is a 64 character Roman set which is optionally expandable to include lowercase capability. This second set allows the terminal to be switched from APL to the ASCII mode where the 2641A can then double as a fully Teletype-compatible terminal.

Math: The optional Math set is available to include equations right in the text that's composed on the screen. This feature helps to impart realism and assists in understanding technical material.

Line drawing: The optional Line Drawing set assists users in defining forms that appear like those normally filled out in your company's day-to-day operations. Now operator errors can be minimized due to the familiarity associated with the constructed forms.

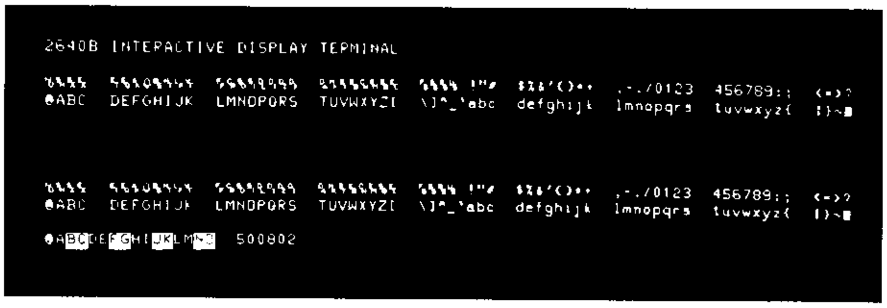
Norwegian: The 2640N and 2645N have the Danish/Norwegian set in place of the standard uppercase set. The non-standard characters must be handled by application programs.

Large character set: With this option, characters approximately ½ in. tall can be constructed. This is useful for warehousing operations where, for example, fork-lift drivers may have to read computer derived instructions from a distance. Application programs are responsible for constructing characters from supplied segments.

Cyrillic: The 2640C has 128 character Cyrillic set in addition to standard uppercase set. The non-standard characters must be handled by application programs.

Swedish: The 2640S and 2645S have 64 character Swedish (Finnish) set in place of standard uppercase set. The non-standard characters must be handled by application programs.

Arabic: The 2645R has a combined 64 character Roman/31 character Arabic character set in place of the standard uppercase set. The non-standard characters must be handled by application programs.



```
2640B INTERACTIVE DISPLAY TERMINAL

                                          !"#  $%&'()*+  ,-./0123  456789:;  <=>?
@ABC  DEFGHIJK  LMNOPQRS  TUVWXYZ[  \]^_`abc  defghijk  lmnopqrs  tuvwxyz{  |}~█


                                          !"#  $%&'()*+  ,-./0123  456789:;  <=>?
@ABC  DEFGHIJK  LMNOPQRS  TUVWXYZ[  \]^_`abc  defghijk  lmnopqrs  tuvwxyz{  |}~█

@ABCDEFGHIJKLMNO   500802
```

Actual photograph of 2640B display

Special purpose terminal
The 2649A is available for OEMs and others who wish to design a custom terminal from the individual features of the 2645A—developing special assembly language programming for the terminal's microprocessor.

Specifications

General

Screen size: 5" x 10" (127 x 254 mm)

Screen capacity: 24 lines x 80 columns (1,920 characters)

Character generation: 7 x 9 enhanced dot matrix; 9 x 15 dot character cell; non-interlaced raster scan

Character size: 0.097" x 0.125" (2.46 x 3.175 mm)

Cursor: Blinking-underline

Display modes: White on black; black on white (inverse video) blinking, half-bright, underline

Refresh rate: 60 Hz (50 Hz optional)

Tube phosphor: P4

Implosion protection: Bonded implosion panel

Memory: MOS, ROM: 22k bytes (program); RAM; standard 4096 bytes; 12 kilobytes maximum (16k including maximum data communications buffer)

Keyboard: Detachable, additional control and editing keys; ten-key numeric pad; cursor pad; multispeed auto-repeat, N-key roll-over; 4 ft. cable (1.22 m).

Cartridge tape (optional): Two mechanisms
Read/write speed: 10 ips
Search/rewind speed: 60 ips
Recording: 800 bpi
Mini cartridge: 110 kilobyte capacity (maximum per cartridge)

Data communications

Data rate: 110, 150, 300, 1200, 2400 baud, and external switch selectable (110 selects two stop bits)

Standard asynchronous communications interface: EIA standard RS232C; compatible with Bell 103A modems, Bell 202S/T modems with reverse channel, and Vadic VA3400 series modems.

Transmission modes: Full or half duplex, asynchronous
Operating modes: On-line; off-line; character; block
Parity: Switch selectable; even, odd, none

Terminals at a Glance

	FEATURES	2640B	2640N 2640S 2640C	2645A	2645N 2645S 2645R	2641A	2648A
Base Level Features	Two Tape Cartridge Drives	—	—	Opt.	Opt.	Opt.	Opt.
	Inverse Video	Std.	Std.	Std.	Std.	Std.	Std.
	Underline, Blinking, ½ Brite, Line Drawing	Opt.	Opt.	Opt.	Opt.	Std.	Opt.
	Soft Keys	—	—	Std.	Std.	Std.	Std.
	Lower Case	Opt.	Opt.	Std.	Opt.	Opt.	Std.
	Maximum Memory	8kb	8kb	12kb	12kb	12kb	12kb
	Math Character Set	Opt.	Opt.	Opt.	Opt.	Opt.	Opt.
	Large Character Set	Opt.	Opt.	Opt.	Opt.	Opt.	Opt.
	Roman Character Set	Std.	—	Std.	—	Std.	Std.
'International' Character Set Replaces 'Roman' to Produce a Unilingual International Terminal	Norwegian/Danish	—	N Model	—	N Model	—	—
	Swedish/Finnish	—	S Model	—	S Model	—	—
	Arabic	—	—	—	R Model	—	—
'Roman' plus 'International' Character Set Produces a Bilingual Terminal	Cyrillic	—	C Model	—	—	—	—
Advanced Features	APL Character Set	—	—	—	—	Std.	—
	Graphics	—	—	—	—	—	See Note

Note: The 2648A is supported on the HP 3000 as a 2645A.

HEWLETT hp PACKARD

Printing Terminal

model 2635A

The HP 2635A is a dot matrix hard copy terminal featuring high throughput and a wide range of user-definable printing functions. Utilizing Hewlett-Packard's advanced SOS microprocessor technology, the HP 2635A can serve either as the HP 3000 system console or as an optional terminal.

Features

- High throughput
- High resolution dot matrix characters
- 128 USASCII character set
- Long life print head
- Cartridge ribbon
- Single or multipart forms
- 8 channel fixed VFC
- Special user features
  - variable horizontal pitch
  - variable vertical pitch
  - horizontal tabulation
  - display functions
  - self test
  - paper out detection
  - auto line feed
  - auto-underline
  - selectable view mode
- Typewriter style keyboard
- Numeric keypad
- RS232C interface

High throughput

The HP 2635A has a print speed of 180 characters per second. Print speed, however, is only one of several factors which affect throughput. The HP 2635A optimizes throughput by printing bi-directionally, suppressing leading and trailing blanks, skipping over embedded blanks, and returning the carriage at high speed. The HP 2635A utilizes the newly developed HP CMOS/SOS microprocessor to optimize printing functions, data manipulation, and other control functions.

High resolution dot matrix

The HP 2635A utilizes a high resolution seven column by nine row dot matrix character cell for crisp, clear character formation. The nine high matrix allows true descenders, commas, semi-colons, and underlining. Dot matrix is especially well suited for multipart forms; since each dot is formed with equal intensity, the sixth copy has surprisingly good character resolution.

128 USASCII character set

Most terminals will print the 64 USASCII set (upper case alphabet, numbers and symbols), a few will print the 96 USASCII set (upper and lower case alphabet, numbers and symbols), but the HP 2635A is unique in allowing the user to print the entire 128 USASCII character set (upper and lower case alphabet, numbers, symbols, and control codes).

Long life print head

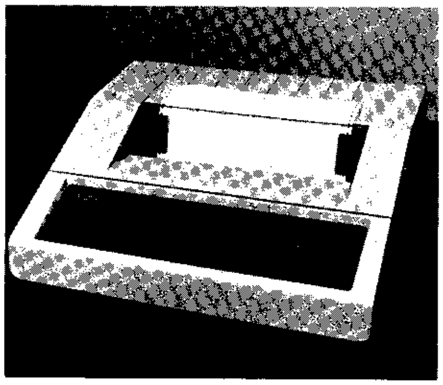
The nine-wire print head is rated at 130 million characters, making replacement an infrequent task. When replacement is required, this self-aligning head is easily replaced using the hex key supplied with the terminal.

Cartridge ribbon

The cartridge ribbon has a long life of more than 10 million characters (about 3 months of normal use). This is made possible by an internal mobius loop design that permits the use of the entire ribbon area. When a ribbon change is necessary, it is a quick, simple and clean task.

Single or multipart forms

Up to 6 part forms (up to 43mm thick) may be accepted by the HP 2635A. Due



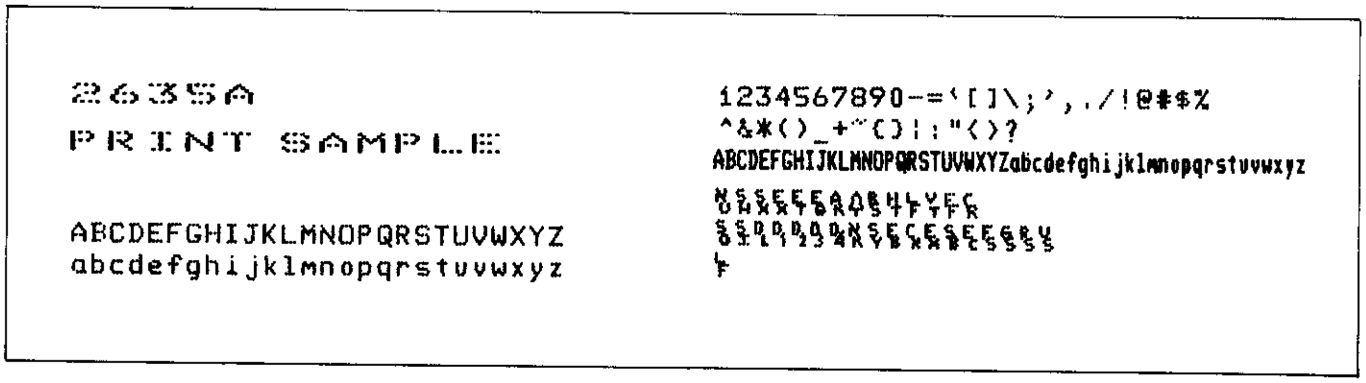
to a unique head design, adjustment for forms thickness variations between .08mm (.003") and 43mm (.017") is usually not required. A lever adjustment on the print head carriage is provided to allow optimization of print quality on various forms.

8 Channel fixed VFC

Vertical forms control is provided through an eight channel fixed electronic VFC. There is no paper tape or paper tape reader to worry about, the VFC is located in read only memory (ROM). The eight channels provide the following forms advance capability:

		Line # @ 6 LPI
• Channel 1	top of next page	0
• Channel 2	bottom of current page	59
• Channel 3	single space	0, 1, 2, ... 59
• Channel 4	next double space	0, 2, 4, ... 58
• Channel 5	next triple space	0, 3, 6, ... 57
• Channel 6	next half page	0, 30
• Channel 7	next quarter page	0, 15, 30, 45
• Channel 8	next tenth space	0, 10, 20, ... 50

Note: "Page" is defined to be 11 inches. "Space" is defined to be 1/6 inch, unless 8 lpi mode is being used, in which case "space" is defined as 1/8 inch.



```
2635A
PRINT SAMPLE

ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
```

```
1234567890-=`[]\;',./!@#$%
^&*()_+~{}|:"<>?
ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz
```

Special user features

By entering at the keyboard specific keystrokes of control sequences or embedding these sequences in the data stream, the user can customize 2635A output.

- Horizontal printing in any of three modes: compressed (16.7 characters per inch); normal (10 cpi); expanded (5 cpi).

  Normal mode is the standard on most terminals and will provide 136 characters per line on 14-7/8 inch wide paper or saving paper (132 characters per line on 8-1/2 inch wide paper). Expanded mode creates characters which are useful for titles and headings. The three different modes may be intermixed on a single line.
- Auto-underlining in all print modes provides emphasis to key words and phrases in your output.
- Vertical spacing is variable at vertical pitches of 1, 2, 3, 4, 6, 8, or 12 lines per inch.
- A view mode allows the user to see the last character printed by moving the print head to the right after a keyboard entry.
- In addition to control sequence operation, certain special features may also be triggered by setting buttons or switches on the 2635A.
- Horizontal tabs may be set, cleared, and accessed in any of 227 print positions.
- In display functions mode, embedded control sequences are printed rather than executed. Carriage return (control M), on-line (escape M), off-line (escape O), and display functions off (escape Z) are exceptions; they are displayed and executed.

Specifications

Reliability

An extensive reliability testing program was implemented on the 2630 family, in which 330 million characters were printed on each of 35 machines, for a total of 11.5 billion characters. The preliminary results of this test show that the mean number of printed characters between failures, (MCBF) excluding print head and ribbons, exceeds 250 million.

The 2635A is recommended for 1 million characters per day, resulting in a Mean Time Between Failures (MTBF) of approximately 1 year. Basic monthly maintenance charges are based on this level of usage.

RS232C interface

The standard interface is an EIA standard RS232C asynchronous interface for use with 103 type modems. The user may optionally add 202 type modem control to the RS232C interface.

Data Rate: 110, 150, 300, 1200, 2400 baud and external; switch selectable; (110 selects two stop bits; external allows baud rates to 9600.)

Transmission Mode: full duplex

Parity: odd, even, or none; switch selectable

Connector: 25 position, similar to Cinch DBM (D subminiature socket)

Power requirements

Input voltage: 100, 120, 220, 240 volts AC (+10% to 12%) selectable from rear panel 48-62 Hz

Power consumption: 140 VA max non-printing
265 VA max printing

Environmental conditions

Temperature, free space ambient:
−40° to 75°C (−40° to 167°F) non-operating
0° to 55C° (32° to 131°F) operating

Humidity: 5% to 95% RH (non-condensing) excluding media

Vibration and shock

Vibration: .38mm (.015") pp. 5-55Hz, 3 axes

Shock: 30G, 11ms, 1/2 sine shock pulse. Also type tested to qualify for normal shipping and handling in original shipping container.

Physical specifications

595mm (23.1")D x 640mm (25.2")W x 215mm (8.5")H
23.5 kg (56 lb) net weight (without pedestal)
31 kg (72 lb) ship. weight (without pedestal)

Digital Magnetic Tape Drives

Hewlett-Packard Digital Magnetic Tape Units are high performance, reliable magnetic tape drives for use in an HP 3000 computer system. IBM compatible NRZI recording mode is used at a density of 800 cpi. High packing density and data transfer rates are achieved by using ANSI-compatible 1600 cpi phase encoded data electronics. Data written on any IBM or ANSI-compatible equipment can be read. Four tape drives can be operated from a single controller.

Features

- **Fast data transfer – up to 36k bytes/sec (NRZI) – up to 72k bytes/sec (phase encoded)**
- **9-track configuration**
- **800 bits/cpi (315 cm), NRZI electronics**
- **1600 bits/cpi (630 cm), phase encoded data electronics**
- **45 ips (114 cm/s) read/write, 160 ips (406 cm/s) rewind/fast forward**
- **Dynamic braking**
- **Up to 10½ inch (26.7 cm) reels**
- **IBM/ANSI compatible**

Reel motors provide direct drive, eliminating troublesome belts and pulleys. Tape tensioning is performed by photo-resistive controlled tension arms, eliminating the need for vacuum system components. Head assemblies consist of read stack, write stack and full width erase head.

Specifications

Number of tracks
Nine

Read/write speed
45 ips (114 cm/s)

Density
800 bits/cpi (315 cm), NRZI electronics (7070B)
1600 bits/cpi (630 cm), phase encoded electronics (7970E)

Data transfer rate
36,000 characters per second maximum, NRZI electronics (7970B)
72,000 characters per second maximum, phase encoded electronics (7970E)

Write enable
Supply reel write enable ring and switch. Ring removal precludes writing.

Reel diameter
Up to 10½ inches (26.7 cm)

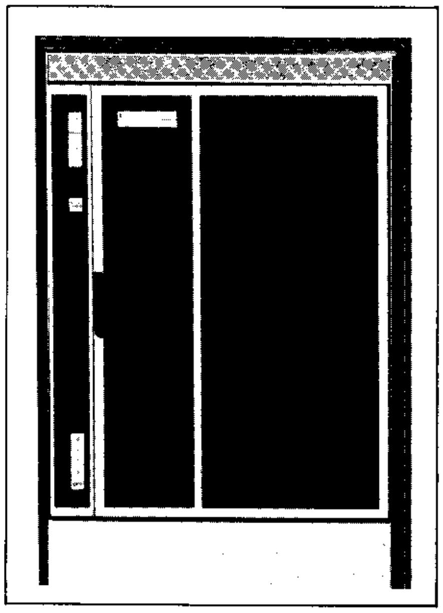
Tape (Computer Grade)
Width: 0.5 inches (12.7 mm)
Thickness: 1.5 mils (0.038 mm)

Rewind speed
160 ips (406 cm/s)

Start/stop times
8.33 ms (read-after-write) at 45 ips (114 cm/s)

EOT and BOT reflective strip detection
Photoelectric, IBM compatible

Operator control panel
Reset Switch: Stops tape travel in any mode and returns unit to local control



Rewind Switch: Rewinds tape at 160 ips (114 cm/s)
On-Line Switch: Places unit under remote control
Load Switch: Initiates loadpoint (BOT) search
Write Enable Indicator: Illuminated when write enable ring is installed on the supply reel

Physical characteristics
The 7970B/E is racked in a standard HP 3000 cabinet. The shipping weight is 190 lbs (86.2 kg)

HEWLETT hp PACKARD

Low Speed Line Printer

model 2613A

The HP 2613A Line Printer represents an optimized combination of high print quality, moderate speed, and low price in a drum printer. This unit is ideally suited to applications requiring a medium speed printer.

Features

- **Prints 136 columns of 64 characters at 300 lines per minute**
- **Prints 136 columns at 240 lines per minute with 96 character set**
- **12-channel vertical forms control**
- **Multiple copies – prints up to 6-part forms**
- **64-character or 96-character ASCII available**
- **6 or 8 lines per inch – operator selectable**

The unit attains a print rate of 300 lines per minute when printing a full 64-character set in a 136-column format. Print rate is 240 lines per minute for 96-character set configurations.

A 12-channel Vertical Format Unit allows convenient and efficient printing in predetermined formats such as business forms. The VFU uses industry standard control tapes. Under program control the VFU may be commanded to slew to the next hole in a given channel of the tape or to slew an absolute number of lines from 0 to 15.

A switch allows the operator to choose 6 or 8 lines per inch (60 or 80 lines/page respectively). A matching VFU tape will synchronize the forms to the VFU.

An ASCII 64-character set is standard with 96-character ASCII optional.

Six part forms are easily handled by the HP 2613A. Forms alignment is simplified by horizontal and vertical paper alignment guides. Fine vertical paper alignment adjustments can be made while printing. A paper receptacle is provided.

Differential line drivers allow the printer to be located up to 500 feet from the computer for flexibility of installation design.

Paper and ribbon loading are facilitated by a swing hinge which allows the drum-gate to be opened 90 degrees.

All EDP category safety requirements are met for equipment listing by Underwriters Laboratories.

Specifications

(Assumes 20% duty cycle)

Paper feed
One set of pin tractors above drum, friction paper tensioner below drum

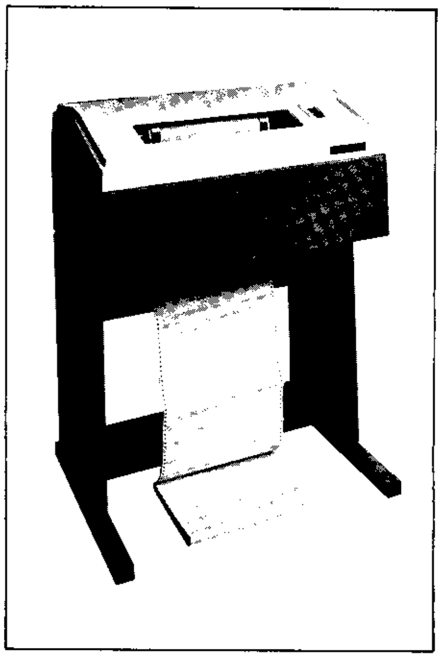
Line advance: 50 milliseconds
Slew rate: 20 inches (50.8 cm) per second

Paper dimensions
Standard fanfold, edge punched paper 4 inches (10.2 cm) to 16 inches (40.6 cm) wide

Paper type
Single copy, 15 lb. bond minimum weight
Multi-copy up to 6 parts, 12 lb. bond with single shot carbon

Character drum
Characters per Line: 136
Character type: Open Gothic
Symbol size (typical): 0.095 inch (2.4 mm) high, 0.064 inch (1.65 mm) wide

Vertical format unit
Number of channels: 12
Addressing: Slew to next hole in channel "n" or slew 0 to 15 lines



Throughput data
Duty cycle: 20%
Print cycle: 40% (per page)
(425 pages/hour for heavy print periods. 681 pages/8 hour day). Refer to "A Guide to Hewlett-Packard Printers" for detailed performance data.

Printing speeds

Char. Set	Drum Speed	136 Char. Line Per Minute
64	1200	300
96	800	240

Physical characteristics
Height: 45 inches (114.5 cm)
Width: 33 inches (83.8 cm)
Depth: 22 inches (55.9 cm)
Weight: Net: 340 lb (154.2 kg)
Shipping: 365 lb (165.6 kg)

HEWLETT PACKARD

Medium Speed Line Printer

model 2617A

The HP 2617A line printer represents an optimized combination of high print quality, speed and low price in a drum printer subsystem. This unit is ideally suited for applications requiring a medium speed printer.

Features

- Prints 136 columns of 64 characters at 600 lines per minute
- Prints 136 columns of 96 characters at 436 lines per minute
- 12 channel Vertical Forms Control
- Single through 6 part forms
- 64 or 96-character ASCII drums
- 6 or 8 lines per inch – operator selectable

The HP 2617A attains a print rate of 600 lines per minute when printing a full 64 character set in a 136 column format. Print rate is 436 lines per minute for 96 character set configurations.

A 12 channel Vertical Format Unit allows convenient and efficient printing in predetermined formats such as business forms. The VFU uses commonly available control tapes. Under program control the VFU may be commanded to slew to the next hole in a given channel of the tape or to slew an absolute number of lines from 0 to 15. Two VFU channels can be monitored by the computer from user-written SPL procedures. This feature may be used to further simplify forms control programming.

A switch allows the operator to choose 6 or 8 lines per inch. An appropriate VFU tape should be mounted on the printer.

Six part forms are easily handled by the HP 2617A. Forms alignment is simplified by horizontal and vertical paper alignment guides. Fine vertical paper alignment adjustments can be made while printing. A paper receptacle is provided on the rear of the printer.

Operation at high paper rates and low humidity is aided by inclusion of an electronic static eliminator.

An ASCII 64 character set is standard with 96 character ASCII optional. Special character sets including non-English language and OCR fonts are available as specials. Consult your local HP Sales Representative for information. This printer meets all safety requirements for listing by Underwriters Laboratories in the EDP equipment category.

Specifications

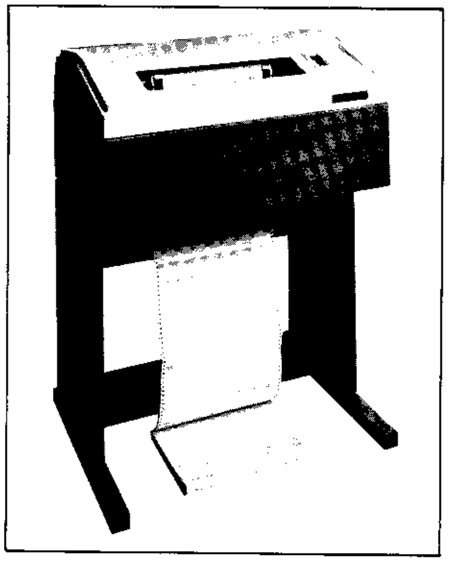
(Assumes 40% duty cycle)

Paper feed
One set of pin tractors above drum, friction paper tensioner below drum.
Line advance: 25 milliseconds
Slew rate: 25 inches (63.5 cm) per second

Paper dimensions
Standard fanfold, edge punched paper 4 inches (10.2 cm) to 16 inches (40.6 cm) wide

Paper type
Single copy, 15 lb. bond minimum weight
Multi-copy up to 6 parts, 12 lb. bond with single shot carbon

Character drum
Characters per line: 136
Symbol size (typical):
0.095 inch (2.4 mm) high
0.065 inch (1.65 mm) wide



Vertical format unit
Number of channels: 12
Addressing: Slew to next hole in channel "n" or slew 0 to 15 lines
Status channels: 9 and 12 may be read back by user programs to determine form position

Throughput data
Duty cycle: 40%
Print cycle: 40% per page
850 pages/hour for heavy print periods; 2747 pages average throughput/8 hour day. Refer to "A Guide to Hewlett-Packard Printers" for detailed performance data.

Printing speeds

Char. Set	Drum Speed	136 Char. Line Per Minute
64	800	600
96	533	436

Dimensions
Width: 33 inches (83.8 cm)
Depth: 26 inches (66 cm)
Height: 45 inches (114.5 cm)
Shipping Weight: 415 pounds (188.2 kg)

High Speed Line Printer

model 2618A

The HP 2618A line printer provides high speed, high quality printed output, and quiet operation. This printer is ideally suited for applications requiring high printing speed.

Features

- **Prints 132 columns at rates up to 1800 lines per minute with 64-character set**
- **Prints 132 columns at rates up to 925 lines per minute with 96-character set**
- **12-channel vertical forms control**
- **6 or 8 lines per inch – operator selectable**
- **Both 64-character and 96-character ASCII available**
- **Multiple copies – prints up to 6-part forms**
- **Electronic static eliminator for improved forms stacking**

With a 64-character set, this unit prints at 1250 lines per minute. When printing a subset of 36 consecutive characters, 1800 lines per minute is achieved. Printing rate is independent of line length.

Two operator selectable drum speeds are incorporated. The higher rpm provides maximum printing speeds while the lower rpm provides enhanced print quality. Data is stored in a full line buffer prior to printing.

A 12-channel Vertical Format Unit uses industry standard control tapes. Under program control, the VFU may be commanded to slew to the next hole in a given channel of the tape or to slew an absolute number of lines from 1 to 15.

A single adjustment allows the operator to choose 6 or 8 lines per inch. A matching VFU tape will synchronize the forms to the VFU.

An ASCII 64-character set is standard and an ASCII 96-character set is optional.

Six-part forms are easily handled by the printer. Forms alignment is simplified by horizontal and vertical paper alignment guides. Fine vertical paper alignment adjustments can be made while printing.

Printer controls have been minimized and are easy to read. A swing hinge allows the drum-gate to be opened 90 degrees to facilitate both paper and ribbon loading.

An electronic static eliminator is provided to counteract charge build-up on forms.

Specifications

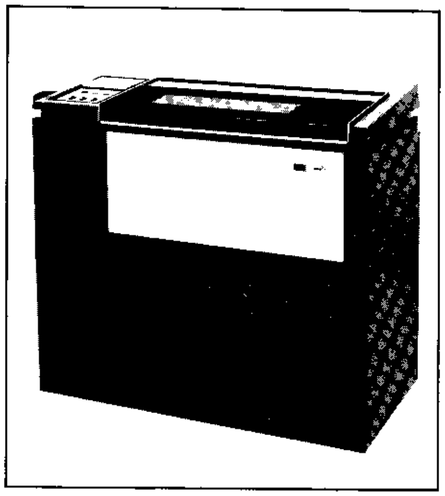
(Assumes 40% duty cycle)

Paper feed
Two sets of pin tractors
Line advance: 14 milliseconds (maximum)
Slew speed: 35 inches (89 cm) per second

Paper dimensions
Standard fanfold, edge punched paper 4 to 19 inches wide. (10.2 cm to 48.3 cm).

Paper type
Single copy, 18 lb. bond minimum weight. Multi-copy up to 6 parts, 12 lb. bond with single-shot carbon.

Character drum
Characters per Line: 132
Character type: Open Gothic
Symbol size (typical):
0.095 inch (2.4 mm) high
0.065 inch (1.65 mm) wide



Vertical format unit
Number of Channels: 12
Addressing: Slew to next hole in channel "n" or slew 0 to 15 lines

Throughput data
Duty cycle: 40%
Print cycle: 40% per page
(1610 pages/hour for heavy print period. 5144 pages average throughput/day). Refer to "A Guide to Hewlett-Packard Printers" for detailed performance data.

Printing speeds

Characters On Drum	Drum Speed Switch Position	#Consecutive Characters Printed	Printer Speed Lines/Minute 132 Columns
64	1800	1-36/37-64	1800/1250
64	1200	1-46/47-64	1200/925
96	1200	1-67/68-96	1200/925
96	800	1-76/77-96	800/675

Physical characteristics
Height: 46 inches (116.8 cm)
Width: 48.5 inches (123.2 cm)
Depth: 36.5 inches (92.7 cm)
Weight: Net: 800 lb. (362.9 kg)
Shipping: 900 lb. (409 kg)

Card Reader Subsystem

model 30106A

The HP 30106A subsystem provides dependable medium speed card reading capability. A vacuum pick mechanism is used in conjunction with riffle air for ease of card picking and minimum card wear. This technique also permits extremely high tolerance to damaged or worn cards. The card track is very short so that at no time is more than one card in motion.

Features

- **Reads 600 punched cards per minute**
- **Vacuum card picking**
- **Slant-top design for smooth card flow**
- **Straight-through card-track for long card life**
- **Automatic feed**
- **1000 card hopper/stacker**

The many checking features of the reader insure safe, dependable operation. These include light/dark check, motion check, pick check for stapled cards, and hopper checks.

Specifications

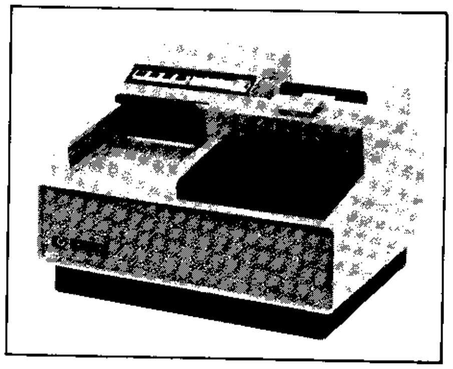
Card rate: 600 cards per minute
Card type: Standard 80-column EIA card
Hopper/stacker capacity: 1000 cards
Light source: Infrared light emitting diodes
Read station: Photo transistor, 12 bits simultaneously
Controls: Stop, Reset, End of File, Power
Indicators: Read Check, Motion Check, Pick Check, Hopper/Stacker

Data formatting

The HP 3000 interface controller provides Hollerith to ASCII conversion with packing and column binary conversion (each column plus four leading zeros packed into two bytes).

Physical characteristics

Height: 15.5 inches (39.4 cm)
Width: 23-1/16 inches (58.6 cm)
Depth: 18 inches (45.7 cm)
Shipping weight: 100 lbs (45.4 kg)



HEWLETT PACKARD

Card Reader Punch Subsystem

model 30119A

The HP 30119A subsystem provides complete punched card I/O capability in a single peripheral device, fully buffered, on-line 80-column card reading, punching, and printing capabilities for use with HP 3000 computers. Off-line data record (keypunch) capability is optionally available. Reading rate is 175 to 200 cards per minute. Punch/print rate is 45 to 75 cards per minute, depending on the number of columns involved. A fast slew rate allows rapid skipping of columns and fields not requiring punching or printing.

Features

- **On-line 80-column card reading, card punching, and card printing**
- **Read, punch, and print functions independently controlled by the computer system**
- **Individual data storage buffers for each function**
- **Dual input hoppers and output stackers selectable under program control**
- **Provides software-driver plus hardware interface for use with HP 3000 computers**

Separately buffered and independently controlled read, punch, and print functions may be utilized. This feature allows operations such as verification of previously punched cards, printing of information different from that punched, or duplication of existing cards.

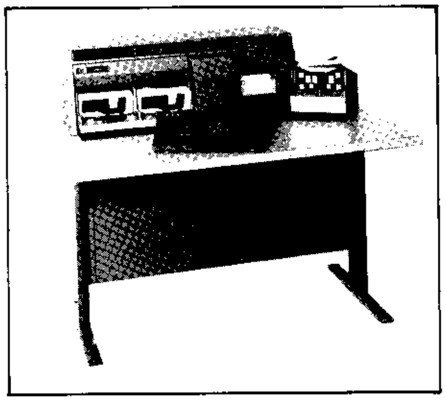
Primary and secondary input hoppers are provided with capacities of 600 and 400 cards respectively. Under program control, cards may be selected from either input hopper and directed to either 400-card output stacker following reading, punching, or printing. This feature eliminates the need for interfiling of blank cards in decks to be duplicated, provides for automatic separation of original and duplicate cards, and allows for sort/merge operations under control of user's programs. Input and output punched code format is Hollerith.

The printer employs a modified ASCII 64-character set as standard. A 64-character Swedish/Finnish character set can be ordered as a special option.

Selection of optional configuration 002, provides the user with the following powerful off-line capabilities:

- Complete off-line preparation and on-line input/output operations in a single unit increases the cost effectiveness of the subsystem.
- A comprehensive control panel allows the operator to select ON-LINE (computer controlled) operation or OFF-LINE (operator controlled) functions. OFF-LINE functions include card format program preparation, program loading, punching, verifying, reproducing, or printing. Included are illuminated indicators for machine status, error and column number.
- Keyed data is stored in an 80-character buffer memory prior to punching or printing. Backspace and Erase keys may be used to edit this data by character, field, or record.

Control programs may be prepared by the operator. These programs define the format for various card operations by designating field boundaries, data type for each field (alpha or numeric) and columns to be skipped. Memory is provided in the unit for storage of up to 4 programs, each of which is keyboard selectable. Reversion to a designated program is automatic following temporary selection of any other program.



When an error has been corrected during verification, the errored card is automatically placed in the unused stacker and a blank card fed and punched with the corrected data.

Specifications

Code compatibility
Hollerith (ANSI X3.26-1970)

Character set (printer)
Uppercase modified ASCII standard
Uppercase Swedish/Finnish available as special order

Performance
Card type: 80 column
Reading rate: 200 cards per minute
Punching and/or printing rate: 45-75 cards per minute depending on number of columns
Input hopper capacity, primary: 600 cards
Secondary: 400 cards
Output stacker capacities, both: 400 cards

Physical characteristics
Height: 40 inches (102 cm)
Width: 42 inches (107 cm)
Depth: 27 inches (69 cm)
Approximate shipping weight: 350 lb. (158.9 kg)

HEWLETT PACKARD

Tape Punch Subsystem

model 30105A

The HP 30105A Tape Punch Subsystem provides a compact and quiet-running paper, plastic, and mylar tape punching facility.

Features

- Compact and quiet-running
- Punches tape at 75 characters per second
- Punches both paper tape and mylar tape

Specifications

Punch speed
75 characters per second, asynchronous

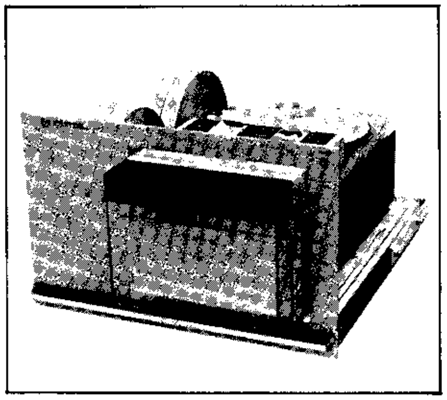
Tape type
Paper, mylar or plastic

Tape width
Standard 5 level 11/16 inch (17.5 mm) and 8 level 1 inch (25.4 mm)

Tape thickness
Paper: 0.003 to 0.005 inch (0.08 to 0.13 mm) oil-base or dry
Mylar: 0.003 to 0.004 inch (0.08 to 0.10 mm)
Plastic: 0.003 to 0.0045 inch (0.08 to 0.11 mm)

Physical characteristics
Height: 10-1/2 inches (26.7 cm)
Width: 16-3/4 inches (42.5 cm)
Depth: 21-3/16 inches (53.8 cm)
Mounting: Fits in HP 3000 cabinet under existing magnetic tape
Shipping weight: 47 lb (21.3 kg)

Operating controls
Power On, DC On, Tape Feed, Ext (external interrupt), Feed Holes, Code Holes



HEWLETT PACKARD

Punched Tape Reader Subsystem

model 30104A

The HP 30104A Punched Tape Reader Subsystem is a high-speed device for the error-free reading of both dry and oil-base tape.

Features

- **Reading speed to 500 characters per second**
- **Error-free reading of both dry and oil-base tape without adjustment**
- **Simple operation and rugged construction for long life**

The high speed of the reader permits rapid read-in while offering the economy and versatility of punched tape input. A significant advantage in reading accuracy is also provided by using a compensating sensing technique. Data reliability is enhanced as each character is read only once with no overshooting of characters. Positive feedhole control and a reliable clutch/brake mechanism ensure that the tape will stop on the character that initiates the stop. Simple operation, rugged construction and electrically-conservative design ensures long life at top performance.

Specifications

Reading speed
500 characters per second (415 characters per second when operated from 50 Hz power)

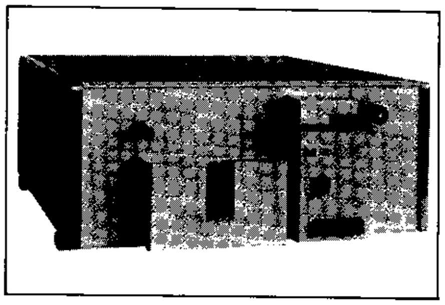
Reading technique
Photoelectric, character-by-character

Tape
Code: 8 level code
Width: 1 inch (2.54 cm)
Material: Any material with less than 60% transmissivity

Start/stop times
Start time: Less than 6 milliseconds
Stop time: Less than 500 microseconds (stops on character)

Controls
Power, Load, Read, Manual Advance

Physical characteristics
Height: 7 inches (17.8 cm)
Width: 17 inches (43.2 cm)
Depth: 16 inches (40.6 cm), not including panel controls and connectors
Shipping weight: 54 lb. (24.5 kg)



Optical Mark Reader

model 7260A

The 7260A Optical Mark Reader offers mark sense document reading capability for terminals of an HP 3000 computer system.

Features

- Single document used as source document and for data entry
- Reads marks made by ordinary soft lead pencil
- Reads standard punched cards
- ASCII and column image reading formats
- Switchable off-line operation with terminals
- Select hopper available
- Data retransmission capability
- FORTRAN, BASIC, COBOL, SPL callable
- Mixed marked/punched cards

The OMR saves data preparation time and prevents errors by using one functional card for both source document and data entry. The data may be marked with an ordinary soft lead pencil, eliminating the need for special marking pencils or keypunch operations. An added advantage is easy correction of errors—mistakes are simply erased.

Each form may contain pencil marks and any combination of prepunched holes and/or preprinted marks.

Customized forms

Any number of columns from one to 96 may be read. Many variations in layout and color may be used to produce a form. Different colors may be used to visually identify different forms or sections on a form. Clock marks allow vertical columns to be positioned where desired on the form. An optional select hopper is available which allows selected cards to be fed into a separate hopper under program control.

Data entry right at the source

Applications are unlimited because the Optical Mark Reader needs only the most common of data entry devices—pencil and paper. A doctor can record his patient's test results right at bedside on a single form, then use it for both his record and computer input.

Custom forms can be designed for the foreman to do labor reporting, the Q.A. inspector for his tests, the teacher for grading exam papers, or for the finance department's payroll needs. Other applications include inventory control, production control, and warehousing. Computer-printed cards provide excellent turnaround documents that can be pencil-marked before return.

A terminal oriented document reader

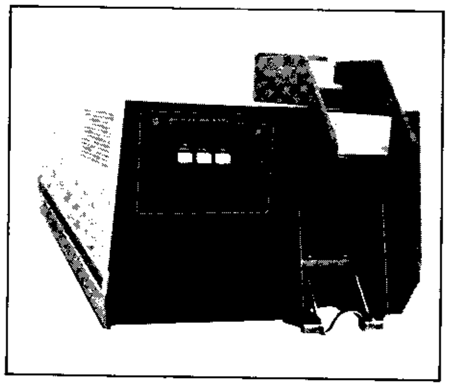
The 7260A Optical Mark Reader is a serial ASCII device. The unit is designed for operation in conjunction with a terminal for remote applications via full duplex modems or hard wired connection to the asynchronous terminal controller of the HP 3000. The HP 3000 has full control over the 7260A. By adding a 7260A to a terminal on the same port of the HP 3000, you are able to supply input from a medium other than the keyboard, saving on line computer time. The terminal can be muted when the reader is transmitting data to the HP 3000.

Multiple data rates

Forms can be read at a rate up to 300 cards per minute. Card feed rate depends on the amount of data to be read on each card and the data transmission rate. Rates of 110, 150, 300, 1200, and 2400 baud can be selected. By proper use of end of record marks, only data marks on cards will be read, trailing spaces will be ignored, thus optimizing data transmission.

ASCII or image reading modes

In the ASCII mode, the card form marks (or punched holes) are interpreted as standard 128 character Hollerith code and the data is transmitted in 7-level ASCII code with even parity. In image mode, the Optical Reader transmits a two character representation of each column of data. This permits all 4096 possible combinations of marks to be read. Switching between ASCII or Image mode can be done under program control.



Operating mode

The HP 3000 supports demand mode operation of the HP 7260A in conjunction with the 2640 series of interactive display terminals.

The maximum number of 7260A Optical Mark Readers that can transmit data to the computer simultaneously is outlined in the table below.

1 unit operating at 2400 baud
1 unit operating at 1200 baud and 2 units at 300 baud
1 unit operating at 1200 baud and 4 units at 150 baud
5 units operating at 300 baud
8 units operating at 150 baud

Because the 7260A is used in conjunction with a terminal (in session mode) it cannot be used as a job accepting device.

Specifications

Forms: Standard tab card size 8.26 x 18.73 cm to 8.26 x 28.26 cm.
Hopper capacity: 450-card input; 450-card output.
UL approval: The unit has U.L. approval, CSA approval pending, and meets IEC specifications.
Controls: Line Switch, Ready, Stop, Terminal Mute, Single Pick, Continuous Pick (not used with the HP 3000), Line/Local, and Full/Half
Indicators: Ready, Pick Fail
Interface: The unit uses Electronic Industries (EIA) RS232C specification.

HEWLETT PACKARD

CalComp Plotter Interface Subsystem

model 30126A

The HP 30126A subsystem interfaces the HP 3000 computer with a CalComp 565 or 702 series plotter. Zip mode format is compatible with the software supplied. The complete interface consists of a single printed circuit board, software driver, basic plotting software and a single cable for interconnecting the plotter and interface.

Features

- **For CalComp series 565 and 702 plotters**
- **Zip mode format supported**
- **Simple subroutines callable from FORTRAN, COBOL, SPL and BASIC**

The user initiated procedures are programmed to translate computed data into distinct plotter commands necessary to direct an on-line plotter. The resulting graphic form can include graphs, three-dimensional drawings, contour maps, charts, etc., and plot annotation (ASCII alphanumeric characters and special graphic symbols). The subsystem is also responsible for file maintenance operations related to the plotter file, and input/output error-handling.

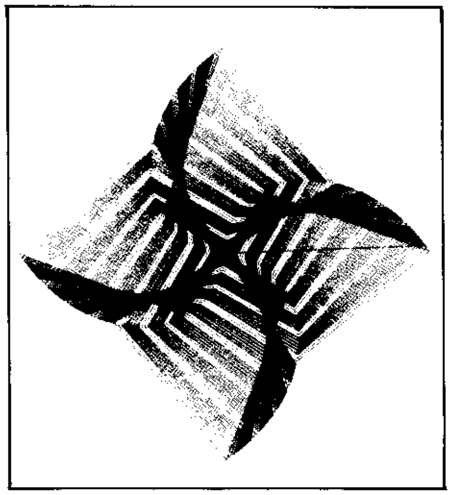
Easy to program

The basic plotting software consists of five FORTRAN callable procedures; their functions are described below:

1. PLOTS: Initializes plotter variables, initializes a user-defined plotter commands buffer, and opens the plotter file.
2. PLOT: Converts X-axis and Y-axis parameters into plotter commands, manages buffering of plotter commands, and closes the plotter file when the plotting sequence is completed.
3. FACTOR: Changes the plot factor (the ratio of the plot physical size to the plot command size).
4. WHERE: Returns the X-axis and Y-axis coordinates of the present pen position (with respect to the current origin) and returns the current plot factor.
5. SYMBOL: Writes plot annotation in the form of ASCII characters and special symbols.

In addition, through the courtesy of CalComp Corporation the following four additional routines are provided.

1. NUMBER: Converts a floating-point number to the appropriate decimal equivalent in order that the number may be plotted in the FORTRAN F-Type format.

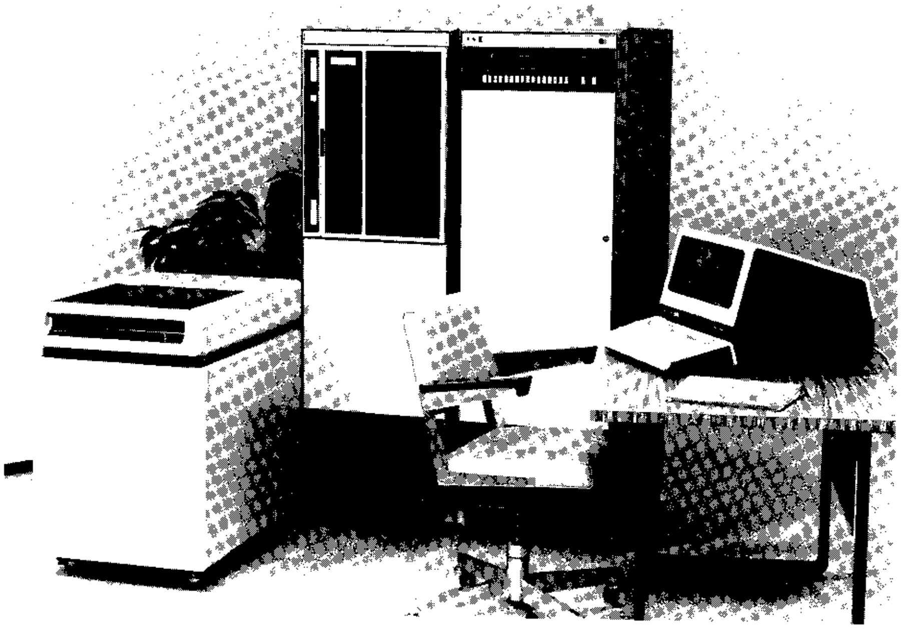


2. SCALE: Examines the data values in an array, also determines a starting value and a scaling factor.
3. AXIS: Indicates the orientation and values of the plotted data points. When both the X and Y axes are needed, AXIS is called separately for each one.
4. LINE: Produces a line plot of the paired data points contained in arrays X and Y. Also computes the page coordinates and scaling factor of these points.

HEWLETT hp PACKARD

The HP 3000 Series II Computer System

The HP 3000 Series II is a general purpose computer system designed to handle a large number of users in commercial, educational, manufacturing, and scientific applications. The Series II features concurrent batch and terminal access, supporting a number of batch peripherals and the capacity for up to 63 interactive terminals. The system can be dedicated to a specific application and still retain its flexibility and performance for adaptation to a variety of tasks.



Features

- **256-kilobyte fault control memory**
- **Multiprogrammed operating system**
- **Concurrent CPU and I/O operation**
- **Integrated terminal access**
- **50 megabytes of disc storage**
- **Software compatible with Series III**
- **Memory expansion to 512kb**

Two standard configurations: Model 6 and Model 8

The HP 3000 Series II is available in two standard configurations, Model 6 and Model 8, allowing you to choose the memory and I/O capacity you require today while providing the growth potential to meet future needs.

The Model 6 standard configuration features 256-kilobytes of fault control memory, with space for 512-kilobytes, and 10 spare I/O slots.

The Model 8 standard configuration features 320-kilobytes of fault control memory, with space for 512-kilobytes, and 23 spare I/O slots.

Both configurations also feature a multiprogrammed operating system (MPE), concurrent CPU and I/O operation, and 50 megabytes of disc storage.

All system software is completely compatible on both the Model 6 and the expanded I/O capacity Model 8. Series II software can also run without modification on a Series III. The performance range growth path of the Series II is extended by two upgrade kits which allow you to expand your system from a Series II to a Series III. One upgrades any Series II to a Series III with 1024kb of memory. It contains replacement CPU, multiplexer, selector channel, memory array and backplane boards, 11 in all. To go beyond 1024kb, however, requires additional power and a memory controller. These components, along with an additional 512kb of memory, are part of the second upgrade kit, which expands memory from 1024kb to 1536kb. To go to the full 2048kb, you need only two additional 256kb memory arrays.

Hardware supplied

The HP 3000 Series II, Model 6 hardware includes:

- Central processing unit (CPU)
- 256-kilobyte fault control memory
- 209 firmware instructions
- System clock
- 16-port asynchronous terminal controller
- Modem support (for type 103A or equivalent)
- 16-channel system I/O multiplexer
- System desk, two bays, card cages, and power supplies
- High-speed selector channel
- 2640B CRT console with 4-kilobyte memory
- 30229A disc controller with 7920A 50Mb disc unit
- 7970E 1600 bpi magnetic tape unit with controller
- 10 spare I/O slots and space for 512k-byte memory
- 60 Hz, 12.6 KVA isolation transformer

The HP 3000 Series II, Model 8 hardware includes:

- Central processing unit (CPU)
- 320-kilobyte fault control memory
- 209 firmware instructions
- System clock
- 16-port asynchronous terminal controller
- Modem support (for type 103A or equivalent)
- 16-channel system I/O multiplexer
- System desk, three cabinet bays, card cages, and power supplies
- High-speed selector channel
- 2640B CRT console with the HP 13234A additional 4-kilobyte memory
- 7920A 50-megabyte disc unit with controller
- 30215A magnetic tape controller with 7970E 1600 bpi magnetic tape unit
- 23 spare slots and space for 512k-byte memory
- 60 Hz, 12.6 KVA isolation transformer

Software supplied

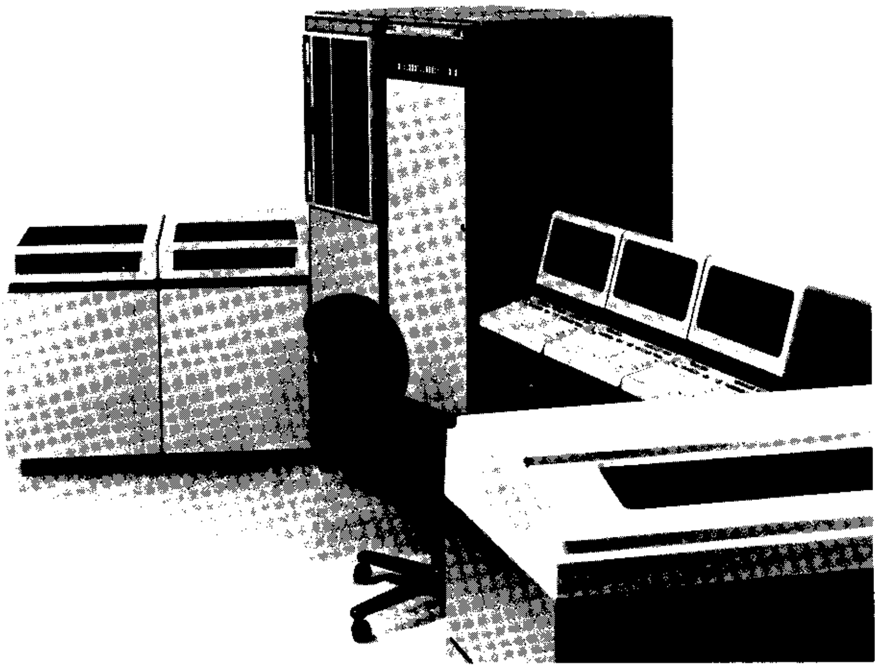
Standard configurations of both Model 6 and the Model 8 supply the same operating system software which includes:

- Multiprogramming Executive III operating system (MPE III)
- HP 3000 Fundamental Operating Software
- System Programming Language (SPL)
- Text editor (EDIT)
- Program debugging aids (DEBUG)
- File copying utilities (FCOPY)
- Sort and merge package (SORT)
- Compiler library

HEWLETT PACKARD

The HP 3000 Series III Computer System

The HP 3000 Series III increases the performance range of the HP 3000 business systems beyond range of the HP 3000 Series II. With capacity for two megabytes of fault control memory, the HP 3000 Series III is designed for those customers who require the performance associated with larger memory configurations for performing transaction processing in commercial, educational, manufacturing, and scientific applications. Similar to the Series II, the HP 3000 Series III features concurrent batch and terminal access, supporting a number of batch peripherals and the capacity for up to 63 interactive terminals. The system can be dedicated to a specific application and still retain its flexibility and performance for adaptation to a variety of tasks.



Features

- 256-kilobyte fault control memory
- Multiprogrammed operating system
- Concurrent CPU and I/O operation
- Integrated terminal access
- 50 megabytes of disc storage
- Software compatible with Series II
- Memory expansion to 2048kb

Standard configuration

The HP 3000 Series III standard configuration features 256-kilobytes of fault control memory, with capacity for 2-megabytes, and 10 spare I/O slots. Supported memory sizes are 256kb, 512kb, 768kb, 1024kb, 1536kb, and 2048kb. Options are available to allow you to expand the I/O capacity to 23 slots and support an additional 32 interactive terminals.

All system software is completely compatible on both the Series II and the larger memory capacity Series III.

Hardware supplied

The HP 3000 Series III hardware includes:

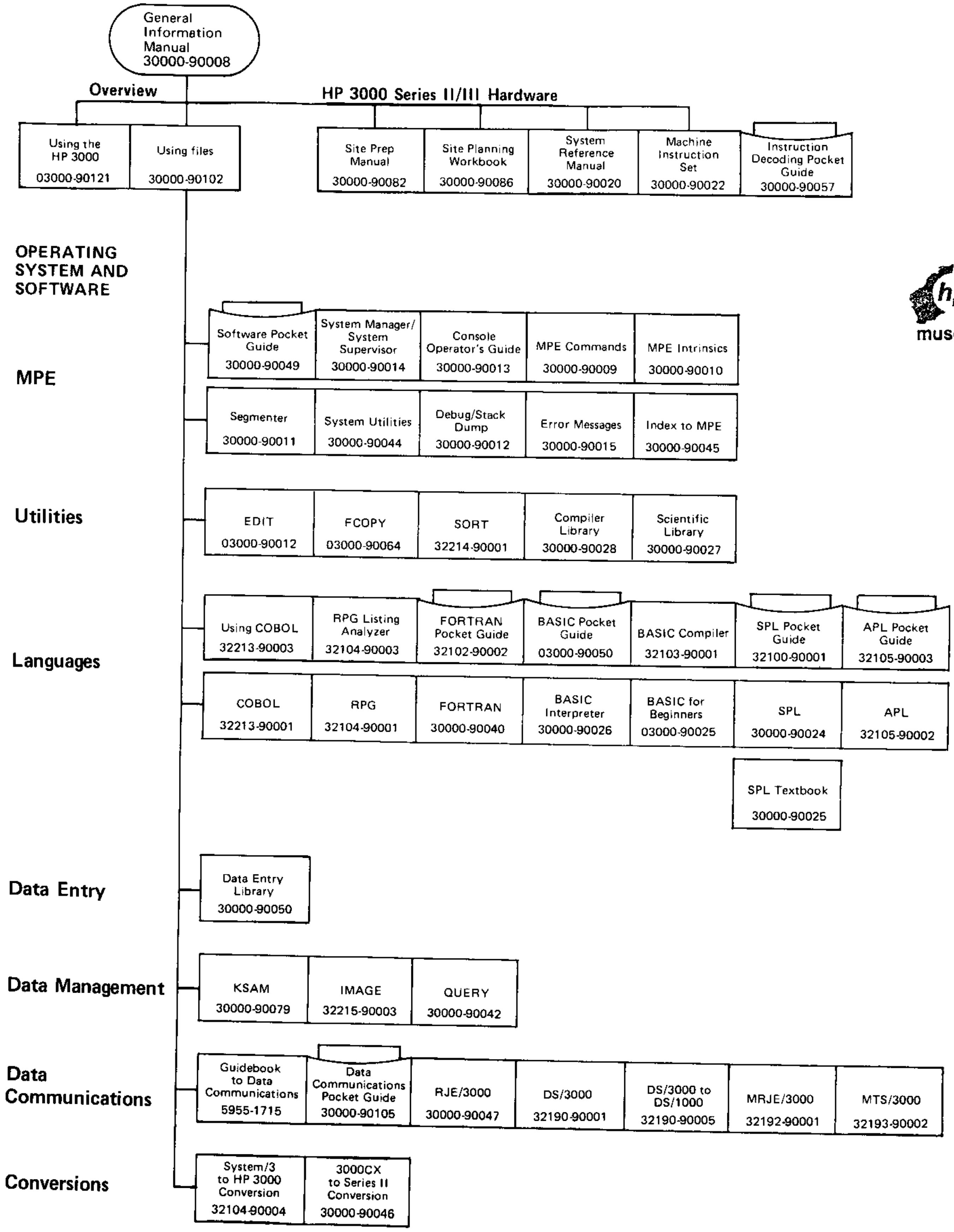
- Central processing unit (CPU)
- 256-kilobyte fault control memory
- 209 firmware instructions
- System clock
- 16-port asynchronous terminal controller
- Modem support (for type 103A or equivalent)
- 16-channel system I/O multiplexer
- System desk, two bays, card cages, and power supplies
- High-speed selector channel
- 2640B CRT console with 4-kilobyte memory
- 30229A disc controller with 7920A 50Mb disc unit
- 7970E 1600 bpi magnetic tape unit with controller
- 10 spare I/O slots and space for 2048k-byte memory
- 60 Hz, 12.6 KVA isolation transformer

Software supplied

Standard configurations of the Series II and the Series III supply the same operating system software which includes:

- Multiprogramming Executive III operating system (MPE III)
- HP 3000 Fundamental Operating Software
- System Programming Language (SPL)
- Text editor (EDIT)
- Program debugging aid (DEBUG)
- File copying utilities (FCOPY)
- Sort and merge package (SORT)
- Compiler library

Appendix A: Documentation


General Information Manual 30000-90008
Overview
HP 3000 Series II/III Hardware
Using the HP 3000 03000-90121
Using files 30000-90102
Site Prep Manual 30000-90082
Site Planning Workbook 30000-90086
System Reference Manual 30000-90020
Machine Instruction Set 30000-90022
Instruction Decoding Pocket Guide 30000-90057
OPERATING SYSTEM AND SOFTWARE
MPE
Software Pocket Guide 30000-90049
System Manager/ System Supervisor 30000-90014
Console Operator's Guide 30000-90013
MPE Commands 30000-90009
MPE Intrinsics 30000-90010
Segmenter 30000-90011
System Utilities 30000-90044
Debug/Stack Dump 30000-90012
Error Messages 30000-90015
Index to MPE 30000-90045
Utilities
EDIT 03000-90012
FCOPY 03000-90064
SORT 32214-90001
Compiler Library 30000-90028
Scientific Library 30000-90027
Languages
Using COBOL 32213-90003
RPG Listing Analyzer 32104-90003
FORTRAN Pocket Guide 32102-90002
BASIC Pocket Guide 03000-90050
BASIC Compiler 32103-90001
SPL Pocket Guide 32100-90001
APL Pocket Guide 32105-90003
COBOL 32213-90001
RPG 32104-90001
FORTRAN 30000-90040
BASIC Interpreter 30000-90026
BASIC for Beginners 03000-90025
SPL 30000-90024
APL 32105-90002
SPL Textbook 30000-90025
Data Entry
Data Entry Library 30000-90050
Data Management
KSAM 30000-90079
IMAGE 32215-90003
QUERY 30000-90042
Data Communications
Guidebook to Data Communications 5955-1715
Data Communications Pocket Guide 30000-90105
RJE/3000 30000-90047
DS/3000 32190-90001
DS/3000 to DS/1000 32190-90005
MRJE/3000 32192-90001
MTS/3000 32193-90002
Conversions
System/3 to HP 3000 Conversion 32104-90004
3000CX to Series II Conversion 30000-90046

Appendix B: MPE III Commands

All MPE III commands are summarized below, grouped alphabetically within capability. The last column denotes when the command can be issued: during a batch job (J), during a session (S), during a break (B), or programmatically (P), through the COMMAND intrinsic.

Standard Capability Commands

Command	Function	When Issued
:( ) COMMAND LOG ON	Begins a session, executes the enclosed MPE command, and ends the session upon completion of the command.	S
:ABORT	Aborts the current program.	B
:ALTSEC	Changes security provisions for a file.	J,S,B,P
:APL	Accesses the APL subsystem.	J,S
:BASIC	Calls BASIC/3000 interpreter.	J,S
:BASICGO	Compiles, prepares, and executes a BASIC/3000 program.	J,S
:BASICOMP	Compiles a BASIC/3000 program.	J,S
:BASICPREP	Compiles and prepares a BASIC/3000 program.	J,S
:BUILD	Creates a new file.	J,S,B,P
:BYE	Terminates a session.	S
:COBOL	Compiles a COBOL/3000 program.	J,S
:COBOLGO	Compiles, prepares, and executes a COBOL/3000 program.	J,S
:COBOLPREP	Compiles and prepares a COBOL/3000 program.	J,S
:COMMENT	Inserts comment into command stream.	J,S,B,P
:CONTINUE	Disregards job-error condition.	J
:DATA	Defines data from outside standard input stream. Cannot be read on $STDINX file. Acceptable for device recognition.	J,S
:DEBUG	Invokes the MPE debug facility.	S,P
:DISMOUNT	Causes a volume set that was mounted by the user to be dismounted.	J,S,B
:DSLINE	Opens or closes communication line with DS/3000.	J,S
:DSTAT	Displays the current status of the disc drives on the system.	J,S,B,P
:EDITOR	Calls the EDITOR.	J,S
:ELSE	Provides an alternate execution sequence for an IF statement.	J,S,B
:ENDIF	Terminates an IF block.	J,S,B
:EOD	Denotes end of data. Cannot be read on $STDINX file.	J,S
:EOF	Simulates hardware end-of-file on input stream from any device.	J,S
:EOJ	Denotes end of batch job. Cannot be read on $STDINX file.	J
:FILE	Defines or redefines a file's characteristics.	J,S,B,P
:FORTGO	Compiles, prepares, and executes a FORTRAN/3000 program.	J,S
:FORTPREP	Compiles and prepares FORTRAN/3000 program.	J,S
:FORTRAN	Compiles a FORTRAN program.	J,S
:FREERIN	Deallocates a global RIN, and returns it to RIN pool.	J,S
:GETRIN	Acquires a global RIN.	J,S,B,P
:HELLO	Initiates a session. Acceptable for device recognition. Requires 1A capability class.	S
:HELP	Accesses the HELP subsystem.	J,S,B
:IF	Used to control the execution sequence of a job.	J,S,B
:JOB	Initiates a batch job. Requires BA capability class.	J,S
:LISTF	Lists descriptions of files.	J,S,B,P
:LISTVS	Produces a formatted listing of volume set definition information.	J,S,B,P
:MOUNT	Requests the console operator to mount a volume set.	J,S,B
:MRJE	Initiates execution of the Multileaving Remote Job Entry (MRJE) facility.	J,S
:PREP	Prepares a compiled program into segmented form.	J,S
:PREPRUN	Prepares and executes a program.	J,S
:PTAPE	Reads a paper tape without X-OFF control.	S,B,P
:PURGE	Deletes a file from the system.	J,S,B,P
:REDO	Allows the user to edit a command entry.	S,B
:RELEASE	Temporarily suspends all security provisions for a file.	J,S,B,P
:REMOTE	Establishes communication between a local computer and a remote computer.	S
:RENAME	Renames a file.	J,S,B,P
:REPORT	Displays total accounting information for a log-on group.	J,S,B,P
:RESET	Resets a formal file designator.	J,S,B,P
:RESETDUMP	Disables the MPE stackdump facility.	J,S,B,P
:RESTORE	Restores a complete fileset, stored off-file.	J,S,B,P
:RESUME	Resumes an interrupted program.	B,S
:RJE	Calls the 2780/3780 Emulator.	J,S
:RPG	Compiles an RPG/3000 program.	J,S

Command	Function	When Issued
:RPGGO	Compiles, prepares, and executes an RPG/3000 program.	J,S
:RPGPREP	Compiles and prepares an RPG/3000 program.	J,S
:RUN	Loads and executes a program.	J,S
:SAVE	Changes a file to permanent status. Requires SF capability for saving files.	J,S,B,P
:SECURE	Restores suspended security provisions for a file.	J,S,B,P
:SEGMENTER	Calls MPE Segmenter.	J,S
:SETCATALOG	Causes the command interpreter to search a catalog of user-defined commands and to establish a directory entry for each command in the catalog.	J,S,B
:SETDUMP	Enables the MPE stackdump facility on abort.	J,S,B,P
:SETJCW	Scans the JCW table for a specified JCW name and updates the value of this JCW.	J,S,B,P
:SETMSG	Disables or enables receipt of user or operator messages at standard list device.	J,S,B,P
:SHOWCATALOG	Lists user-defined command (UDC) files.	J,S,B
:SHOWDEV	Reports status of input/output devices.	J,S,B,P
:SHOWIN	Reports status of input device files.	J,S,B,P
:SHOWJCW	Displays the current state of a job control word.	J,S,B,P
:SHOWJOB	Displays job/session status.	J,S,B,P
:SHOWME	Reports job/session status.	J,S,B,P
:SHOWOUT	Reports status of output device files.	J,S,B,P
:SHOWTIME	Displays current date and time-of-day.	J,S,B,P
:SPEED	Changes input speed or output speed of terminal.	S,B,P
:SPL	Compiles an SPL/3000 program.	J,S
:SPLGO	Compiles, prepares, and executes an SPL/3000 program.	J,S
:SPLPREP	Compiles and prepares an SPL/3000 program.	J,S
:STORE	Stores a set of files off-line.	J,S,B,P
:STREAM	Spools batch jobs or data in session or job mode.	J,S,B,P
:TELL	Transmits a message.	J,S,B,P
:TELLOP	Transmits a message from the user to the computer operator.	J,S,B,P
:VSUSER	Prints a listing of all users of a currently mounted volume set.	J,S,B

Standard Capability Commands (Segmenter Commands)

Command	Function	When Issued
-ADDRL	Adds a procedure to an RL.	J,S
-ADDSL	Adds a segment to an SL.	J,S
-ADJUSTUSLF	Adjusts directory space in a user subprogram library (USL) file.	J,S
-AUXUSL	Designates a source of RBM input for COPY command.	J,S
-BUILDRL	Creates a permanent, formatted RL file.	J,S
-BUILDSL	Creates a permanent, formatted SL file.	J,S
-BUILDUSL	Creates a temporary, formatted USL file.	J,S
-CEASE	Deactivates one or more entrypoints in a USL.	J,S
-COPY	Copies an RBM or segment from one USL to another.	J,S
-EXIT	Exits from Segmenter, returning control to MPE command interpreter.	J,S
-EXPANDUSLF	Changes length of a USL file.	J,S
-HIDE	Sets an RBM internal flag on.	J,S
-INITUSLF	Initializes buffer for a USL file to the empty state.	J,S
-LISTRL	Lists the procedures in an RL.	J,S
-LISTSL	Lists the segments in an SL.	J,S
-LISTUSL	Lists the RBMs in a USL.	J,S
-NEWSEG	Changes the segment name of an RBM.	J,S
-PREPARE	Prepares RBMs from a USL into a program file.	J,S
-PURGERBM	Deletes one or more RBMs from a USL.	J,S
-PURGERL	Deletes an entrypoint or a procedure from an RL.	J,S
-PURGESL	Deletes an entrypoint or a segment from an SL.	J,S
-REVEAL	Sets an RBM internal flag off.	J,S
-RL	Designates an RL for management.	J,S
-SL	Designates an SL for management.	J,S
-USE	Activates one or more RBM entrypoints.	J,S
-USL	Designates a USL for management.	J,S

System Manager Capability Commands

Command	Function	When Issued
:ALTACCT	Changes an account's characteristics.	J,S,B,P
:ALTVSET	Modifies volume set definitions.	J,S,B,P
:LISTACCT	Lists attributes of an account.	J,S,B,P
:NEWACCT	Creates a new account.	J,S,B,P
:NEWVSET	Defines private volume sets and classes.	J,S,B,P
:PURGEACCT	Removes an account and users from the system's or the volume set's directory.	J,S,B,P
:PURGEVSET	Deletes an existing volume set.	J,S,B,P
:REPORT	Displays an account's resource usage.	J,S,B,P
:RESETACCT	Resets resource-use counters for an account and its groups.	J,S,B,P
:RESUMELOG	Resumes logging following suspension caused by an error.	J,S,B,P
:SHOWLOG	Displays log file status.	J,S,B,P
:SHOWQ	Displays scheduling subqueue information.	J,S,B,P
:STORE	Stores disc files onto magnetic tape or serial disc.	J,S,B,P
:SWITCHLOG	Closes the current log file, and creates and opens a new log file.	J,S,B,P
:SYSDUMP	Starts configurator dialog and copies MPE to magnetic tape or serial disc.	J,S

Account Manager Capability Commands

Command	Function	When Issued
:ALTGROUP	Changes a group's attributes.	J,S,B,P
:ALTUSER	Changes a user's attributes.	J,S,B,P
:LISTACCT	Lists attributes of user's log-on account.	J,S,B,P
:LISTGROUP	Lists attributes of a group in user's log-on account.	J,S,B,P
:LISTUSER	Lists attributes of a user in log-on account.	J,S,B,P
:NEWGROUP	Creates a new group in log-on account.	J,S,B,P
:NEWUSER	Creates a new user in log-on account.	J,S,B,P
:PURGEGROUP	Removes a group from the system's or the volume set's directory.	J,S,B,P
:PURGEUSER	Deletes a user from log-on account.	J,S,B,P
:REPORT	Displays resource-usage counts for log-on account and its groups.	J,S;B,P

System Supervisor Capability Commands

Command	Function	When Issued
:ALLOCATE	Permanently allocates a program or procedure in virtual memory.	J,S
:DEALLOCATE	Removes a program or procedure from virtual memory.	J,S,P
:JOBPRI	Sets or changes the priority for batch jobs or sessions.	J,S,B,P
:QUANTUM	Changes a circular subqueue time-quantum.	J,S,B,P
:RESTORE	Returns files to the system.	J,S,B,P

Console Operator Commands

Command	Function
=ABORTIO	Aborts pending I/O requests for a device.
=ABORTJOB	Aborts a job or session.
=ACCEPT	Permits a device to accept job/sessions and/or data.
=ALTFILE	Alters attributes of output spooling files.
=ALTJOB	Alters attributes of waiting job or session.
=BREAKJOB	Suspends an executing job.
=DELETE	Deletes any ready device file.
=DISMOUNT	Logically dismounts a private volume set/class.
=DOWN	Removes a device from normal system use.
=DSLINE	Enables or disables the data communications link under control of the DS/3000 subsystem.
=DSTAT	Displays disc configuration information.
=GIVE	Assigns a DOWNed device to the diagnostics.
=HEADOFF	Stops header/trailer output to a device.
=HEADON	Resumes header/trailer output to a device.
=JOBFENCE	Defines input priorities.
=LIMIT	Limits the number of concurrently running jobs/sessions.
=LOGOFF	Aborts all jobs/sessions and prevents further log-ons by all except HIPRI jobs/sessions.
=LOGON	Enables job/session processing following a LOGOFF command.
=MOUNT	Logically mounts a private volume/class on a non-system domain disc drive.
=MPLINE	Enables or disables the data communications link under control of the MTS/3000 subsystem.
=MRJE	Enables or disables the data communications link under control of the MRJE/3000 subsystem.
=OUTFENCE	Defines priorities for output spooled files.
=RECALL	Displays all pending console REPLY messages.

Command	Function
=REFUSE	Disallows jobs/sessions and/or data on a designated device.
=REPLY	Replies to a pending console request.
=RESUMEJOB	Resumes a suspended job.
=SESSION	Permits the console to be used for a session.
=SHOWDEV	Displays information for a particular device, a class of devices, or for all devices.
=SHOWIN	Displays the status of input devicefiles.
=SHOWJOB	Displays the status of current jobs/sessions.
=SHOWOUT	Displays the status of output device files.
=SHOWQ	Displays scheduling information and the processes currently defined in the system.
=SHOWTIME	Prints the current date and time.
=SHUTDOWN	Closes down the operating system.
=SPOOL	Enables or disables I/O device spooling.
=STREAMS	Enables or disables the users' ability to submit job/session and/or data streams.
=TAKE	De-assigns a device that was GIVEn to the diagnostics.
=TELL	Sends a message to jobs and sessions.
=UP	Allows a DOWNed device to function again.
=VMOUNT	Enables or disables the private volumes facility.
=VSUSER	Lists information about current private volumes users.
=WARN	Sends an urgent message to jobs and sessions.
=WELCOME	Defines the message users receive when they log on the system.

Appendix C: MPE III Intrinsics

All MPE III intrinsics (system procedures) are summarized below, listed alphabetically. Any special capabilities required to call a particular intrinsic are noted.

Intrinsic	Function
ACCEPT	Accepts (and completes) the request received by the preceding GET intrinsic call.
ACTIVATE	Activates a process. (Requires PH capability.)
ADJUSTUSLF	Moves information block on a USL file.
ALTDSEG	Changes the size of an extra data segment. (Requires DS capability.)
ARITRAP	Enables or disables internal interrupt signals from all hardware arithmetic traps.
ASCII	Converts a number from binary to ASCII code.
BINARY	Converts a number from ASCII to binary code.
CALENDAR	Returns the calendar date.
CAUSEBREAK	Requests a session break.
CLOCK	Returns the time of day.
COMMAND	Executes an MPE command programmatically.
CREATE	Creates a process. (Requires PH capability.)
CTRANSLATE	Converts a string of characters from EBCDIC to ASCII or from ASCII to EBCDIC.
DASCII	Converts a value from double-word binary to ASCII code.
DBINARY	Converts a number from ASCII to double-word binary value.
DEBUG	Sets breakpoints and modifies or displays stack or register contents.
DLSIZE	Changes size of DL-DB area.
DMOVIN	Copies block from data segment to stack. (Requires DS capability.)
DMOVOUT	Copies block from stack to data segment. (Requires DS capability.)
EXPANDUSLF	Changes length of a USL file.
FATHER	Requests PIN of father process. (Requires PH capability.)
FCARD	Drives the HP 7260A Optional Mark Reader (OMR).
FCHECK	Requests details about file input/output errors.
FCLOSE	Closes a file.
FCONTROL	Performs various control operations on a file or terminal device.
FERRMSG	Returns message corresponding to FCHECK error number.
FGETINFO	Requests access and status information about a file.
FINDJCW	Searches the job control word table for a specified job control word (JCW).
FLOCK	Dynamically locks a file.
FMTCALENDAR	Converts the calendar date obtained with the CALENDAR intrinsic.

Intrinsic	Function
FMTCLOCK	Converts the time of day obtained with the CLOCK intrinsic.
FMTDATE	Converts calendar date and time of day obtained with the CALENDAR and CLOCK intrinsics.
FOPEN	Opens a file.
FPOINT	Resets the logical record pointer for a sequential disc file.
FREAD	Reads a logical record from a sequential file.
FREADDIR	Reads a logical record from a direct-access disc file.
FREADLABEL	Reads a user file label.
FREADSEEK	Prepares, in advance, for reading from a direct-access file.
FREEDSEG	Releases an extra data segment. (Requires DS capability.)
FREELOCRIN	Frees all local RINs from allocation to a job.
FRELATE	Declares a file pair interactive or duplicative.
FRENAME	Renames a disc file.
FSETMODE	Activates or deactivates file access modes.
FSPACE	Spaces forward or backward on a file.
FUNLOCK	Dynamically unlocks a file.
FUPDATE	Updates a logical record residing in a disc file.
FWRITE	Writes a logical record to a sequential file.
FWRITEDIR	Writes a logical record to a direct-access disc file.
FWRITELABEL	Writes a user file label.
GENMESSAGE	Accesses the message system.
GET	Receives the next request from the remote master program.
GETDSEG	Creates an extra data segment. (Requires DS capability.)
GETJCW	Fetches contents of job control word.
GETLOCRIN	Acquires a local RIN.
GETORIGIN	Determines source of process activation call. (Requires PH capability.)
GETPRIORITY	Reschedules a process. (Requires PH capability.)
GETPRIVMODE	Dynamically enters privileged mode. (Requires PM capability.)
GETPROCID	Requests PIN of a son process. (Requires PH capability.)
GETPROCINFO	Requests status information about a father or son process. (Requires PH capability.)
GETUSERMODE	Dynamically returns to non-privileged mode. (Requires PM capability.)
INITUSLF	Initializes a buffer for a USL file to the empty state.
IODONTWAIT	Initiates completion operations for an I/O request.
IOWAIT	Initiates completion operations for an I/O request.

Intrinsic	Function
KILL	Deletes a process. (Requires PH capability.)
LOADPROC	Dynamically loads a library procedure.
LOCKGLORIN	Locks a global RIN.
LOCKLOCRIN	Locks a local RIN.
MAIL	Tests mailbox status. (Requires PH capability.)
MYCOMMAND	Parses (delineates and defines parameters) for user-supplied command image.
PAUSE	Suspends the calling process for a specified number of seconds.
PCHECK	Returns an integer code specifying the completion status of the most recently executed slave program-to-program intrinsic.
PCLOSE	Terminates the remote slave program's process.
PCONTROL	Transmits a tag field to the remote slave program and receives a tag field back from the slave.
POPEN	Initiates and activates a slave process in a remote HP 3000 and initiates program-to-program communication with the slave program.
PREAD	Sends a read request to the remote slave program asking the slave to send a block of data back to the master.
PRINT	Prints character string on job/session list device.
PRINTFILEINFO	Prints a file information display on the job/session list device.
PRINTTOP	Prints a character string on operator's console.
PRINTOPREPLY	Prints a character string on the operator's console and solicits a reply.
PROCTIME	Returns a process's accumulated central-processor time.
PTAPE	Accepts input from tapes not containing X-OFF control characters.
PUTJCW	Puts the value of a particular job control word (JCW) in the JCW table.
PWRITE	Sends a block of data to the remote slave program.

Intrinsic	Function
QUIT	Aborts a process.
QUITPROG	Aborts a program.
READ	Reads an ASCII string from an input device.
READX	Reads an ASCII string from an input device.
RECEIVEMAIL	Receives mail from another process. (Requires PH capability.)
REJECT	Rejects a request received by the preceding GET intrinsic call and returns an optional tag field back to a remote master program.
RESETCONTROL	Resets a terminal to accept a CONTROL-Y signal.
RESETDUMP	Disables the abort stack analysis facility.
SEARCH	Searches an array for a specified entry or name.
SENDMAIL	Sends mail to another process. (Requires PH capability.)
SETDUMP	Enables the stack analysis facility.
SETJCW	Sets bits in job control word.
STACKDUMP	Dumps selected parts of stack to a file.
SUSPEND	Suspends a process. (Requires PH capability.)
SWITCHDB	Switches DB register pointer. (Requires PM capability.)
TERMINATE	Terminates a process.
TIMER	Returns system-timer bit-count.
UNLOADPROC	Dynamically unloads a library procedure.
UNLOCKGLORIN	Unlocks a global RIN.
UNLOCKLOCRIN	Unlocks a local RIN.
WHO	Returns user attributes.
XARITRAP	Arms the software arithmetic trap.
XCONTRAP	Arms or disarms the CONTROL-Y trap.
XLIBTRAP	Arms or disarms the software library trap.
XSYSTRAP	Arms or disarms the system trap.
ZSIZE	Changes size of Z-DB area.

Appendix D: HP 3000 Series II/III Hardware Features

This appendix summarizes the hardware features of the HP 3000 Series II and Series III computer systems. Both systems are referred to together as the HP 3000 except in the discussion of the memory modules. Any differences are noted by system. To learn more about the HP 3000 architecture, refer to Chapter 2–System Introduction and Architecture.

The design of the HP 3000 hardware provides an efficient and powerful foundation upon which the software is built, as illustrated in Figure D-1. Communication between modules occurs over the central data bus. The central processing unit (CPU) and the input/output processor (IOP), although independent of one another, share a common module address. This is resolved by giving the IOP a higher priority in the case where both processors concurrently request use of the bus. The CPU is controlled by a specially designed microprocessor to allow a great deal of flexibility in the machine instruction set. The HP 3000 also employs high-speed, semiconductor memory modules which use automatic fault detection and correction with no loss in performance.

The basic structure of independent modules organized around a central data bus permits high-speed internal data rates. When not communicating over the bus, each module can run independently at its own speed.

The presence of a separate IOP bus which is totally dedicated to input/output data transfers means that the HP 3000 can always respond immediately to the needs of I/O devices regardless of what transfers are currently in progress between the various system modules. It also means that many I/O operations can be handled concurrently with CPU, main memory, and high-speed selector channel operations.

Data may be transferred directly between main memory and high-speed peripheral devices in block mode by way of high-speed selector channels connected to the central data bus. For lower speed devices, data may be multiplexed on a word-by-word basis by way of the IOP and a multiplexer channel. In both cases the I/O channels execute in parallel with CPU operations. Direct control of devices attached to the IOP bus is also possible through the use of the CPU's direct I/O instructions.

The configuration of hardware modules and peripheral devices is easily changed to accommodate system expansion.

Central processing unit (CPU)
The significant features of the HP 3000 central processing unit (CPU) are listed in Table D-1.

Table D-1. HP 3000 CPU Features

Architecture
- **Hardware-implemented stack**
- **Separation of code and data**
- **Non-modifiable, re-entrant code**
- **Variable-length code segmentation**
- **Virtual memory for code**
- **Dynamic relocatability of programs**

Implementation
- **Microprogrammed CPU**
- **175 nanosecond microinstruction time**
- **Automatic restart after power failure**
- **Central data bus**
- **Bus parity checking**
- **Concurrent CPU and I/O operations**

Instructions
- **209 powerful instructions**
- **All instructions except stack operations are 16 bits in length (stack operations may be packed two per word)**
- **16- and 32-bit integer arithmetic**
- **32- and 64-bit floating point arithmetic**
- **28-digit packed decimal arithmetic**
- **Special instructions that optimize the efficiency of the operating system.**

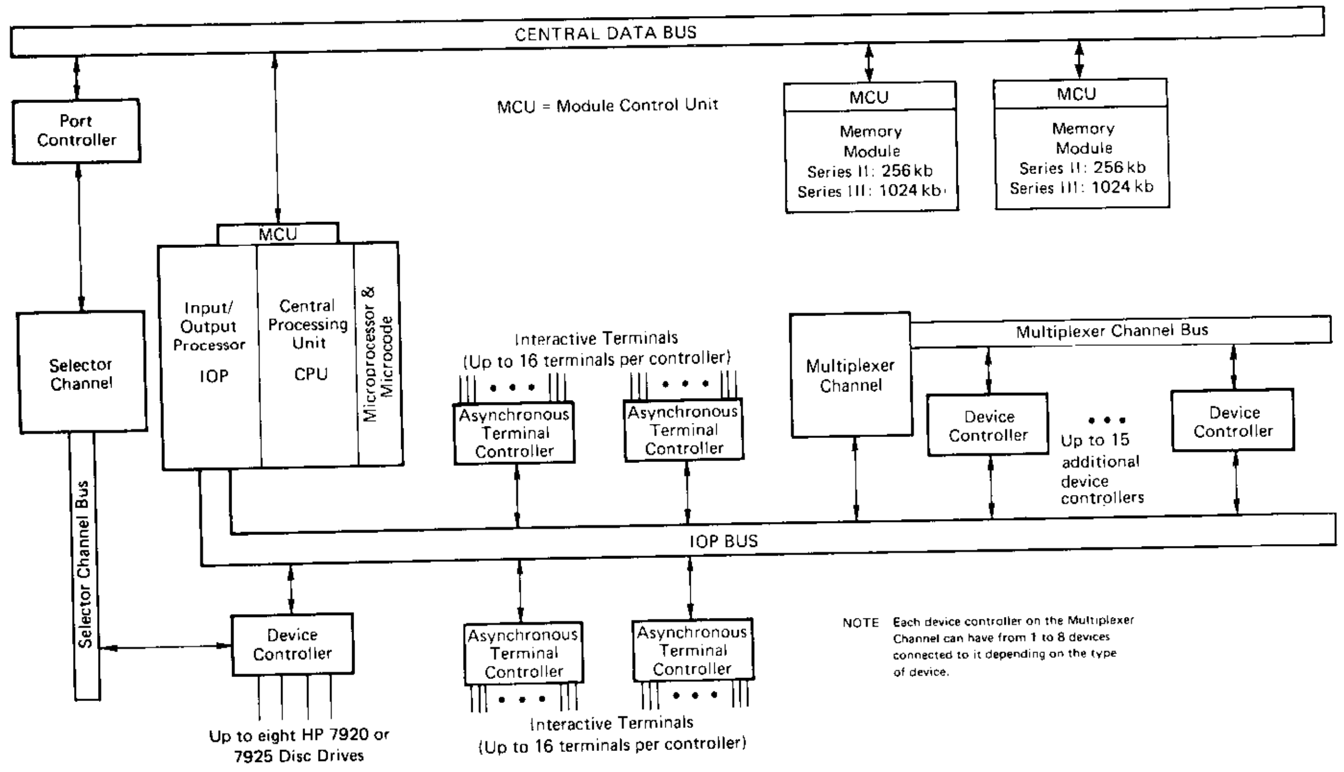

CENTRAL DATA BUS
MCU = Module Control Unit
Port Controller
MCU
Memory Module
Series II: 256 kb
Series III: 1024 kb
MCU
Memory Module
Series II: 256 kb
Series III: 1024 kb
MCU
Input/ Output Processor
IOP
Central Processing Unit
CPU
Microprocessor & Microcode
Selector Channel
Interactive Terminals
(Up to 16 terminals per controller)
Asynchronous Terminal Controller
Asynchronous Terminal Controller
Multiplexer Channel
Multiplexer Channel Bus
Device Controller
Up to 15 additional device controllers
Device Controller
IOP BUS
Selector Channel Bus
Device Controller
Up to eight HP 7920 or 7925 Disc Drives
Asynchronous Terminal Controller
Asynchronous Terminal Controller
Interactive Terminals
(Up to 16 terminals per controller)
NOTE Each device controller on the Multiplexer Channel can have from 1 to 8 devices connected to it depending on the type of device.

Figure D-1. HP 3000 Hardware Organization (Maximum Configuration)

The three major components of the CPU are the instruction decoder, firmware storage and control, and hardware processor.

The instruction decoder unit converts an instruction in the current instruction register (CIR) into a starting address for the microcode contained in a read-only memory (ROM), and determines various initial conditions required for executing the instruction. As the current instruction is being executed, the next instruction is fetched and placed into the next instruction register (NIR). Upon completion of the current instruction, the contents of NIR are loaded into CIR and the cycle is repeated. This "pipelining" of the current instruction execution with the next instruction-fetch improves throughput by overlapping operations. The HP 3000 instruction set is presented in Appendix E. All instructions are 16 bits in length except stack operations, which are 8-bit instructions. These include a variety of memory reference, branch, arithmetic and data manipulation instructions that operate on integer, real, logical, packed decimal, character and string data. Floating point arithmetic can be performed in single precision (32 bits) or double precision (64 bits), integer arithmetic in 16-bit and 32-bit lengths, and packed decimal instructions extend to 28 digits of precision. In addition, there are a number of instructions designed to aid in creating the multiprogramming environment of the system. These include procedure call and exit instructions and others which implement various operating system functions previously done in software.

Firmware storage and control consists of microcode, stored in read-only memory (ROM), and associated control logic. Microcode routines control the operation of the instruction decoder and the hardware processor, in order to create the HP 3000 operating environment–including the 209 instructions available to the programmer. The control storage has a cycle time of 175 nanoseconds and an average microinstruction execution time of 175 nanoseconds.

The hardware processor consists of an arithmetic-logic unit, shifting network, 38 specific purpose registers–20 of which are accessible to user programs, and related data manipulating and testing logic. Since the HP 3000 architecture (see Chapter 2) is structured on code segments and data segments, most of the CPU registers are used for defining the segment limits and operating elements within the segments. As shown in Figure D-2, three of the CPU registers point to locations in a code segment defined as the current code segment. Six of the registers point to locations in a data segment defined as the current data segment. Table D-2 lists all 38 registers and their associated functions.

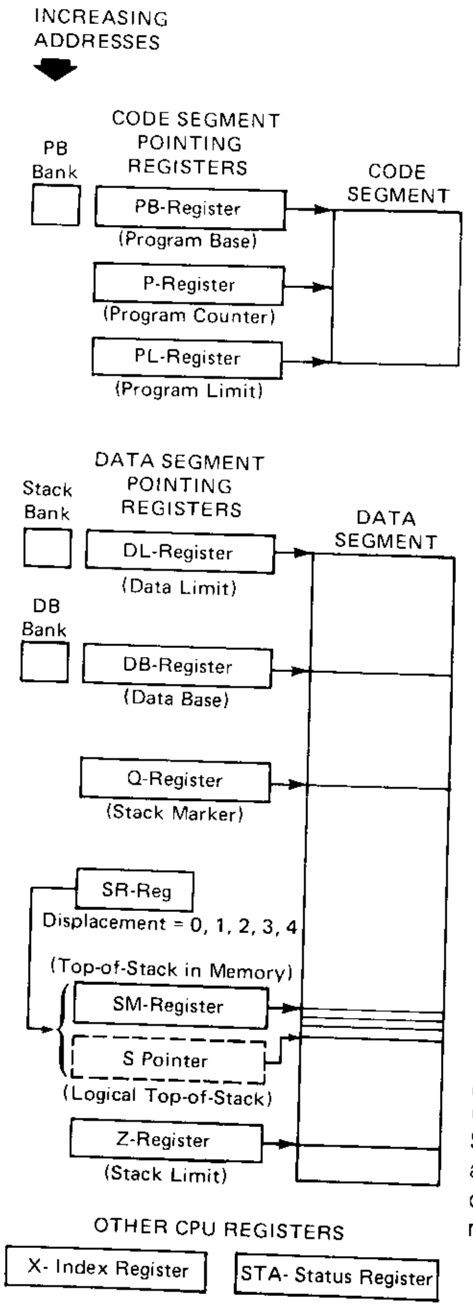


Figure D-2. HP 3000 CPU Registers

The four top of stack registers are of special interest. In order to improve execution speed, up to four elements from the top of a data stack may be contained in these registers. This allows many functions to be treated as register-to-register operations rather than the slower speed memory-to-register or register-to-memory type operations. These registers are manipulated by the CPU, and their use is fully transparent.

Table D-2. HP 3000 Hardware Registers

Register	Function
Registers Accessible to User Programmers	
PB, P, PL, PB-Bank	Code Segment Pointers
DL, DB, Q, SM, SR, Z, DB-Bank, S-Bank	Stack Pointers
RA, RB, RC, RD	Top of Stack Registers
X	Index Register
STA	Status Register
SWCH	Switch Register
PCLK	Program Clock Register
Registers Dedicated For System Use	
CIR	Current Instruction Register
NIR	Next Instruction Register
SP0, SP1, SP2, SP3, CTR, ABS-Bank, CPX1, CPX2, MOD	Scratch Pad, Flag, and Interrupt Registers
IOA, IOD	I/O Registers
ACOR, DCOR, OPND	Memory Address and Data Registers
RAR, SAVE	Firmware Address Registers

Main memory

The significant features of the HP 3000 main memory are listed in Table D-3.

Table D-3. HP 3000 Main Memory Features

- High-speed, semiconductor, random access memory
- Automatic fault detection and fault correction
- Memory sizes ranging from 256k bytes to 512k bytes in Series II, 256k bytes to 2 megabytes in Series III
- Write: 700 nsec minimum cycle time
- Read: 350 nsec access, 700 nsec cycle time
- 45-90 minute rechargeable battery packs to maintain memory data during power failure

The HP 3000 uses high-speed, semiconductor, random access memory (4k MOS RAM in Series II, 16k MOS RAM in Series III) which provides single-bit fault correction. Some double bit faults are detected on the Series II, and all double bit faults are detected on the Series III. The parity system is retained with this feature, thus maintaining integrity over the various busses.

Due to the modular design of the HP 3000, any system can be easily expanded from one memory size to another. When there is more than one megabyte of memory in the Series III (256kb in the Series II), it is divided into two modules: one megabyte in the first module (256kb on the Series II) and the remainder in the second. These modules can operate concurrently which improves execution time.

The word length transmitted over the bus is 17 bits–16 bits of data (one word or two bytes) and one parity bit. In the memory modules on the Series III the word length is expanded to 22 bits–16 bits of data and six bits for automatic fault detection and correction. The Series II uses a 21 bit word length in the memory modules–16 bits of data and five bits for automatic fault detection and correction.

All detected memory faults are automatically logged to special (non-user) storage. MPE periodically reads this storage and writes the information to the disc file. This file is accessed by an HP Customer Engineer, from a terminal on the system, while performing preventative maintenance. If memory chips have a history of failures, they are replaced during maintenance.

Operating power for the memory modules is supplied by rechargeable battery packs in the semiconductor memory power supplies. When the power supply input voltage is removed, battery power is available for up to 90 minutes (depending on memory size and battery condition) to maintain memory data.

The memory modules interface with other system modules by way of the central data bus. The other system modules may request transfers of data to or from the memory modules on that bus. Operation of the memory modules with other system modules on the central data bus is controlled by module control units, one for each module.

Input/output

All access to input/output devices is by way of the device-independent MPE file system. All location of data, buffering, data transfers, and deblocking are handled automatically by MPE. When you ask to read a named file, you are only implicitly specifying the actual disc address of the file; the file system determines the explicit address and performs the read. At another level, when you ask the file system for a certain type of device by specifying a device class name (e.g., magnetic tape, line printer, etc.), the file system takes care of allocating an actual device. If you must have actual contact with specific devices, you may address them directly. Below this single, flexible interface is a powerful and carefully balanced hardware/software input/output system.

All devices can be operated concurrently (within system bandwidth). Peripherals that fail are taken off-line from the operating system by operator command.

There are two distinct means of implementing I/O: direct I/O (DIO) and programmed I/O (SIO).

Direct I/O (DIO): Direct I/O allows for single word transfers of data, status, or control information between a device connected to the IOP bus and the top of your data stack.

Programmed I/O (SIO): Programmed I/O can be used with devices on either the selector channel (high-speed devices) or the multiplexer channel (medium- to low-speed devices). With programmed I/O, the CPU simply issues a "Start I/O" instruction to the device controller. The device controller, in turn, initiates the SIO program for the particular device which then runs under the control of the selector channel or under the control of the multiplexer channel and the IOP. The SIO program uses a unique set of commands (optimized for I/O operations) to transfer information between main memory and the external device. Both the selector channel and the multiplexer channel (via the IOP) have direct access to main memory. The SIO program and CPU processing run concurrently until the appropriate I/O command terminates the device transfer. This I/O command can also cause an interrupt signal to be sent to the CPU, thus informing the CPU that the I/O task is complete.

Input/output processor (IOP): The input/output processor controls the IOP bus and interrupt lines, providing the communications and data path between the CPU and I/O devices for direct I/O operations and interrupt processing. It also offers a data path between memory and I/O devices for programmed I/O operations.

I/O operations are divided into three categories, direct I/O, programmed I/O, and interrupt processing. Direct I/O operations take place as a result of I/O instructions executed by the CPU; they result in transfer of a word of information between the CPU and an I/O device through the IOP or cause a control function to take place in the I/O system. Devices connected to a multiplexer (see Peripheral I/O Hardware) may use programmed I/O. Once started, programmed I/O operations can, (through the execution of I/O programs stored in memory) transfer block(s) of data between I/O devices and memory, and perform other device control functions without further CPU intervention or attention. The IOP also accepts interrupt requests from the device controllers, interrogates them by means of a poll line to find the highest priority request, and sends an interrupt signal together with the number of the interrupting device to the CPU.

Peripheral I/O hardware: HP 3000 peripheral I/O hardware consists of the selector channel, IOP bus, interrupt lines, multiplexer channel, device controllers, and the peripheral units. Asynchronous terminal controllers interface log-on and data-entry terminals. Table D-4 lists the features which this peripheral I/O hardware offers.

Table D-4.
HP 3000 Peripheral I/O Hardware Features

- 16 ports per terminal controller
- Up to 4 terminal controllers per system
- Support for 16 device controllers on the multiplexer channel
- Options for type 202S and 202T modems
- Type 103A and 113B modem support

Asynchronous terminal controller: The asynchronous terminal controller (ATC) is designed to interface terminals to the HP 3000 via the IOP bus. Up to 16 terminals (including the system console) can be interfaced via one ATC, and up to 4 ATCs may be connected to the system. Terminals can be hardwired or connected through type 103A2, 103A3, 103J, 113B, 202S/T, and Vadic VA 3400 modems. Terminals interfaced through the ATC can be configured to the Multiprogramming Executive as data entry terminals under user program control, or as log-on terminals accessing all the capabilities of the system. Since the system console occupies one terminal port, fifteen additional terminal ports are available before expansion to another controller is necessary.

Selector channel: The selector channel interfaces high speed peripheral devices to the HP 3000. Connecting to the central data bus through a port controller, it can accommodate up to eight disc drives connected to one device controller. The selector channel data transfers bypass the IOP completely to provide data transfer rates of up to 2.86 million bytes per second for a single device. Unlike the multiplexer channel (see next section) which switches between device controllers on demand, based on hardware priority, the selector channel maintains the connection for one device controller until it has completed the I/O program. Thus only one I/O program is current at a time for one selector channel. All selector channel data transfers are performed in block mode and the data is applied directly to main memory via the central data bus.

Multiplexer channel: The multiplexer channel acts as a switch to enable one of the 1 through 16 device controllers connected to it to transfer one word of data to or from memory via the IOP, then to allow another controller—based on priority—to perform its transfer. Operating in conjunction with the IOP, the multiplexer allows the device controllers connected to it to run concurrently, interleaving their transfers on a word-by-word basis. By multiplexing, asynchronous cumulative data rates of up to 1,038,000 bytes per second (inbound) and 952,000 bytes per second (outbound) are possible. Data from the multiplexer channel is supplied directly to the IOP for transfer to main memory via the central data bus.

A solid state memory in the multiplexer is divided into 16 sections, one for each device controller. Typically, this memory contains the current I/O program word and related information for each device. When a device is selected for service, the multiplexer executes the indicated operation (or portion thereof) in conjunction with the device controller.

Device controller: The device controller is the hardware linkage between a peripheral device and the computer system. Its primary function is to translate programmed I/O commands from a multiplexer channel or selector channel (or direct I/O commands from the I/O processor) to the unique signals required to control a particular device. When an I/O program is in execution, the device controller responds to and requests service from the multiplexer channel or selector channel. The device controller also generates interrupts when required by some device condition or by direct or programmed command.

Device reference table (DRT): Device controllers are identified by a device number which is used to access the device reference table (DRT). The DRT is known to both hardware and software containing, among other things, a pointer to the start of the SIO program for each device controller. Since there can be a maximum of 125 entries in this table, the HP 3000 logic design allows for up to 125 device controllers in its I/O system (the actual limitation is the 7-bit I/O address bus). Certain device controllers may control several devices. In such cases, each device attached to the controller is addressed separately using a unit number assigned when the device is installed.

Data service and interrupt priorities: In addition to a device number, there are two other characteristic numbers associated with each device. These are the data service priority and interrupt priority. Each of these values is completely independent of the other, and neither is related to the physical location of devices or controllers. This mutual independence of characteristics provides the following advantages:

1. Device numbers can be assigned consecutively, starting with number 3 and proceeding up to the last assigned device in the system. When a new device controller is added, it is merely assigned the next available number (or any vacant number).
2. A new device added to the system may have its controller connected anywhere in the interrupt or data service priority chains, independent of physical location within the cabinet.
3. Since data service priority and interrupt priority are independent of each other, a device which requires a high data transfer rate but interrupts infrequently (such as a disc) may be assigned a high data service priority and a low interrupt priority.

Interrupt system: The interrupt system provides for up to 125 external interrupt levels. When interrupts occur, the microprogrammed interrupt handler identifies each interrupt and grants control to the highest priority interrupt. Current operational status is saved by the microprogram, which then sets up the interrupt processing environment and transfers control to the interrupt routine.

Interrupt routines operate on a common stack (interrupt control stack) which is known to both hardware and software. This feature permits nesting of interrupt routines in the case of multiple interrupts, thus allowing higher priority devices to interrupt lower priority devices.

The interrupt system also provides for 22 internal interrupts (for user errors, system violations, hardware faults, and power fail/restart) plus fourteen traps for arithmetic errors and illegal use of instructions.

Peripherals: Peripheral devices receive output data for storage or display, or supply input data to the computer. Usually, one device controller controls one peripheral device; however, some device controllers are capable of controlling several devices.

Mass storage devices consist of moving-head disc drives. These units provide storage capabilities of up to 120 million bytes and data transfers of nearly one megabyte per second. On-line storage can be greatly expanded by connecting additional disc drives to the system.

Low cost magnetic tape units are available in 9-channel models with recording densities of 800 or 1600 bytes per inch. A card reader which operates at 600 cards per minute is also optional.

An HP 2640B interactive display terminal is supplied as standard equipment for the system console. Also available are the HP 2641A APL terminal for use with the APL\3000 language, the HP 2645A mini data station for general purpose use and optional off-line mini tape cartridge storage, and the HP 2648A graphics terminal. Each of these terminals can have directly connected to it either the HP 9871A hardcopy printer, the HP 9866A thermal printer, or the HP 2631 dot-matrix printer. The printers provide high quality, permanent records of the alphanumeric data stored in the terminal's memory.

The HP 2635A printing terminal is available for general purpose use where keyboard entry and a high quality printed record are desired in a single unit.

Line printers are available with operating speeds of 300, 600, or 1250 lines per minute. The various line printers provide 132-column print lines, using 64- or 96-character ASCII sets.

High-speed punched tape equipment reads at 500 characters per second. Punched tape output is available as a separate unit operating at 75 characters per second. Paper, plastic, or mylar tape may be used with all units.

An interface is offered to connect CalComp Series 565 and 702 plotters to the system.

A card reader/punch provides fully buffered, on-line, 80-column card reading, punching, and printing capabilities in a single device. Off-line data recorder (keypunch) capability is optionally available. The device reading rate is 120 cards per minute; the punching/printing rate is 45 to 75 cards per minute, depending upon the number of card columns involved. The punch employs the ANSI Hollerith card code and the printer employs a modified 64-character ASCII set as standard.

Also provided are the asynchronous terminal controller and synchronous interfaces for data communications. There may be up to four asynchronous terminal controllers connected to the system, each controlling up to 16 asynchronous terminals. The synchronous interfaces are used for RJE/3000, DS/3000, MRJE/3000, and MTS/3000 operation.

Hewlett-Packard furnishes available peripherals as complete I/O subsystems (including the device, interface, cables, etc.) to facilitate system expansion.

Automatic restart after power failure
An integral part of the HP 3000 power supply is a power fail/automatic restart capability. When the system AC line voltage falls below 170 volts, the system initiates a power fail warning (PFW). During PFW the system (hardware and MPE) writes all register contents to a reserved section of main memory, activities in the system are successfully completed, and then the power down signal is generated and the system is shut down. The battery back-up power supply refreshes main memory and ensures its validity for 45 to 90 minutes, depending on memory size and battery condition.

The system is automatically restarted when all power supply voltages reach 90% of their normal values. There is a minimum restart delay of 0.6 seconds following the power on signal before the system is restarted, during which another power failure is possible. After the delay all register values are automatically restored and processing resumes.

Diagnostics
Several levels of diagnostic software help identify and diagnose hardware problems in the HP 3000 computer system. The levels of diagnostics are:

- Verification programs for peripherals run under MPE
- Stand-alone diagnostics to verify all system modules
- Panel microdiagnostics in PROM for CPU, memory, and I/O

Appendix E: HP 3000 Series II/III Machine Instructions

STACK OP INSTRUCTIONS

ADAX	Add A to X	FIXT	Fix and truncate
ADBX	Add B to X	FLT	Float an integer
ADD	Add A to B	FMPY	Floating point multiply
ADXA	Add X to A	FNEG	Floating point negate
ADXB	Add X to B	FSUB	Floating point subtract D,C − B,A
AND	Logical AND of A and B	INCA	Increment A
BTST	Test byte on TOS and set CC	INCB	Increment B
CAB	Rotate A-B-C	INCX	Increment X
CMP	Integer compare B, A and set CC	LADD	Logical add A + B
DADD	Double integer add D,C + B,A	LCMP	Logical compare B, A and set CC
DCMP	Double integer compare and set CC	LDIV	Logical divide C,B ÷ A
DDEL	Double delete TOS	LDXA	Load X into A
DDIV	Double integer divide	LDXB	Load X into B
DDUP	Double duplicate TOS	LMPY	Logical multiply B × A
DECA	Decrement A	LSUB	Logical subtract B − A
DECB	Decrement B	MPY	Multiply integers, integer product
DECX	Decrement X	MPYL	Multiply integers, long integer product
DEL	Delete TOS	NEG	Integer negate
DELB	Delete B	NOP	No operation
DFLT	Float a double integer	NOT	Logical complement TOS
DIV	Integer divide B by A	OR	Logical OR of A, B
DIVL	Divide long integer C,B ÷ A	STAX	Store A into X
DMUL	Double integer multiply	STBX	Store B into X
DNEG	Double integer negate	SUB	Integer subtract B − A
DSUB	Double integer subtract D,C − B,A	TEST	Test TOS and set CC
DTST	Test double word on TOS and set CC	XAX	Exchange A and X
DUP	Duplicate TOS	XBX	Exchange B and X
DXCH	Double exchange	XCH	Exchange A and B
DZRO	Push double zero onto stack	XOR	Logical exclusive OR of A, B
FADD	Floating point add D,C + B,A	ZERO	Push integer zero onto stack
FCMP	Floating point compare and set CC	ZROB	Zero B
FDIV	Floating point divide D,C ÷ B,A	ZROX	Zero X
FIXR	Fix and round		

SHIFT INSTRUCTIONS

ASL	Arithmetic shift left	DLSR	Double logical shift right
ASR	Arithmetic shift right	LSL	Logical shift left
CSL	Circular shift left	LSR	Logical shift right
CSR	Circular shift right	QASL	Quadruple arithmetic shift left
DASL	Double arithmetic shift left	QASR	Quadruple arithmetic shift right
DASR	Double arithmetic shift right	TASL	Triple arithmetic shift left
DCSL	Double circular shift left	TASR	Triple arithmetic shift right
DCSR	Double circular shift right	TNSL	Triple normalizing shift left
DLSL	Double logical shift left		

LEGEND

TOS	Top of stack	A	Top of stack	D	Location below C
CC	Condition Code	B	Location below A	DB	Data Base
X	Index Register	C	Location below B	DL	Data Limit

FIELD AND BIT INSTRUCTIONS

DPF	Deposit field, A bits to B	TCBC	Test and complement bit, set CC
EXF	Extract specified field, right-justifiy	TRBC	Test and reset bit, set CC
SCAN	Scan bits	TSBC	Test and set bit, set CC
TBC	Test specified bit and set CC		

BRANCH INSTRUCTIONS

BCC	Branch on specified CC	BRO	Branch on TOS odd (bit 15 = 1)
BCY	Branch on carry	CPRB	Compare range and branch
BNCY	Branch on no carry	DABZ	Decrement A, branch if zero
BNOV	Branch on no overflow	DXBZ	Decrement X, branch if zero
BOV	Branch on overflow	IABZ	Increment A, branch if zero
BR	Branch unconditionally	IXBZ	Increment X, branch if zero
BRE	Branch on TOS even (bit 15 = 0)		

MOVE INSTRUCTIONS

CMPB	Compare bytes in two memory blocks	MVB	Move bytes in memory, addresses +/−
MABS	Move using absolute addresses	MVBL	Move words from DB+ to DL+ area
MDS	Move using data segments	MVBW	Move bytes while of specified type
MFDS	Move from data segment	MVLB	Move words from DL+ to DB+ area
MOVE	Move words in memory, addresses +/−	SCU	Scan bytes until test or terminal byte
MTDS	Move to data segment	SCW	Scan bytes while equal to test byte

PRIVILEGED MEMORY REFERENCE INSTRUCTIONS

LDEA	Load double word from extended address	PSTA	Privileged store into absolute address
LSEA	Load single word from extended address	SDEA	Store double word into extended address
LST	Load from system table	SSEA	Store single word into extended address
PLDA	Privileged load from absolute address	SST	Store into system table

IMMEDIATE INSTRUCTIONS

ADDI	Add immediate to integer in A	LDXI	Load X immediate
ADXI	Add immediate to X	LDXN	Load X negative immediate
ANDI	Logical AND immediate with A	MPYI	Multiply immediate with A
CMPI	Compare A with immediate, set CC	ORI	Logical OR immediate with A
CMPN	Compare A with negative immediate	SBXI	Subtract immediate from X
DIVI	Divide immediate into A	SUBI	Subtract immediate from A
LDI	Load immediate to TOS	XORI	Logical exclusive OR immediate
LDNI	Load negative immediate to TOS		

REGISTER CONTROL INSTRUCTIONS

ADDS	Add operand to stack pointer	SETR	Set specified registers from stack
PSHR	Push specified registers onto stack	SUBS	Subtract operand from stack pointer
RCLK	Read clock	XCHD	Exchange DB and TOS
SCLK	Store clock		

PROGRAM CONTROL AND SPECIAL INSTRUCTIONS

DISP	Dispatch	PCN	Push CPU number
EXIT	Exit from procedure	PSDB	Pseudo interrupt disable
HALT	Halt	PSEB	Pseudo interrupt enable
IXIT	Interrupt exit	RSW	Read switch register
LLBL	Load label	SCAL	Subroutine call
LLSH	Linked list search	SXIT	Exit from subroutine
LOCK	Lock resource	UNLK	Unlock resource
PAUS	Pause, interruptable	XEQ	Execute stack word
PCAL	Procedure call		

I/O INSTRUCTIONS

CIO	Control I/O, direct	SIN	Set interrupt
CMD	Send command to module, direct	SIO	Start I/O, block transfer
RIO	Read I/O, direct	SMSK	Set device mask
RMSK	Read device mask	TIO	Test I/O, direct
SED	Set enable/disable external interrupts	WIO	Write I/O, direct

LOOP CONTROL INSTRUCTIONS

MTBA	Modify variable, test against limit, branch	TBA	Test variable against limit, branch
MTBX	Modify X, test against limit, branch	TBX	Test X against limit, branch

MEMORY ADDRESS INSTRUCTIONS

ADDM	Add memory to TOS	LDX	Load X
CMPM	Compare TOS with memory	LOAD	Load word onto stack
DECM	Decrement memory	LRA	Load relative address onto stack
INCM	Increment memory	MPYM	Multiply TOS by memory
LDB	Load byte onto stack	STB	Store byte on TOS into memory
LDD	Load double word onto stack	STD	Store double on TOS into memory
LDPN	Load double from program, negative	STOR	Store TOS into memory
LDPP	Load double from program, positive	SUBM	Subtract memory from TOS

EXTENDED INSTRUCTION SET

Extended-Precision Floating Point		Decimal Arithmetic	
EADD	Add	ADDD	Decimal add
ECMP	Compare	CMPD	Decimal compare
EDIV	Divide	CVAD	ASCII to decimal conversion
EMPY	Multiply	CVBD	Binary to decimal conversion
ENEG	Negate	CVDA	Decimal to ASCII conversion
ESUB	Subtract	CVDB	Decimal to binary conversion
		DMPY	Double logical multiply
		MPYD	Decimal multiply
		NSLD	Decimal normalizing left shift
		SLD	Decimal left shift
		SRD	Decimal right shift
		SUBD	Decimal subtract

Appendix F: HP 3000 Guide to Synchronous Modems

Selection of a modem is a critical factor in planning a geographically dispersed network. A wide variety of modems is available, so this guide has been prepared to aid in choosing the proper unit. Hewlett-Packard recommends that the modems listed below be used. If other modems are selected, they must be functionally and electrically identical (at the system interface) with the recommended modems.

Modem selection criteria

In planning an HP 3000 network, the selection, installation, and proper functioning of common carrier or third party modems is your responsibility. Hewlett-Packard accepts responsibility for maintaining hardware and software compatibility only with the modems recommended herein. To determine the correct modem for your system, the following parameters must be considered.

Operating mode: Switched network (dial-up) or leased (private) line.

Type of circuit: Two-wire (half duplex) or four wire (full duplex).

Transmission speed: 2000 bps, 2400 bps, 4800 bps, or 9600 bps.

Line conditioning: Unconditioned or conditioned.

Modem supplier: Common carrier (PTT) or third party suppliers.

A communications network's performance, in terms of throughput and responsiveness, is heavily influenced by the decisions made concerning these parameters. For low volume networks, low speed communications may be appropriate. Most users normally will select the highest performance service that meets their budgetary requirements. The five parameters are closely related, and, because of technical considerations or common carrier policy, some decisions will dictate others. In all cases it is wise to discuss these parameters in detail with your common carrier data services representative.

Since modem selection is also based on software related considerations, you should also consult your Hewlett-Packard representative regarding the HP communications software and applications which you will be using.

Operating mode: Switched network (dial-up) operation normally is advisable when communications occur only periodically (once or twice per day) and the volume of data to be transmitted is low to moderate. Generally, only two-wire circuits are available for switched networks, with operation limited to 4800 bps or below.

Leased line service is appropriate for higher volumes of data and continuous on-line service.

Type of circuit: Two-wire communications circuits send/receive in one direction at a time, and to reverse the direction of transmission a "line turnaround" must be performed. Since frequent, time-consuming turnarounds are necessary, two-wire circuits should be selected only for low volume/response applications.

Four-wire circuits can send and receive simultaneously, which eliminates turnaround delays. Most leased line service is four-wire. To maintain responsiveness and the best transmission throughput, four-wire circuits are suggested.

Transmission speed: Transmission speed is the most critical decision to be made in selecting communications facilities for distributed systems networks. Usually the speed of the service, including circuits and modem, is directly associated with the cost; higher speed means higher cost. Availability of service from the common carrier, the volume of traffic, and the need for responsiveness during interactive access are key factors in selecting the transmission speed.

In many areas, only leased-line service is available to 9600 bps on voice-grade circuits.

- In most countries 4800 bps leased-line service is offered.
- Some countries also allow 4800 bps service on a switched circuit (dial-up) basis, although usually with only two-wire circuits.
- Nearly all countries provide both switched and non-switched (leased line) service at 2400 bps.

Line conditioning: By applying certain internal compensation to a communications circuit, the common carrier can remove noise and other degrading characteristics to improve the quality of the circuit. Special line conditioning is not always available in every area, but, fortunately, recent advances in modem electronics frequently obviate the need for line conditioning.

It is recommended that you seek the advice of the common carrier and modem suppliers regarding line conditioning and the class of service desired at each location in the network.

Modem supplier: Performance, availability, and cost should be considered in selecting a modem supplier. Generally, common carrier modems are furnished on a monthly rental basis, while third party modems are usually purchased, but sometimes may be leased. Costs and rates very widely. In many countries modems are obtainable only from the common carrier.

Recommended modems

HP 3000 networks operate with the modems listed below. If you choose other modems you are responsible for working with their supplier to ensure that the units are fully equivalent to the HP recommended equipment.

For Canada and the United States:

Switched service (dial up)	
Bell 201C	(2400 bps)
Bell 208B	(4800 bps)
Leased line service	
Bell 201C	(2400 bps)
Bell 208A	(4800 bps)
Bell 209A	(9600 bps)

For international areas:

Switched service (dial-up)

Use the locally approved modem compatible with CCITT recommendation V.26 bis with modulation alternative "B."

Operation with CCITT V.23 modems and Datel 2400 service using type 7 modems (in the United Kingdom) is not recommended, although locally acceptable in a few countries.

Either the user or Hewlett-Packard must demonstrate satisfactory operation with the locally supplied modem prior to HP's acceptance of responsibility for compatible operation with the supplied communications software. In many countries the RACAL/MILGO 26 LSI modem is an approved alternative to common carrier (PTT) modems. It complies with CCITT V.26 bis for switched service. Check with your HP representative regarding availability of switched service in your area.

Leased line service

For 2400 bps service in compliance with CCITT V.26, use RACAL/MILGO 26 LSI.

For 4800 bps service in compliance with CCITT V.27, use RACAL/MILGO MPS 48.

For 9600 bps service in compliance with CCITT V.29, use RACAL/MILGO model 96.

Other locally supplied modems which are compatible with CCITT V.26 (2400 bps), CCITT V.27 (4800 bps), or CCITT V.29 (9600 bps), and locally demonstrated as compatible with the HP communications hardware and software may also be selected. Check with your Hewlett-Packard representative regarding recommended alternatives.

Index

Worldwide support

Sales and Service Offices

For more information on Hewlett-Packard computer systems, contact your local Hewlett-Packard representative, or write

Hewlett-Packard
General Systems Division
Marketing Dept.
5303 Stevens Creek Blvd.
Santa Clara, CA 95050
Telephone (408) 249-7020

In Europe: Hewlett-Packard S.A.
7, rue du Bois-du-Lan,
P.O. Box CH-1217 Meyrin 2
Geneva, Switzerland
Tel: (022) 82 70 00

In Japan: Yokogawa-Hewlett-Packard
29-21, Takaido-Higashi 3-chome,
Suginami-ku, Tokyo, 151
Tel: 03-331-6111

In Canada: Hewlett-Packard Ltd.
6877 Goreway Drive
Mississauga, Ontario L4V 1L9
Tel: (416) 678-9430

Other International Locations:
Hewlett-Packard
3200 Hillview Ave.
Palo Alto, Calif. U.S.A. 94304
Tel: (415) 493-1501

Part No. 30000-90008
5953-0550(47)
Printed in U.S.A. 6/78

HEWLETT hp PACKARD

Sales and service from 172 offices in 65 countries.
5303 Stevens Creek Blvd., Santa Clara, California 95050