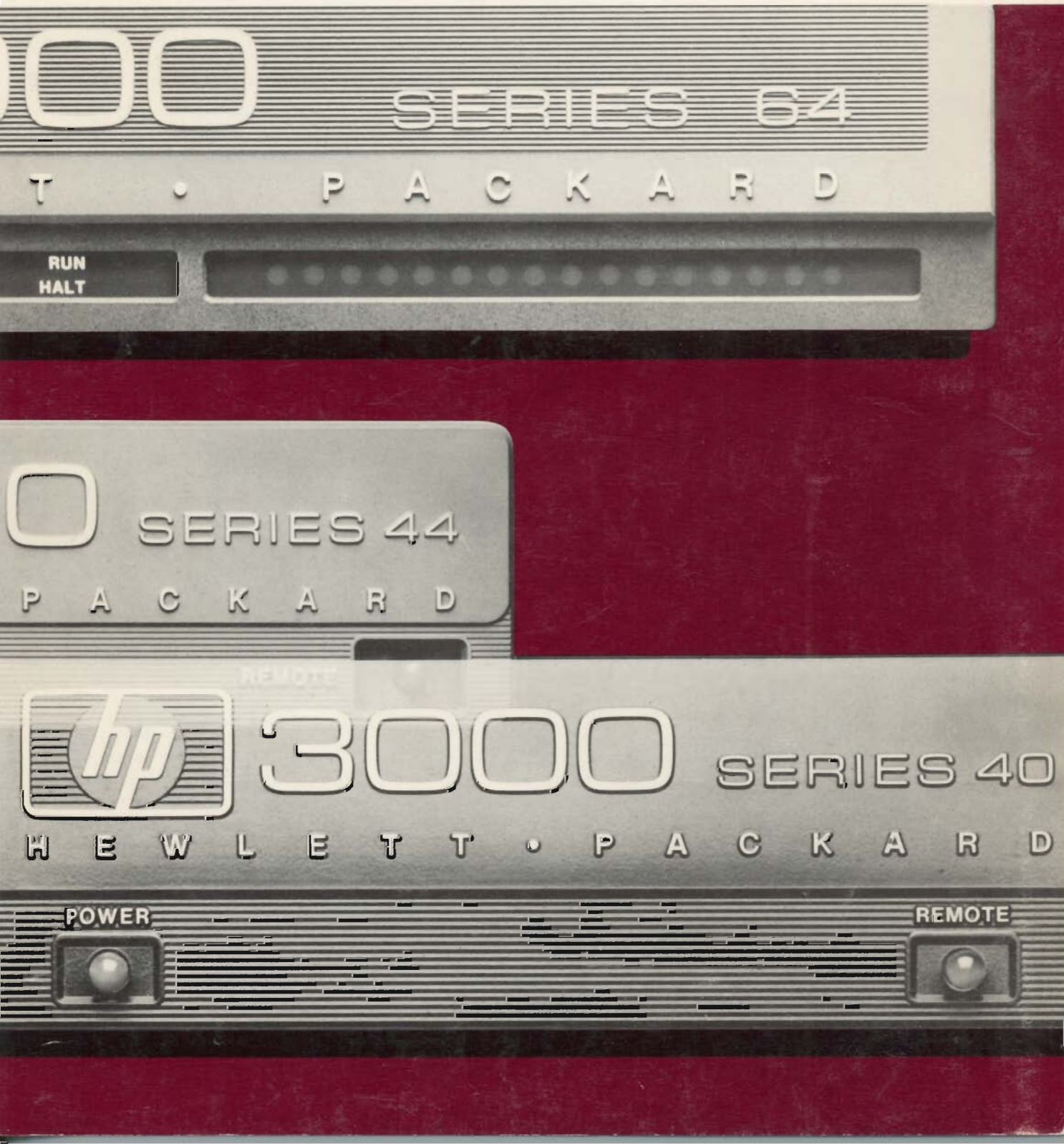
HP 3000 Computer Systems

When performance must be
measured by results

HEWLETT PACKARD

General Information Manual

Effective July, 1982



HP Computer Museum
www.hpmuseum.net

For research and education purposes only.

HP 3000 Computer Systems General Information Manual

HEWLETT PACKARD
19447 Pruneridge Avenue
Cupertino, California 95014

Notice

The information contained in this document is subject to change without notice.

HEWLETT-PACKARD MAKES NO WARRANTY OF ANY KIND WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Hewlett-Packard shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance or use of this material.

Hewlett-Packard assumes no responsibility for the use or reliability of its software on equipment that is not furnished by Hewlett-Packard.

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. HP 3000 computer systems are delivered ready to run application programs. Complete data base management inquiry facilities are included to make the basic HP 3000 system a fully capable and operational system.

Several office oriented software packages are available to HP 3000 users. For reports, documentation, and general office correspondence, a versatile interactive word processor, and two powerful text editors (including a commandless, full screen text editor) are available. A decision support graphics package allows users to interactively design and produce business graphs. Users may also design logos, forms, and even signatures at a CRT for printing on the HP 2680 Intelligent Page Printer, by utilizing the interactive design and formatting systems offered to the HP 3000 user.

For manufacturers, Hewlett-Packard offers user-customizable materials planning and control software. Ten functionally integrated modules are available to address all of the important phases of materials planning and control. Another product consisting of six application modules, provides a user customizable, interactive system for managing the production planning and control function of a manufacturing operation. These interactive, data base oriented HP products can help you improve inventory management, control costs, and obtain more timely and accurate information on which to base purchasing and manufacturing decisions.

For in-house software development, Hewlett-Packard provides tools for efficient interactive program development. With six programming languages, debugging aids, programming utilities, and high level procedures for data base management, data entry, data inquiry and other programming tasks, programmer productivity on the HP 3000 is high.

Advanced distributed processing capabilities through a wide variety of data communications products allow HP 3000 systems to bring information directly to the people who need it, when they need it.

- This manual contains a thorough discussion of the HP 3000 operating system and system architecture. The appendices outline the various operating system and machine level commands and intrinsics, and expand on the hardware features of the computer systems. Other HP 3000 General Information Manuals cover the HP 3000 transaction processing environment, office systems software, data communications, and manufacturing applications systems.

- A separate HP 3000 Specification Guide contains reference sheet specifications of the hardware systems, operating system, hardware peripherals, and software/hardware support services.

Contents

System Introduction

Management Overview

HP 3000 computer systems are a compatible family of interactive, data-base oriented business processing systems. They have an integrated hardware and software design ideally suited to multi-purpose business data processing and dedicated manufacturing applications. Their advanced design provides a wide range of user-oriented capabilities that can be utilized with equal ease on both stand-alone systems and those in distributed processing networks.

HP 3000 systems can be configured in three different ways: as a ready-to-run production system, as a program development system, and as an integrated element of a distributed data processing network (see Figure 1-1).

Ready-To-Run Production Systems

HP 3000 computers are delivered as complete systems with all the hardware and software necessary to run and operate applications programs immediately. In addition to hardware, every HP 3000 computer system includes Fundamental Operating Software (FOS) which makes it a completely functional production system. Included in FOS is the Multi-Programming Executive operating system (MPE) that schedules, monitors, and controls processing, and a complete set of software subsystems for data management, data entry and other capabilities needed in business processing applications. The Fundamental Operating Software enables you to immediately run application programs developed and compiled on any HP 3000 without installing any additional software on the executing system other than the program itself. One HP 3000 can satisfy the program development needs of an entire network of HP 3000 Computer Systems.

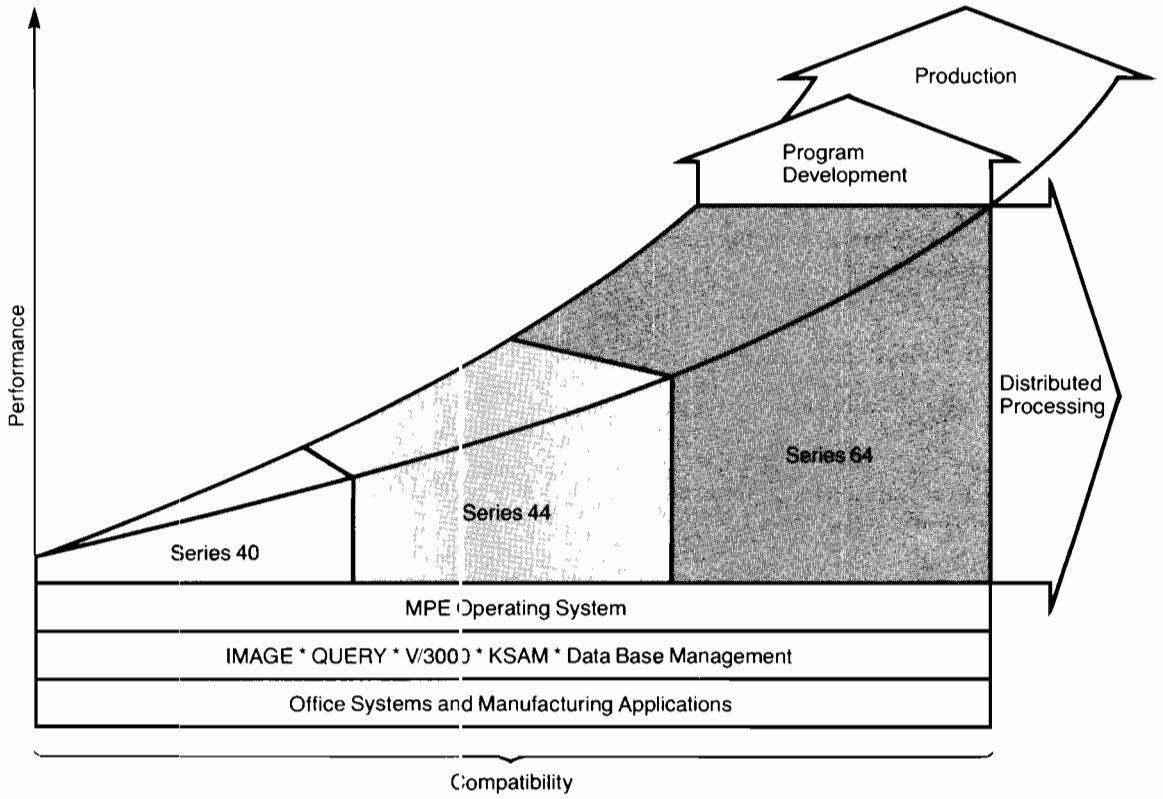


Figure 1-1 Each HP 3000 computer system can be configured to provide the exact mix of capabilities you require along three functional dimensions: Production, Distributed Data Processing, and Program Development.

Program Development Systems

Optional software subsystems, such as language compilers, are available for program development. Highly compatible interfaces between subsystems have been created so that programmers can use the Fundamental Operating Software, in conjunction with the compilers, as a high-level program development tool. The subsystems of FOS include a wide range of high-level procedures for data management and data entry that are callable from any of the HP 3000 programming languages. This enables programmers to give application programs comprehensive capabilities and makes program development on the HP 3000 quick and convenient.

Distributed Data Processing Systems

Data communication subsystems are also available to enable you to set up distributed processing networks that conform to the way you want to do business. A variety of terminal communications products allow you to place terminals wherever you wish without regard to distance or location. HP Distributed Systems software permits sharing of resources between HP 3000 computer systems and other Hewlett-Packard computers. HP 3000 systems can also provide batch and interactive access to IBM mainframe computers. These capabilities are supplied as high level services which free the programmer from complex tasks and allow you to place the processing power where you need it—regardless of the level of expertise at the remote site.

Combined Function Systems

The above three HP 3000 configurations are by no means mutually exclusive. All HP 3000 computer systems can simultaneously perform transaction processing operations, interactive program development, word processing, batch processing, and data communications.

You can distribute your processing according to functional needs or geographical location while retaining the degree of control and security you desire. At each location, you can specify the function and capability you want your HP 3000 to have: a single dedicated application, multiple applications, program development or a combined general use system.

A Commitment to Your Success

Hewlett-Packard is committed to making our products easy to use by non-data processing professionals, whether through HP supplied application programs or the interactive data access facilities for terminal users. Also, the wide range of program development tools makes it easy for programmers to create user-oriented programs.

Superior hardware and software is not enough. Comprehensive support services are available to assist you in maintaining all phases of your operation. The Series 40, Series 44, and Series 64 can all be diagnosed remotely from an HP office by system experts. Another innovative service is the Phone-In Consulting Service (PICS) that enables your System Manager to receive immediate assistance via telephone to resolve software related problems. Personal attention from a well trained Systems Engineer is available whenever you need it. A full range of documentation, training, consulting, and support programs are offered to insure that you have the assistance you need to be successful. Hewlett-Packard supplies a full range of support services and gives you the flexibility to tailor them to your needs.

Hewlett-Packard documentation, training and support services are briefly discussed at the end of this chapter and in more detail in this and other general information manuals.

System Components

An HP 3000 computer system consists of the system hardware, the Fundamental Operating Software, and additional software subsystems, all of which contribute to the easy development and operation of user applications. Hewlett-Packard also provides a set of ready-to-use applications developed for a manufacturing environment. These components and their relationships are illustrated in Figure 1-2.

An Integrated System

The fundamental HP 3000 computer system consists of the system hardware, and the Fundamental Operating Software that includes the MPE operating system, utility programs, data base management, and data entry subsystems.

The HP 3000 software is integrated with an advanced hardware design that includes stack architecture, variable length code segmentation, a hardware-assisted virtual memory scheme, user protection, and dynamic storage allocation. Hardware and software work together, with hardware performing many of the operations conventionally performed by software, such as interrupts or subroutine calls.

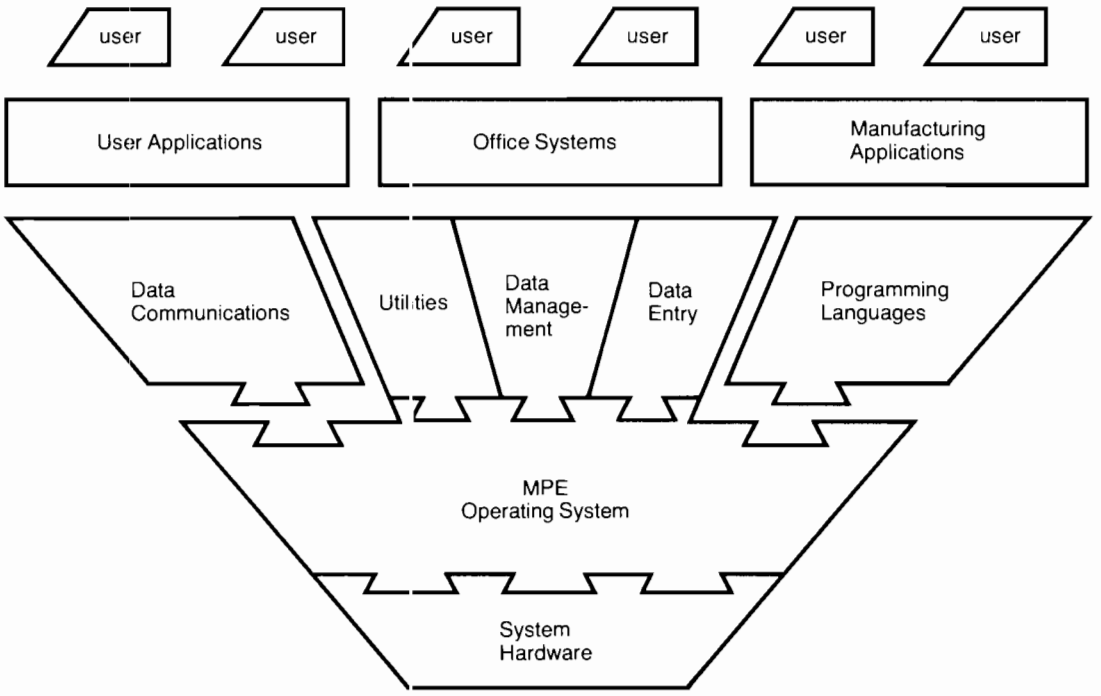


Figure 1-2 HP 3000 Computer System: System Components

System Hardware

The HP 3000 incorporates many features otherwise found only on very large computer systems, including:

- Stack Architecture—provides private, hardware protected data storage for each user plus automatic movement of this data to and from the central processor.
- Virtual Memory—consists of main memory plus an extensive storage area on magnetic disc to provide a total memory far exceeding the main memory size.
- Separation of Code and Data—strictly separate domains for code and data permits code sharing and reentrant execution while maintaining data privacy.
- Fault Control Memory—high speed semiconductor memory modules provide automatic fault detection and single-bit correction.
- Microcode—more than 200 instructions are microcoded operations within the system, plus a full set of microcoded system operations.
- Concurrent I/O and CPU operation—allows input/output to be performed concurrently with CPU operation.

Although the same software is used by all HP 3000 computer systems, specific system hardware differs for the Series 40, Series 44, and Series 64. The Series 64 is described in Appendix D, the Series 44 is described in Appendix F, and the Series 40 is described in Appendix G. Generally, HP 3000 system hardware includes the central processor unit (CPU), main memory, and various peripheral devices available on each system series. Refer to Chapter 3 for a full discussion of the system architecture, and the HP 3000 Configuration Guide for configuration details.

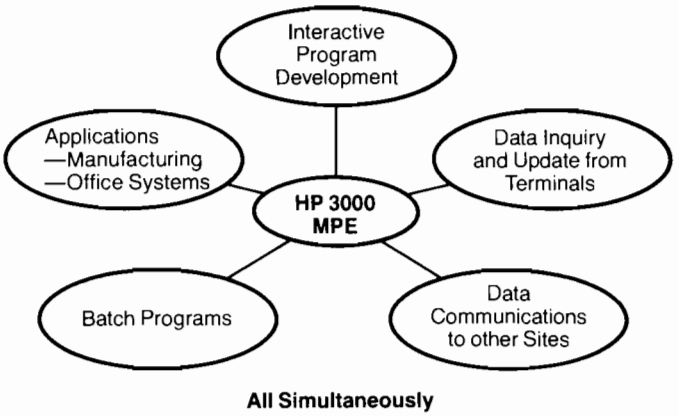


Figure 1-3 HP 3000 MPE Operating System controls and supervises all processing.

Fundamental Operating Software

MPE Operating System

The Multiprogramming Executive operating system (MPE) is the disc-based software system which supervises the processing of all programs that run on the HP 3000. MPE dynamically allocates such system resources as main memory, the central processor, and peripheral devices to each program as needed. In addition, MPE coordinates all user interaction with the system, providing an easy to use command language interface and a powerful programmatic interface in the form of intrinsics and a versatile file system.

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. By relieving you of many program controls, input/output, and other housekeeping responsibilities, MPE makes the HP 3000 extremely easy to use.

The major features of the operating system are:

- Multiprogramming: Concurrent transaction processing, data communications, on-line program development 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, PASCAL and SPL
- File system with file backup, user logging, security and Interprocess Communications
- Access 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

Data Base Management

A wide choice of data base management facilities is available to HP 3000 users. The award winning IMAGE/3000 data base management system allows information to be related logically between data sets (files), minimizing data redundancy and facilitating information retrieval. IMAGE/3000 handles multiple files and makes it easy to define and create a data base tailored to your specific needs. QUERY/3000 allows both programmers and non-programmers to access an IMAGE data base with simple English-like commands. The Keyed Sequential Access Method subsystem (KSAM/3000) also 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 HP Relational Dictionary/3000 is an on-line dictionary and directory that supports the definition of, and relationships between, data items, IMAGE data bases, KSAM files, sequential files, programs, HP VPLUS/3000 forms, usage functions and security rules. It can serve many different needs ranging from a simple support tool to a central information base from which all data processing functions are driven.

Utilities

A set of utility programs, standard on each HP 3000, eases program development and file manipulation and aids in system administration. The utilities included in the Fundamental Operating Software are:

- **EDIT/3000**—A powerful and easy to use text editor
- **FCOPY/3000**—A program for general file copying
- **SORT-MERGE/3000**—A facility for ordering records in a file and merging sorted files
- **System Utilities**—provide administrative controls, reports on system resources, and other special purpose capabilities.
- Facility to execute compiled programs without the source language compiler on the system.

The utilities, data entry and data management software subsystems are described more fully in the HP 3000 Software General Information Manual.

Additional Software Subsystems

In addition to the Fundamental Operating Software, a full set of programming languages, extensive data communication and data entry facilities are offered for use on HP 3000 computer systems.

Languages

Designed to provide you with language flexibility, the HP 3000 offers six high-level programming languages which let you select the language best suited to the task. Programs can be written in:

- **COBOL II**
- **RPG**
- **FORTRAN**
- **BASIC**
- **PASCAL**
- **SPL** (Systems Programming Language—a high level machine dependent language that takes full advantage of HP 3000 design features)

Applications written in different languages, or a combination of languages, can be run simultaneously. Data files and peripheral devices can be used in common by programs in any language without program changes.

Within MPE and the other subsystems of the Fundamental Operating Software reside high level procedures for terminal handling, data entry, and data management. These procedures, or intrinsics as HP refers to them, can easily be called from HP 3000 programming languages. Thus, the addition of a compiler is all that is needed to equip the HP 3000 as a program development machine. Because FOS is standard on all HP 3000's, compiled programs run on all HP 3000 computer systems and can take full advantage of the data entry, data management, and other high level program calls.

Performance Measurement Software

As applications grow, demands on HP 3000's grow. HP recognizes the desire to manage this growth to ensure maximum system performance. For the system analyst, the On-line Performance Tool/3000 (OPT/3000) performance measurement package is designed to help HP 3000 users gather and utilize performance data for system management, capacity planning, and application development activities.

The OPT/3000 package is discussed in detail in Chapter 2.

HP Distributed Systems Network (DSN) Software

The Distributed Systems Network (DSN) is a comprehensive set of data communications capabilities that cover three broad areas: Terminal to HP system communications, HP system to HP system communications and HP system to IBM mainframe communications.

These sets of capabilities provide the customer with a powerful set of tools to distribute his processing both functionally and geographically, thus making information and computer resources available at locations they are needed.

The Point-to-Point and Multipoint Terminal Capability

In an environment where a large number of users need quick and easy access to the system it is desirable to be able to connect a number of terminals to the system. The point-to-point terminal capability provides a means of accomplishing this; this capability allows multiple terminals to be attached locally to an asynchronous port through either an EIA RS-232-C or RS-422 interface. This capability is also available remotely over a supported modem by using a EIA RS-232-C or RS-449 interface. The number of terminals that can be so connected depends on the specific model of HP 3000 being used.

When the number of terminals that have to be connected to the system and the distance become large the cost of individual lines becomes prohibitive. The DSN/Multipoint Terminal Software (DSN/MTS) capability provides an economical way of connecting multiple terminals to the HP 3000 over a single data communications line either locally using the DSN/Data Link, or remotely over supported modems.

HP Systems to HP Systems Communication

As companies grow, the need for data processing grows. Increased data processing needs are met through an increase in the number of systems that must be used. This results in an increased need for exchanging data between systems and a need for sharing system resources.

DSN provides a means for distributing data and resources among HP systems through the DSN/Distributed Systems (DSN/DS) capability. Specifically DSN/DS provides the following:

- Direct Access of data bases on remote systems
- Access to files on remote systems
- Transfer of files between systems
- Access of remote resources such as peripherals and application programs
- Ability of programs on two systems to communicate

HP to IBM Communications

Often large IBM compatible mainframe computers are installed to process data gathered by many geographically/functionally distributed systems. However, before any data is available to the mainframe computer it must be able to communicate with the distributed systems. HP to IBM (and IBM plug compatible systems) communication products fulfill this need.

Two types of products are available in this category: Products for Batch communications—DSN/Remote Job Entry (DSN/RJE) and DSN/Multileaving Remote Job Entry (DSN/MRJE); and products for interactive communications—DSN/Interactive Mainframe Facility (DSN/IMF). Batch products provide the capability for submitting large jobs or files to the mainframe for further processing whereas the product for interactive communication provides the capability to access data bases on the mainframe (such as IMS/VS and CICS/VS) or to do program development on the mainframe (using TSO). DSN/IMF also allows programs on the distributed HP 3000 systems to exchange data with programs on the mainframe through a set of easy to use intrinsics (callable routines).

These communication capabilities are covered in greater detail in the Data Communications General Information Manual.

Manufacturing Application Software

Materials Management/3000

Materials Management/3000 is a customizable, interactive system for managing the materials planning and control function of a manufacturing operation. Materials Management/3000 consists of ten application software modules:

- Master Production Scheduling
- Rough Cut Resource Planning
- Parts and Bills of Material
- Routings and Workcenters
- Material Issues and Receipts
- Inventory Balance Management
- Work Order Control
- Purchase Order Tracking
- Material Requirements Planning
- Standard Product Costing

A discrete manufacturer who assembles standard, multipiece products in lots represents the ideal candidate for Materials Management/3000. However, the software is applicable to most manufacturing operations.

Features of Materials Management/3000

- On-line data base update.
- Customizable user interface and data base.
- On-line terminal data entry, field editing, and error correction using the capabilities of HP's Data Entry and Forms Management System.
- Easy to use, on-line transaction menus.
- Automatic transaction logging.
- Tailorable data entry screens and retrievals.
- On-line assistance via "help" screens.
- Use of "intelligent" HP CRT terminals.
- Pre-defined materials data base.
- Use of proven materials management techniques.
- Advanced security capabilities.
- Automated operator functions.

An important feature of Materials Management/3000 is the ease and speed with which the user can enter, retrieve, and modify data via an interactive terminal. The user has the capability to select a variety of "menu-like" screens on the terminal to perform the tasks associated with materials planning and control.

Materials Management/3000 is available in a form which meets the input, output, and processing requirements of most manufacturing companies. Manufacturing personnel may, however, modify data entry screens, data edits, and information retrieval screens to suit their specific needs. They may also add, delete, and/or modify data items in the Materials Management/3000 data base. *All of these changes can be accomplished easily and without the need for computer programming.*

Materials Management/3000 is a standard (object code) application product which is fully supported by Hewlett-Packard. A comprehensive support package is available and includes:

- 11 user reference and installation manuals.
- 2 customer training courses.
- Planning, implementation, and customization consulting by Hewlett-Packard Manufacturing Industry Specialists.
- A wide range of software support services.

Materials Management/3000 is currently available in several language versions including German, French, Finnish, Swedish, Italian, Norwegian, and British as well as American English.

Each of the ten Materials Management/3000 modules is related and integrated with the others. By maintaining bills of material and current information about parts, current inventory and order information, and time phased master schedule information, the customer can take advantage of material requirements planning to meet the company's production plan. Additionally, by storing labor, material, and overhead costs in the bills of material and bills of labor, the user can calculate standard costs for each item in inventory.

Production Management/3000

Production Management/3000 is a user-customizable, interactive system for managing the production planning and control function of a manufacturing operation.

Production Management/3000 consists of six software application modules:

- Routings and Workcenters
- Work-In-Process Control
- Work Order Scheduling
- Shop Floor Dispatching
- Work Order Tracking
- Capacity Requirements Planning

A discrete manufacturer who assembles standard, multi-piece products in lots represents the ideal candidate for Production Management/3000. However, the software is applicable to most manufacturing operations.

Features of Production Management/3000

- On-line data base update.
- Customizable user interface and data base.
- On-line terminal data entry, field editing, and error correction using the capabilities of HP's Data Entry and Forms Management System.
- Easy to use, on-line transaction menus.
- Automatic transaction logging.
- On-line assistance via "help" screens.
- User manuals tailored to each job function.
- "Intelligent" HP CRTs and Factory Data Capture terminals.
- Use of proven production management techniques.
- Advanced security capabilities.
- Automated operator functions.

An important feature of Production Management/3000 is the ease and speed with which the user can enter, retrieve, and modify data via an interactive terminal. The user is presented with a series of "menu-like" screens from which any desired function can be easily selected using "soft keys".

Production Management/3000 is available in a form which meets the input, output, and processing requirements of most manufacturing companies. Manufacturing personnel may, however, modify data entry screens, data edits, and information retrieval screens to suit their specific needs. They may also add, delete, and/or modify data items in the Production Management/3000 data base. *All of these changes can be accomplished easily and without the need for computer programming.*

Production Management/3000 is a standard (object code) application product which is fully supported by Hewlett-Packard. A comprehensive support package is available including:

- Comprehensive user reference and installation manuals
- Two customer training courses
- Planning, implementation and customization consulting by Hewlett-Packard Manufacturing Application Specialists
- A wide range of software support services

Production Management/3000 will soon be available in several language versions including German, French, Finnish, Swedish, Italian, Norwegian, and British as well as American English.

The six functional modules of Production Management/3000 are integrated with one another to provide a comprehensive set of production management functions. In addition, interfaces exist which allow HP's Materials Management/3000 (or some other materials system) to complement the functions of Production Management/3000 and provide a comprehensive manufacturing control system.

Office Systems Software

Using the HP 3000 as the foundation, Hewlett-Packard's Office Systems products can help to bring your office into the computer age. We call it The Interactive Office . Hewlett-Packard's commitment to increase productivity in your office begins with tools for document management, decision support, personal support and organizational communication. Best of all, The Interactive Office products operate on all HP 3000 systems simultaneously with your data processing activities.

Document Management

An important step towards managing office information starts with controlling the voluminous amounts of text that must be handled every day. HP's document management products offer flexibility for the different types of document needs throughout your office.

The HPWORD word processing software teams with the HP 2626W Word Processing Station and HP 2601 Daisywheel Printer to provide a powerful secretarial system. HPWORD is a complete, powerful word processor that new users can easily learn through self-paced training materials, and experienced users will appreciate because of its advanced operator-oriented features. HPWORD provides a full complement of document creating, editing, storing, printing, and recalling capabilities for quick memo, letter, and report generation. In addition, the HP 2626W can operate as a full data processing terminal on the HP 3000.

Text and Document Processor/3000 is a full capability text editing and document formatting system for the HP 3000. Using a command environment, TDP/3000 has extensive formatting features that can be used to create manuals, contracts, and lengthy proposals. TDP/3000 operates with a variety of HP terminals and printers.

HPSLATE is a commandless, text processor that has a menu driven set of functions for entering, formatting, revising, printing, and saving shorter documents. Operating on a variety of terminals, HPSLATE utilizes screen-labeled function keys to perform the various editing activities.

Decision Support

Having timely access to accurate, more detailed information can help you make better decisions. The problem, however, is to quickly identify what information is relevant to each business decision. With HP Decision Support products, you have a more effective way to interpret information.

Decision Support Graphics/3000 is a fully interactive graphics software package that lets users design business charts without programming. It is a powerful management tool that offers many capabilities for efficient and timely management decision making. Requiring no special programming knowledge or graphics design knowledge, DSG/3000 is truly for the non-computer professional who needs the convenience of presentation graphics output at the press of a button.

Organizational Communication

Effective and timely communication is essential to your organization's day-to-day business. Information communication is critical to maintain accurate up-to-date records on accounts and documents which are distributed throughout your firm.

The HP 2680 Laser Printing System adds a new dimension to the way you distribute your office information. Coupled with interactive software, the HP 2680A delivers clear, easy-to-read reports produced from HPWORD or TDP/3000 at 45 pages per minute on letter sized paper. Illustrated with symbols, logos, and specialized characters, these reports can be printed on business forms generated and stored electronically on the HP 3000.

The HP 3000 Distributed Systems Network (DSN) ties together your systems for distributed information processing. With DSN products, reports and documents can be sent among HP 3000 systems, between HP 3000s and other HP systems, and between HP 3000s and IBM or IBM-compatible mainframes.

Transaction Processing Tools

Several new productivity tools are offered to complement HP 3000 users in information management and retrieval. These new products offer a new and revolutionary concept in data management and application development. HP Dictionary/3000 provides for centralized control and standardization of information resources and creates logically related groups of information for fast user reporting. HP Report/3000 is a non-procedural report-writer that can generate and format simple or complex reports from IMAGE data bases, KSAM, or MPE sequential files. HP Transact/3000 consists of a transaction processor and a high-level task-oriented language for fast application development.

To simplify data entry, HP VPLUS/3000, a comprehensive data entry and forms management subsystem provides both a ready to use data entry program, ENTRY, and a programmatic interface for your terminal-oriented application programs. It includes facilities for immediate on-line entry and modification of data, a wide range of data editing and validation, record reformatting, and interactive forms design and forms management.

HP PLUS—A Program For Locating User Software

HP PLUS provides HP customers with more software solutions. Many third-party software suppliers have written applications and utilities for the HP 3000 computer systems. The HP PLUS program locates this software and matches customers with the most appropriate package in one of two ways: by enabling customers to order these application packages directly through HP sales representatives, or by referring customers to the third-party software supplier.

Compatibility

The HP 3000 family of compatible business systems is comprised of the Series 40, Series 44 and Series 64 systems, each of which offers a full range of peripheral options and expandable hardware configurations. All HP 3000 systems feature compatible system software and application programs. Applications developed on the HP 3000 system can be executed on any other HP 3000 system. Each HP 3000 is supplied with the facility to execute compiled programs without the source language compiler on the system. The systems differ only in performance and expandability.

Features Common to the HP 3000 family

- Patented stack architecture
- Virtual memory
- Automatic memory fault detection and fault correction
- Microprogrammed CPU
- Multiprogramming operating system
- Complete system security and automatic accounting of resources
- Concurrent CPU and I/O operations
- Integrated terminal access with modem support
- Rechargeable battery packs to maintain memory data during power failure
- Automatic restart after power failure
- Microprocessor based diagnostic and control unit
- Fundamental Operating Software
- Software compatibility between all current systems.

Availability of software and hardware products is subject to change. Refer to the HP 3000 Configuration Guide for the most current product availability information.

Users

The final element in an HP 3000 computer system is the user. You interact directly with the computer system through the software subsystems and the operating system which in turn uses the hardware and the system peripherals. You may also interact indirectly through an application program. The tasks you perform usually fall into one of the following three categories:

- On-line interaction with application programs
- Program development
- System management

All of the system resources previously described are at your fingertips when you sit down at a terminal and access an HP 3000 computer system. Hewlett-Packard is committed to making the time you spend at the terminal friendly and highly productive.

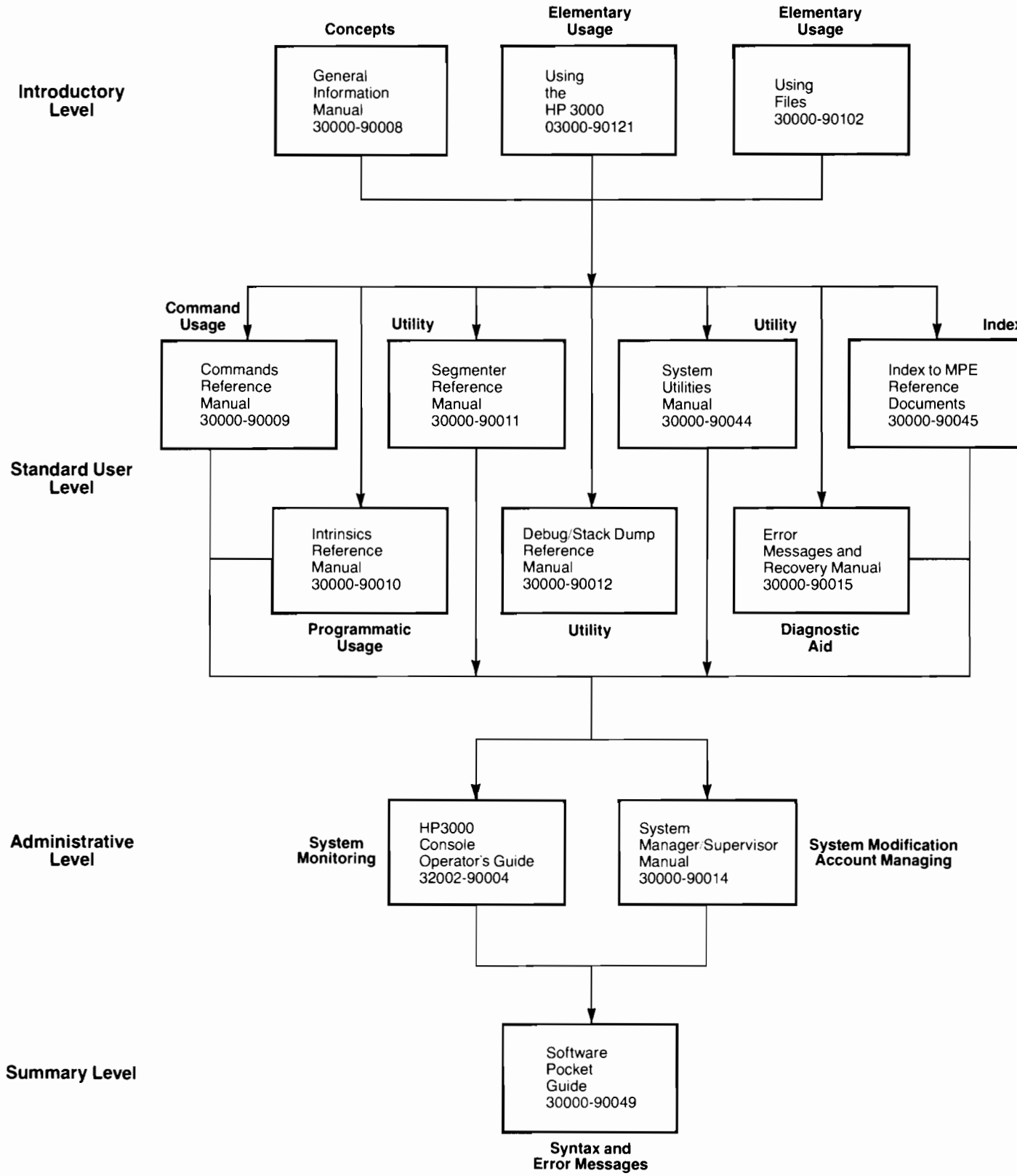
Documentation

Included with the HP 3000 computer system is a comprehensive set of user manuals. Complete documentation is provided for the operating system, and for special tasks such as system installation and program conversion, in addition to documentation for individual subsystems.

Figure 1-4 shows the four functional levels of documentation provided and how the manuals relate to one another. A more complete set of system related manuals available for the HP 3000 is presented in Appendix A.

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. Particularly informative are the "Using" series of manuals (Using the HP 3000, Using Files, Using HP VPLUS/3000) that contain practical examples showing the new user how to perform a variety of common tasks. These include how to build files, design terminal screens, compile and run programs, equate files to input/output devices, and obtain hardcopy listings.


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 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
HP3000 Console Operator's Guide 32002-90004
System Manager/Supervisor Manual 30000-90014
System Modification Account Managing
Summary Level
Software Pocket Guide 30000-90049
Syntax and Error Messages

Standard User Level Manuals

These manuals provide complete reference documentation for the full line of HP 3000 software. They fall into two general categories:

- Program Development manuals—provide complete reference specifications for the programmer on using the operating system, the various language compilers, the data base and file management subsystems, as well as the data editing and data entry subsystems, and all the HP 3000 utility programs. An Error Messages and Recovery Manual summarizes the error messages and recovery actions for operating system, language, and utility subsystem errors. Another useful manual, the Index to the MPE Reference Documents, provides a master index to the entire set of MPE operating system manuals.
- Applications manuals—provide full instructions for non-programmers on using the Hewlett-Packard application programs available for use on an HP 3000 computer system.

Administrative Manuals

These manuals provide a guide for the day-to-day management of the system. System monitoring is covered in the HP 3000 Console Operator's Guide. The more complex information required for system modification and account managing is covered in the System Manager/Supervisor Manual.

Summary Level Manuals

These are a set of convenient pocket size manuals that summarize the reference specifications for the MPE operating system, and for various programming languages and subsystems. These guides are primarily memory aids and assume a familiarity with the effects of the syntax they summarize. An essential pocket guide is the Software Pocket Guide that summarizes the MPE commands and system procedures, the file system error messages, and the commands that control the more commonly used utility programs.

A quick reference guide is available for the data entry operator who uses the HP VPLUS/3000 data entry application program, ENTRY.

Training

Hewlett-Packard offers a variety of training courses that enable you to derive maximum benefit from the system's capabilities. A full curriculum of training courses spans the needs of a variety of users—from the system administrator, to the program developer, to the non-technical user of a software application.

Among the courses offered are:

- A Programmer's Introduction
- System Management
- SPL/FILE System Introduction
- HP 3000 Special Capabilities
- System Operator
- Application Design
- HP VPLUS/3000
- HP DSG/3000 Programmatic Use
- IMAGE/DBMS 3000
- Using Materials Management/3000
- Using Production Management/3000
- HP Manufacturing Systems Customization and Operations
- MPE Internals
- System Performance Training for OPT/3000
- TDP/3000
- Distributed Systems/3000
- Programming in Transact/3000
- Using Dictionary/3000

Courses may last from one to ten days, and in most cases, can be presented at your facility or at an HP Technical Training Center. Experienced professional instructors and hands-on computer time for students combine to make the training experience an invaluable asset for your operation.

Self-Paced Training

In addition to classroom instruction, Hewlett-Packard offers self-paced courses which allow users to learn, unassisted, at their own facility according to their own schedule. Among the courses offered are:

Self-Paced Learning

- HP 3000—A Guided Tour
- Learning Cobol II
- Using DSG/3000

Full descriptions of the courses offered are given in the HP 3000 Software Specification Guide.

Software Support Services

A well-defined set of software support services is offered with HP 3000 computer systems. Since customer support needs can differ considerably, these support services are available to provide a flexible range of support. The software services are divided into two main categories:

- **CSS—Customer Support Service,** for customers who choose a close support relationship with Hewlett-Packard.
- **SSS—Software Subscription Service,** for customers who prefer to rely on their own resources for software support, but still want to receive documentation and updates.

Documentation Distribution Services

In addition to the software support services described above, Hewlett-Packard offers two types of Documentation Distribution Services that are appropriate for customers with a large programming staff who wish to be individually informed of software problems or keep their documentation up to date. These services are:

- **SNS—Software Notification Service,** provides one copy of the Software Status Bulletin and the Communicator.
- **MUS—Manual Update Service,** provides one copy of updates or new editions to manuals automatically whenever they are issued. The various sets of manuals that can receive this support are specified in the Configuration Guide.

The reference sheets in the HP 3000 Specification Guide provide a full description of the Software Support Services outlined above.

Hardware Support Services

Hewlett-Packard offers a full range of system and product maintenance services so that you can select the support program that best meets your needs. Your Sales Representative along with your Customer Engineer, will assist you in how to best plan your support program.

These services are purchased under the Customer Support Services Agreement (CSSA) which provides a known monthly maintenance cost that can be budgeted. All hardware services include the necessary parts and labor for remedial maintenance. Engineering improvements are also installed to assure thay your system performs to Hewlett-Packard's high standards. A short term maintenance option, from 90 days to one year, is available for OEMs.

The following are the services available under the

System Maintenance Services

System maintenance services provide the most comprehensive hardware support program HP offers. The highly-trained HP Customer Engineer assigned to your account works with your team to optimize system availability. The following system maintenance services are available:

Guaranteed Uptime Service (GUS). This service provides the highest level of maintenance service for Hewlett-Packard's most advanced computer systems. This service is designed for the user who requires maximum system availability. The following features are included:

- System uptime to exceed 99% over any three consecutive month coverage period.
- Service credit of one month's charges for all products covered by GUS whenever the uptime percentage is reported below 99%.
- Monthly activity report is provided listing all maintenance activity over the reporting period.
- Round-the-clock continuous coverage, four hour response to all service requests within 100 miles of an HP Service Responsible Office (SRO).
- Common phone number for hardware and software support.
- GUS coverage provided during warranty at no additional charge when service is ordered prior to installation.
- All account management features are included under GUS.

Standard System Maintenance Service. This service provides a high level of service with the quickest response. A choice of nine different coverage periods—13, 16 or 24 hours per day, 5, 6 or 7 days per week—are offered to meet the specific needs of your operation. The following features are included:

- Same-day response, typically within 4 hours on all service requests placed during normal working hours, at sites within 100 miles of a Service Responsible Office.
- Site Environmental Surveys performed periodically to assure a sound operating environment.
- Scheduled preventative maintenance to identify potential problems before they occur.
- Work will continue even after your coverage period has ended once on-site work on a remedial repair has commenced and progress is being made.
- Installation services for HP computer system products added to your system already covered by this maintenance service are included at no additional charge.
- Warranty services will be extended to match the coverage provided under this maintenance service at no extra charge.

Basic System Maintenance Service. This service offers all the features of the Standard System Maintenance Service, but with a lower response for a reduced cost. Next-day response, Monday thru Friday, to service requests for sites within 100 miles of an HP Service Responsible Office is provided. Other response times are specified for service beyond 100 miles.

Product Support Services

Product Support services are provided exclusively for HP workstation products—terminals, small printers, and plotters. Augmenting spare units with these services will yield a very high degree of equipment availability at the lowest possible prices. The following product support services are available:

On-Site Product Maintenance Service. This service is Hewlett-Packard's lowest cost on-site support program. Offered is a next-day response, Monday thru Friday, to service request for sites within 100 miles of an HP Service Responsible Office. Scheduled preventative maintenance for products covered is typically unnecessary or performed by the user.

Field Repair Center Maintenance (FRC) Service. FRC service offers the lowest hardware support costs for selected HP workstation products. An approximate 50% cost savings over other on-site programs can be realized through this program. In the event repairs are necessary, the unit is shipped to the nearest HP Field Repair Facility. HP will repair the units and reship it back to you within 3 days of receipt at the Field Repair Center.

A 90-day on-site warranty is included for all HP computer products purchased with your HP 3000. After the warranty period, service is continued under the Customer Support Services Agreement.

The HP *Computer Systems Support Services Data Book* contains more information about all Hewlett-Packard support service product specifications.

Consulting Services

To help you improve your productivity with your HP 3000, Hewlett-Packard offers on-site Systems Engineering consulting, both as standardized services and on a time and materials basis. Standardized consulting services provide assistance for clearly defined and commonly encountered areas of need. Time and materials services are also available for problems which do not fall into the standard categories.

The standardized consulting products presently available for the HP 3000 include:

- **Installation Management**—This service is designed to help the System Manager of a new installation become immediately productive by reducing start-up problems, customizing procedures to your objectives, and planning for long-term operational success. This includes discussion and recommendations on performance variables, accounting structure design, back-up strategy, system start-up methods, problem management, disaster recovery, job scheduling and load management.
- **System Performance Evaluation**—This service can help you optimize system performance by identifying bottlenecks and their causes, and by recommending a strategy for corrective action. A specially trained HP Systems Engineer uses HP software to analyze the system's job mix and workload, and recommends changes in areas such as job mix, memory management, utilization of I/O devices and files, and subsystem configuration.

The scope of these services is described in individual data sheets, available from your local HP sales office. For more information on Consulting Services, please contact your HP Sales Representative or Systems Engineer.

HP 3000 International Users Group

The Users Group is an independent world-wide organization for the purpose of exchanging techniques and ideas among HP 3000 users. Hewlett-Packard's liason works closely with the Users Group to promote communication between Hewlett-Packard and HP 3000 users. Membership is open to any interested individual using an HP 3000. You will find that meeting with other users and exchanging ideas provides a stimulating environment in which to sharpen your programming and operational techniques. For further membership information contact:

HP 3000 International Users Group
289 S. San Antonio Rd.
Los Altos, CA 94022 U.S.A.

System Software

MPE Operating System
Performance Measurement
Flexible Disccopy

MPE Operating System

The functional heart of the HP 3000 is the Multiprogramming Executive, MPE. This general purpose, disc-based operating system supervises all processing and maintains all user interface with the HP 3000. Two major attributes of MPE are its versatility and ease of use.

Designed to take full advantage of the computer's hardware features such as virtual memory and stack architecture, MPE demonstrates its versatility by enabling the HP 3000 to perform transaction processing, on-line program development, data communications, and batch processing concurrently. In addition, MPE permits system resources to be accessed simultaneously by multiple users, each of whom interfaces with the system independently.

MPE demonstrates its ease of use with its many user assistance features such as a powerful, straightforward command language and an on-line HELP facility which guides you in using MPE commands. In addition, MPE simplifies the programming task by monitoring and controlling program input, compilation, execution, and output. MPE regulates the order in which programs are executed, and dynamically allocates any hardware and software resources the programs require.

A complete account structure and automatic resource accounting are standard features of MPE. Easy to use MPE commands allow the system manager to set up a hierarchical accounting structure on the system in a style similar to a company organization chart. MPE then automatically keeps track of the system resources used by the various groups in the account structure. This resource usage information can then be used for billing, accounting, or any other application that requires such data.

MPE provides complete security, enabling you to operate in an environment protected from interference or illegal access by other users. This security is accomplished by means of multiple logon passwords, file lockwords, heirarchical access restrictions, and user capability sets.

MPE's interprocess communication facility allows processes to communicate efficiently with each other via disc files. All processes which have access to a message file via the security system may communicate with each other using standard file system intrinsics.

MPE handles all input/output to peripheral devices, receiving the I/O requests, queueing them if necessary, and performing the actual data transfer. Because MPE treats I/O devices as files, you can write programs without concern for the physical source or destination of the data, and you can run them in either batch or interactive mode without changing the names of the files they reference.

MPE also handles asynchronous terminal communications. Synchronous data communication to terminals, other HP computers, or even non-HP computers, is provided by optional data communications subsystems.

Figure 2-1 illustrates the major components of the Multiprogramming Executive Operating System.

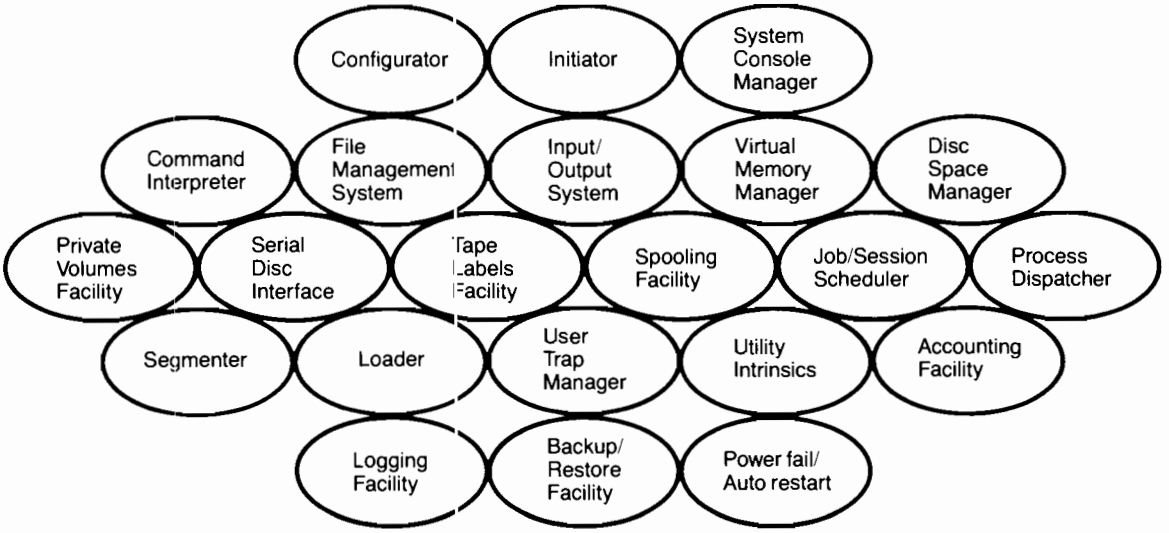


Figure 2-1 Components of the MPE operating system.

Efficient Versatility

All HP 3000 computer systems operate under a single operating system—MPE. This means that programs prepared under MPE can be run without modification on any other HP 3000 operating under MPE, provided that all devices required by the programs are connected on-line. It also means that you can move from one HP 3000 to another without undergoing additional training for a new environment.

Multiprogramming

One of the major ways in which operating efficiency is achieved in an HP 3000, is by multiprogramming—the concurrent execution of multiple programs. Multiprogramming allows system resources to be allocated among several competing programs. While one program is awaiting an I/O operation, for instance, control of the central processor is shifted to the next highest priority program waiting for the CPU. MPE is designed to allocate, schedule, and dispatch control of the central processor, storage, and input/output devices among the competing programs. This controlled competition for system resources reduces turnaround time, and significantly increases system throughput. Operating in conjunction with the architecture of the central processor, MPE provides complete protection against one program interfering with another.

The number of programs that can be processed concurrently depends upon such factors as hardware configuration, program operating modes, and the application programs involved. MPE is designed so that the maximum number of concurrently running programs can be increased or decreased by changing a single system configuration parameter.

MPE allows the concurrent execution of programs from two types of input media—traditional batch input devices and interactive terminals. Programs are independent of their input mode and the same system code is used to perform particular functions in either mode. This results in storage economy and reduced overhead.

Interactive Processing

When using the interactive processing mode, you enter commands and data through a keyboard terminal and receive immediate responses to your input. This type of interaction is called a session and is especially useful for program development, text editing, data entry, information retrieval, computer-assisted instruction, and other applications where a direct dialogue with the system is preferred. Sessions can be used to access:

- Operating system commands and subsystems
- Language and utility programs
- Data base management programs
- Data communications programs
- Application programs
- Office System programs

A session begins when you enter the HELLO command from an on-line terminal and MPE connects you to the command interpreter. You may then enter commands to use language compilers or other subsystems such as the text editor, to run programs, or to modify your files. The session continues until you enter a :BYE command, a new :HELLO command, or the system operator intervenes to abort the session.

As an example, let's assume you want to create a COBOL program and then compile, prepare, and run it during an interactive session. Figure 2-2 shows the various commands entered during such a 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 colon prompt character. You log on to MPE by entering a HELLO command containing your assigned user and account names, then 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 a file named YOURFILE by entering a KEEP editor command, and then terminate the editor subsystem. Now the 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 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 terminate the session by entering the BYE command.

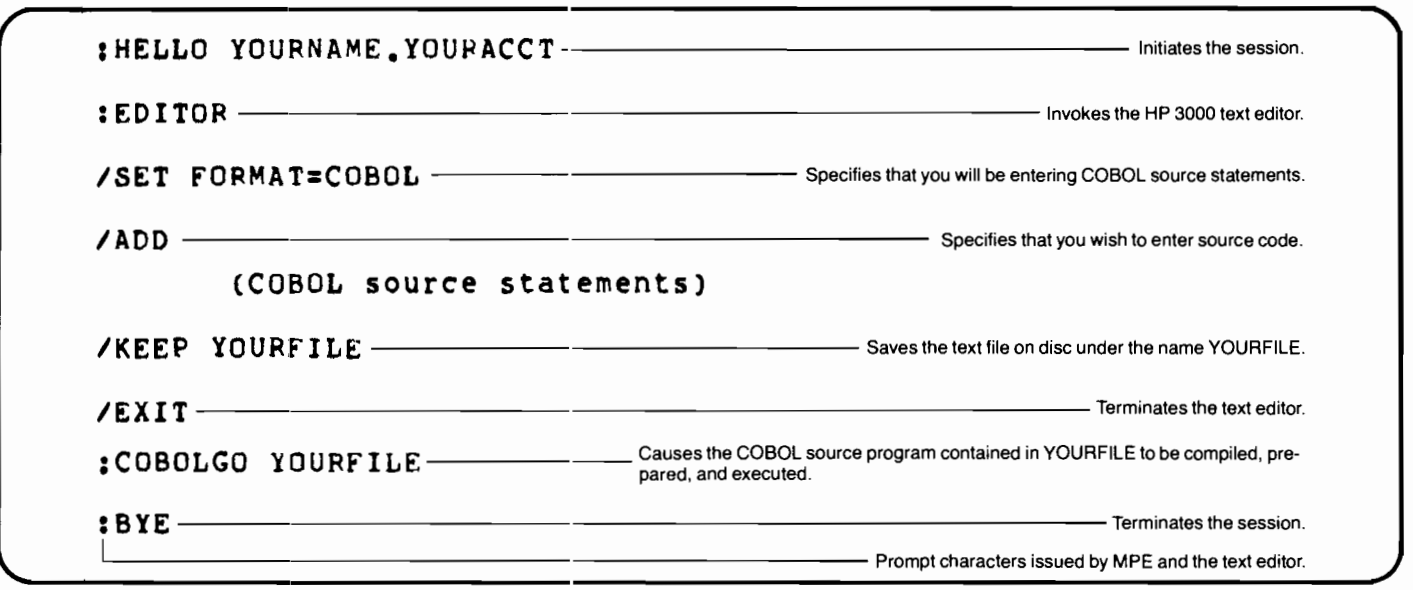


Figure 2-2 Sample session

The example above 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 is a logical extension of the interactive functions available through MPE. Any capability, with the exception of BREAK, that is available in one mode, is available in the other and employs the same MPE commands. Languages, utilities, and applications development software can be run in either batch or interactive mode without changes. The standard input and output devices are automatically redefined.

The batch processing mode 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 the terminal during the course of an interactive session.

Several jobs can be submitted to the system from multiple devices concurrently. Batch job input is spooled on disc and MPE schedules each job for execution according to its job input priority specified in the JOB command. Additional commands are provided for monitoring job selection. The system operator specifies the maximum number of jobs that can be executed concurrently and can dynamically adjust the job selection criteria.

When a job enters execution, the commands within it are executed sequentially 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 job or session (if available) immediately enters execution. Spooled output on disc is selected for output processing according to the output priority specified in the JOB command.

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 will be executed in its entirety, as pre-defined in the job control statements, without active intervention.

A User-oriented Operating System

The many features and capabilities of the Multiprogramming Executive operating system are designed around the concept of the user. There are two distinct types of users:

- the end user, such as a data entry clerk, whose only concern is running a compiled application program, and
- system administrators, such as application programmers, who are responsible for the creation and maintenance of application programs as well as the day-to-day operation of the system.

Each type of user is associated with a particular set of capabilities and responsibilities, and each has access to MPE features which assist him with his specific tasks.

User Classification

Programmers are users who create application programs which run on the system. MPE provides two major areas of system interface for these users: an interactive interface which includes a command language, an on-line HELP facility, and job control facilities; and a programmatic interface which includes programming intrinsics and the MPE file system.

The end user, who can range from an order entry clerk to a functional manager, takes advantage of all the capabilities of the operating system through an application program which he can run without any knowledge of MPE itself.

System administration is performed by a hierarchy of users whose overall responsibility is the successful adminstration of the computer system. The various levels of MPE administration are defined in Table 2-1. In a small installation, a single user may perform the functions associated with all levels of administration. In a larger installation, the capabilities may be divided among several individuals at each administrative level. Thus it is more appropriate to think of "a user with system manager capabilities" rather than a formally titled "system manager."

Table 2-1 System Administrators

System Manager: Manages the overall system by creating accounts (basic structures for user access) and defining resource-use limits.
System Supervisor: Manages the system on a day-to-day basis, controls scheduling queues, alters system configuration, and maintains the system library.
Account Managers: Maintain accounts by defining the valid users and file groups for the accounts and specifying resource-use limits for them.

The system operator is the user who operates the system console and is responsible for responding to all system requests. MPE provides a range of operational capabilities which augment the performance of day-to-day operations such as system start-up, back-up, maintenance and recovery, as well as helping the system operator keep the system operating as smoothly and efficiently as possible.

User Capabilities

Capability sets are assigned to each system user based on the kinds of tasks each user needs to perform. These capability sets are divided into three categories:

- User attributes
- File access attributes
- Capability-class attributes

User attributes include system manager, system supervisor, and account manager capabilities. Additional capabilities such as 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, may also be assigned.

There are two file access attributes: *save*, which permits the user to save files by declaring them permanent, and *non-shareable devices*, which allows a user to use non-shareable devices, such as a magnetic tape unit.

Capability-class attributes refer to the ability to access special MPE facilities. Included here are interactive access, local batch access, process handling, extra data segment acquisition, multiple resource identification numbers, privileged mode, user logging, private volumes, and data communications.

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

User Interface

Three user-oriented software facilities provide a comprehensive interface between the system/application programmer and MPE. These tools are: the MPE command language, the on-line HELP facility, and job control facilities.

Command Language

The simplicity of the MPE command language greatly enhances the system's usability. MPE commands enable you to initiate a session and specify the various MPE operations you wish to have performed. When you specify a command, a portion of MPE called the command interpreter reads the command, checks its validity, and then causes the appropriate action to be taken. After the requested action is successfully completed, the interpreter processes your next command. You may enter commands interactively during a session or through a batch input device. Commands may also be issued programmatically from a running program.

MPE uses a colon (:) to prompt you for a command during an interactive session. When a batch job is submitted, MPE commands within the job are designated by a colon in column one.

The MPE command language is composed of many commands. Each command enables you to request a specific action of MPE. Collectively they provide a powerful system-usage tool. The full range of MPE commands is presented in Appendix B of this manual. The following is a list of common command uses:

- Intiating and terminating jobs and sessions.
- Running system programs or compilers.
- Running programs.
- Running system utilities.
- Creating, managing, or deleting files.
- Displaying file information.
- Displaying job, session, or device status.
- Transmitting messages.
- Assisting in program debugging.
- Establishing communication between a local and a remote computer.

If the command interpreter detects an error during a session, MPE informs you with an error message which specifies the erroneous parameter. MPE then requests that a new command be entered. You can instantly correct command errors by retyping the command or using the REDO command, which allows you to edit the erroneous command, and the session can continue.

During a batch job, MPE lists the error on the listing device. Input from that point through the next EOJ, DATA, or JOB command are usually ignored. You can, however, use the CONTINUE command to request that the job be continued despite the error.

User-Defined Commands

MPE allows you to define your own commands by combining several MPE commands into a command procedure and assigning the procedure a name. The name can then be used as a command. Thus it is possible to enter a single command name which you have defined and cause several commands to be executed. These user-defined command sets can be created by each individual user as well as being made available to entire accounts and all accounts systemwide. It is also possible to redefine existing MPE commands and messages to suit your particular situation.

On-line HELP Facility

Whenever you need assistance with command language syntax or even the name of a particular command, you can invoke the on-line HELP facility. This facility provides graduated information on any MPE command or set of commands. The HELP messages displayed coincide with the information contained in *HP3000 Command Reference Manual*.

Figure 2-3 demonstrates the two ways in which the HELP facility can be used. In the "immediate" mode, you merely enter the HELP command followed by a parameter. Information detailing that parameter is displayed immediately. In the "subsystem" mode you enter the HELP command without any parameters. The system then displays a menu of valid parameters, and prompts you with a greater than sign (>) for the parameter you wish explained.

```
:HELP REMOTE,EXAMPLE
```
Example of HELP being used in the "immediate" mode.

```

EXAMPLE

To establish the communications link 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.

```
:HELP
```
Example of HELP being used in "subsystem" mode.

```
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

Figure 2-3 Using the on-line HELP facility.

Job Control Facilities

MPE contains job control words (JCW) and conditional execution functions which permit the user to design job streams whose execution can be dynamically altered based on the results of previous job steps.

You can use both system defined and your own job control words to store job status information and to pass such information between programs and between a program and the MPE command interpreter. JCWs are defined and accessed by commands from the command interpreter and by intrinsics from your programs.

You can also use JCWs in conjunction with conditional execution function statements. These statements specify a logical expression (TRUE or FALSE), and are evaluated during program execution. If the value found is TRUE, the remaining statements related to that condition are executed. If the value is FALSE, any existing alternative statements are executed instead.

The following example illustrates the use of JCWs and a conditional execution function. The sample job runs a program which edits and verifies transaction cards and counts 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.

```
: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)
```

Program Development

The Multiprogramming Executive provides meaningful assistance with the task of generating application programs. MPE programming assistance includes:

- Consistent command language interface to all compilers.
- Program preparation performed by the MPE segmenter.
- Procedure libraries for external references.
- A device-independent file system.
- Flexible file security.
- Subroutines callable across languages.
- Access to all system intrinsics.

The use of these software tools during program generation is described below.

Creating Programs

Three steps are required to take a program from source form to an executable state. The first step is to compile the source program into relocatable binary modules, called 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, or 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 step is done by the MPE segmenter.

The third and final step is to have the MPE loader 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 data stack.

Often all of these steps are initiated by a single MPE command. (This was illustrated in Figure 2-2.) When necessary, however, you can initiate each step individually, thereby controlling what happens along the way.

Accessing Compilers

Program compilation is the first step in converting a source program to an executable state. The format of the commands used to access a language compiler is consistent for all the MPE language compilers. Thus you do not have to learn a new method of program preparation for each programming language you employ.

Three commands are used in the process of program preparation. The first command compiles the program only and stores the resulting RBMs in a USL file for later use. The second command compiles and prepares the program, creating a program file for use in program execution. The third command compiles, prepares and executes the program.

The compiler name is used in the format of the commands. To illustrate, the commands for accessing the COBOL compiler are:

- COBOL, which compiles the source program
- COBOLPREP, which compiles and prepares the program
- COBOLGO, which compiles, prepares, and executes the program

Access to the other compilers is identical, except that the name "SPL," "BASIC," "RPG," "PASCAL," and "FORTRAN" replace "COBOL" in each of the three commands. The command BASIC invokes the BASIC interpreter, whereas the command BASICOMP invokes the BASIC compiler.

It is important to note that the data files created by these languages are all generated by the same MPE file system. Thus 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. This file-sharing characteristic also carries down to many HP 3000 subsystems such as KSAM and IMAGE.

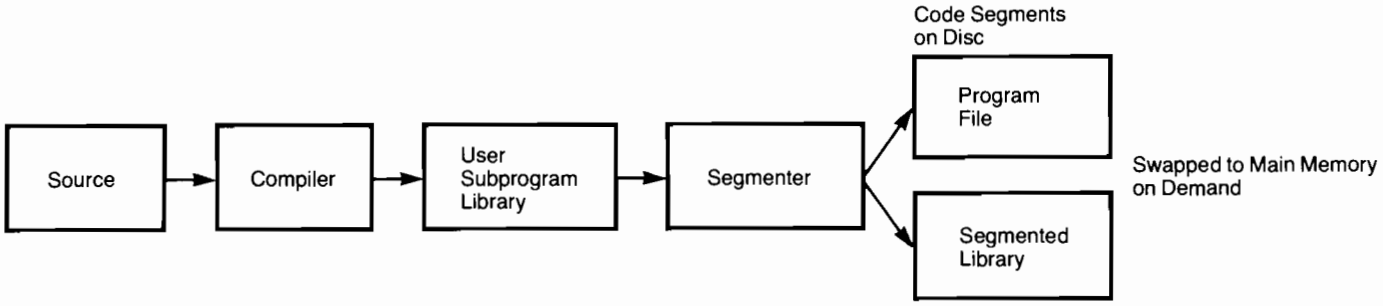


Figure 2-4 Code segment evolution

Segmenter

Program preparation is actually performed by the MPE segmenter in response to a PREP command or one of the combining forms which include PREP or GO. Segmenter commands may be used to manage USL files by adding, deleting, activating, or deactivating 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 other systems the program would have to be recompiled. With MPE, however, the segmentation can be modified by using the segmenter to rearrange RBMs and then preparing the USL file into a new program file.

Another feature of the segmenter allows different versions of the same subprogram to be stored within a single USL file. An optional index capability of the segmenter lets you activate and deactivate entry points within various versions of the subprograms.

Procedure Libraries

When a program is allocated and scheduled for execution, MPE searches the following libraries for unresolved external references:

- The user's log-on group library
- The public group library of the user's log-on account
- The public group library of the system account.

Each library can possess two types of library files: segmented library files, and relocatable library files. Segmented library or SL files contain procedures in segmented form which may be shared between programs. Relocatable library or RL files contain procedures in RBM form which must be prepared before they can be loaded with your program.

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 predefined 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.

The combination of segmented libraries and relocatable libraries gives great flexibility for storing often used subroutines and 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 relocatable or segmented libraries at the group, account, or system level.

Special segmenter commands are provided by MPE which enable you to build an SL or RL within a particular group. In addition, commands are available to add or purge routines, and list the procedures contained in either library.

Code Segmentation

To fully appreciate the programming assistance provided by the segmenter, it is necessary to understand the logic behind program code segmentation. In the HP 3000 computer system, program code is grouped into topical entities which consist of one or more procedures or subroutines. Each code segment may be up to 32 Kbytes in length (where K = 1024). Programs may be broken into multiple segments with procedures or subroutines fully contained within one segment.

A code segment consists entirely of information that is not subject to change during program execution. This includes program instructions and constants. No modifiable data may be interspersed with the instructions in a code segment, and it is only possible to change a code segment by recompiling the modified source. This feature ensures that all 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 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 segments contained in a system segmented library shared by all programs which request those services. Similarly, all users executing the same program file share the program's segments. Only one copy of a segment needs to be in main memory, no matter how many users may be executing it concurrently. Thus the system is able to handle more users in a given memory capacity.

Another advantage of re-entrant code segments 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. Code is swapped only into main memory, never out. The resulting reduction in swap traffic leads to more efficient memory management.

File System

One of the main uses of a computer system is information management, i.e., the input, processing, and output of data. MPE manages information by means of its file system. With MPE a file is a body of information or data identified by a user-assigned name. A file may contain commands and/or programs, as well as information, and may be stored on disc, tape, or cards.

MPE also treats peripheral devices as files. Access to such files is device-independent, meaning that a program can read data from a card reader, terminal, magnetic tape, or disc by means of the same request. MPE automatically locates, buffers, transfers, and deblocks the data.

When you ask to read a named disc file, you are only implicitly specifying the disc address of the file; the MPE system determines the explicit address and performs the read. In the same manner, when you ask for a certain type of device by specifying a device class name (disc, line printer, etc.), the file system allocates the actual device for you.

The MPE file system permits sequential access to all files. Disc files with fixed- or user-defined length records may also be accessed randomly. Extensive disc file backup facilities are provided for all types of disc files. The STORE command copies files to a serial storage device; the RESTORE command restores the files to disc.

Files can be accessed from any programming language by means of standard MPE file system intrinsics. A file can be accessed simultaneously by multiple programs; MPE automatically resolves any contention problems which might occur.

MPE file system commands enable programs to reference files without specifying their actual names, addresses, or characteristics. A file can be redefined without a major change to the program. For example, a program's input file named CARDIN designated as a card reader could be changed to a disc file through the use of a FILE command. File specifications can also be altered at run time by means of commands. To illustrate, a program could be coded to open a file with a record length of 128 characters. If at run time it is determined that the file has only 64 characters, you can override the file opening with a FILE command that designates the file to be 64 characters. Reprogramming is not required.

MPE file system commands can be used to build, purge, rename, and display file characteristics. You may specify how disc space is to be dynamically acquired, whether to deal with logical or physical records, whether to include special characters, and so forth.

The user logging facility of MPE allows users and subsystems to record additions and modifications to files. In the event of a loss of a file, the user logging record can be used to recover the data in the file. In addition, the logging file can be used as a record of the activity in that file. The entire user logging facility is implemented through the use of MPE commands and intrinsics designed for this purpose.

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

Subroutine Compatibility

MPE allows programs written in one language to call subroutines written in another language. Once the subroutine is written in the chosen language, it is stored in the system SL (segmented library). Then, user programs written in FORTRAN, BASIC, COBOL, PASCAL, or SPL can access this subroutine with a standard subroutine call. RPG programs can also execute these SL subroutines via an SPL subroutine call.

This ability to intermingle different programming languages significantly expands programmer productivity and application efficiency. Your programmers are free to program in the languages that they are most familiar with. The different language blocks can then be linked by a master program for execution. This way, segments of programs can be written in the language that is most efficient for the operation being performed. This MPE ability allows you to expand the efficiency and capabilities of your application programs.

File Security

MPE provides two general methods of file security. The first is the use of passwords. The creator of a file can establish passwords (also referred to as lockwords) which must be correctly supplied when 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 2-2.

ne system manager specifies the file access mod es al- owed for an account and the types of users to wh om ney are available. The account manager specifies the ac- ess modes allowed for a group and the types of users to vhom they are available. Finally, the creator of a file pecifies the file access modes allowed for the file and he types of users to whom the file is available.

n this manner, access to files can be controlled at sev- ral levels which range from unrestricted access (mak- ng the file available to anyone), to controlled access making the file available to its creator only). For exam- le, you can make your data file available to any c ther ser in a "read-only" mode, while only members of our account can append data to the file.

Often a need exists to save general purpose utility pro- rams in public groups or accounts which may then be ccessed by all system or account users. MPE provides a pecial system account named "SYS", and a publ c roup named "PUB" which exists under any acc c unt vith less-restrictive default security provisions.

ntrinsics

A multitude of additional system functions are available n the form of special MPE procedures, called intrinsics, vhich may be invoked by calls from your program. In- rinsic calls are acted upon when the segmenter pre- ares the program containing the intrinsic calls fc r xecution. The segmenter establishes a link between the xecuting program and the MPE procedure specified by he intrinsic call.

ystem intrinsics are written in the HP 3000 Systems rogramming Language (SPL) and follow the rules and onstraints of that language. They may be called from COBOL, BASIC, FORTRAN, PASCAL, or SPL pro- rams, and from RPG programs by way of an SPL ubroutine.

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
- User logging
- Handling software interrupts

Table 2-2 MPE file access modes and user types

Access Modes	
Reading	Allows the user to read files.
Appending	Allows the user to add information and disc extents to the existing files.
Writing	Allows the user to delete or change information already present in existing files
Executing	Allows the user to run programs stored in existing files.
Locking	Provides a logical lock which gives the user exclusive access to a file if desired.
Save Files	Allows the user to save permanent disc files within the user's 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.

Appendix C 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 re-

moved from the system; an interactive session resumes with a message and a prompt for another command. The CONTINUE command can 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 action to be taken in the event of an irrecoverable error.

A Dynamic Environment

The Multiprogramming Executive environment is a dynamic one where programs are run on the basis of processes. A process is the basic executable entity of MPE. It is not a program, but the unique execution of a program by a particular user at a particular time. MPE automatically creates, manages, and deletes processes.

When you execute a program, 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. To illustrate, when multiple users access the BASIC interpreter, a separate process is created for each of them. They all use the same code, there is only one BASIC interpreter; but each has his own data stack (environment) created by MPE.

The creation, maintenance and deletion of processes is accomplished by means of three MPE components: the virtual memory manager, the job/session scheduler, and the process dispatcher.

Virtual Memory

MPE's virtual memory manager uses both main memory and disc storage to greatly expand the total amount of memory space available. In fact, virtual memory allows programs up to 2 Megabytes in length to be executed in minimal memory systems. MPE logically divides programs into variable length segments of code and data. These segments reside in disc memory and are brought into main memory only when required for program execution, as shown in Figure 2-5. When a code segment is no longer needed, it is overwritten by a new segment. If the code segment is needed again later, it is simply copied again from the disc on which it resides.

Data segments are dynamic and are handled somewhat differently. Since the content of a data segment may change during program execution, a data segment is copied automatically back to the system disc when it is no longer needed, thereby replacing the previous version of that segment.

This approach of segmenting code and data and transfering the segments back and forth between main memory and disc memory, results in the allocation of local storage only as needed. In addition, since program code segments are not modified during execution, multiple users are able to share a single copy of the program code.

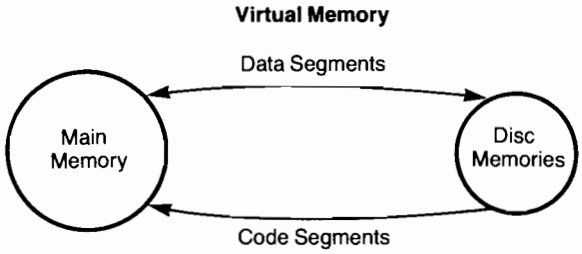


Figure 2-5 MPE virtual memory

A closer look at memory management identifies additional advantages. MPE divides main memory into two areas. The first, fixed memory, contains only those items required to be memory resident such as 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 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 manager 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. An attempt by an executing process to access code or data not present in main memory causes the memory manager to allocate main memory space for the missing segment. Other processes may execute while the missing segment is being brought into memory. When the absent segment is swapped from the disc memory to the main memory, the executing process is again eligible for execution. Frequently used segments remain in main memory, and may never be swapped, while rarely used segments are in disc memory most of the time. This results in high 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.

Automatic Scheduling

The MPE job/session scheduler schedules jobs and sessions according to their assigned priorities. When the execution of one process is interrupted for any reason, such as I/O, an internal interrupt, or an interrupt from the Scheduler itself, control is passed to the process with the next highest priority which is awaiting CPU resources. When two or more programs have the same priority, the oldest process is selected first.

obs and sessions are scheduled by means of a master ueue which is ordered by priority as shown in Figure 6. This master queue is divided into areas called prior- y classes. Each area is bounded by two priority num- ers established by the system manager.

IPE automatically assigns priority classes to each pro- ess executing on the system. The user may, however, pecify priority classes by selecting a general category of rocess dispatching priority for the program. This is one by including the PRI = parameter in your JOB or ELLO command. The five process dispatching priority ypes (queues) available are:

- AS—system processing only
- BS—very high priority
- CS—interactive
- DS—batch
- ES—very low priority (background)

IPE actually translates priority types into numerical nges which are ordered in a master queue (see Figure 6). The numerical range of each priority type can be nanged at any time to ensure that an optimal balance of ervices is maintained among the processes on the ystem.

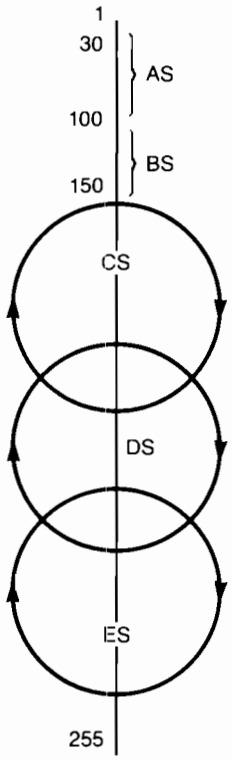


Figure 2-6 MPE master queue structure

Process Execution

The MPE process dispatcher 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. The dispatcher attempts to execute the highest priority process first. If that process is not in memory, the dispatcher instructs the memory manager to make enough of the process's segments present in memory to allow it to run. The next highest priority process is then chosen for execution while the segments are coming into memory.

As a process runs it may require another code or data segment. If the segment is not present in main memory, the memory manager is instructed to retrieve the segment before the process is allowed to continue executing. While the process waits for the needed segment to be transferred, MPE transfers control to the next process ready to be executed.

In order to bring a process segment into memory, the memory manager first looks to see if there is an available area of memory big enough for the segment. If so, the best fit available region is reserved for the segment and the I/O system will handle the swap. Otherwise, space is made by scanning main memory and making relatively unused portions available. This is done until a space big enough for the segment is found and reserved.

The objective of the process dispatcher and the memory manager is to provide for optimum efficiency in the use of system resources while satisfying the requirements of executing processes. This is done automatically by MPE without assistance from the system users.

System Operation

This section deals with the day-to-day operational aspects of running an HP 3000 installation. For all of its power and features, MPE is surprisingly easy to maintain. General operation of the system is primarily the responsibility of three users: the system manager, system supervisor, and the console operator. The console operator is responsible for routine day-to-day system operations. It is his task to respond to system messages and to keep the system and the peripheral devices functioning smoothly and efficiently. The system supervisor, who is appointed by the system manager, has day-to-day responsibilities in several areas. Included in his duties are the maintenance of the system logging and resource accounting facilities. He also has the capability to retrieve information and change parameters relating to the master scheduling queue. The system manager implements MPE's unique account/group/user organization and appoints account managers to monitor account usage.

System configuration, the first operation undertaken with a new HP 3000, establishes the boundaries under which MPE will function in terms of memory and peripheral devices. An interactive process that takes place with the operator at the system console, system configuration requires intervention only when options other than the system defaults provided are specified, and can be accomplished in a matter of minutes. Once configured, the system is ready for use.

The next operation involves the building of various accounts and the identification of system users. Usually, the first user identified is the system manager who will have overall control of the system. Among his responsibilities is the task of allocating various accounts within the system and identifying an account manager for each. The account managers in turn identify the users who may access the system through their respective accounts.

Once the system is configured and the account and user structure has been defined, the operation of program development begins. Programming may be done interactively on the HP 3000 by means of terminals and the many programming aids provided by MPE for the development of source programs; or programs can be developed in batch processing mode. Once developed, programs are executed as frequently as required.

Occasionally the account manager may need to add new users to their accounts, or it may be necessary to reconfigure the system as more memory or new peripherals are added. But the primary continuing operations are the development of programs and their execution. Figure 2-7 illustrates the interrelation of these general HP 3000 system operations.

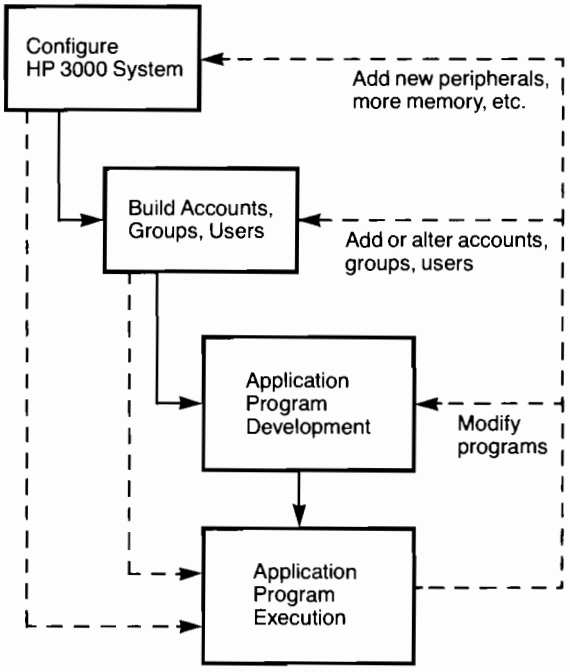


Figure 2-7 HP 3000 operations

Console Operator Function

The console operator function is available to any one user logged onto a non-REMOTE interactive terminal o the system. Operator commands are entered in SESSION mode just as are MPE commands. The console operator has the ability to move the console to other terminal devices, and may selectively allow users to exe cute specific operator commands. The operator can also allow individual users to have operator control over spe cific devices. Operator commands deal primarily with peripheral and other hardware system requirements, a well as with system start-up, back-up and recovery procedures.

Start-up and Modification

MPE is initially brought up on an HP 3000 by the operator at the system console, through one of several restar operations:

- WARMSTART—The system is restarted from the dis Spoolfiles are recovered, and incompletely processec spooled jobs are automatically restarted.
- COOLSTART—The system is restarted from the disc The spoolfiles in existence when the system was shu down are not saved, but all resident user files are saved.
- COLDSTART—The system is read from a serial storage device (magnetic tape or serial disc). The I/O con figuration and system configuration from the serial storage device used to define the system are merged with the directory and files present on the disc.

- UPDATE—Similar to cold start, except that the I/O configuration on the disc is used to define the system. This start-up mode is used to define the system. This start-up mode is used when starting the system from an updated version of MPE supplied by 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.

System Backup and Recovery
Periodically, the MPE system and user files are copied on a serial storage device for back-up purposes. The serial storage device is then available for reloading the system in the event of a hardware or software failure, or to transfer the system to another hardware installation.

In the latter case, a new I/O and system table configuration may be specified during an interactive dialogue. Areas which may be changed include:

- I/O devices
- System table sizes
- System disc allocation
- Logging facility
- Scheduling changes
- Segment size limits
- System modules to be allocated or made memory resident

The SYSDUMP command (which requires System Supervisor capability) may be used to create media for reloading on a different or changed HP 3000. A new I/O configuration and system table configuration may be specified during the interactive dialogue which occurs. A back-up date may also be specified which enables the operator to back-up only those disc files which change each day.

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 in use when the failure occurred, or if the power outage lasts longer than the built-in system memory battery backup).

In addition to the system configuration aids mentioned above, MPE provides a range of software aids specifically designed for use by the system operator. Among these are:

- Spooling facility
- Tape label facility
- Disc options.

Spooling Facility
This MPE facility permits the 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 disc where it is processed later. This procedure is called spooling. (SPOOL is an acronym for Simultaneous Peripheral Operations On-line.)

To illustrate, if six users need to produce output on a line printer at approximately the same time, their output is directed to spoolfiles on disc from which the output 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 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.

The spooling of batch job input can be initiated not only from a card reader or magnetic tape unit, but also from within an interactive session as described earlier.

Tape Labeling Facility
MPE provides a tape labeling facility for use in reading and writing labels on magnetic tapes. The facility can be used to:

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

The facility can be used to read, but not write, IBM-standard tape labels; read and write ANSI-standard labels; and read and write user-defined labels on previously labeled magnetic tapes.

Disc Options
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 system 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.

MPE also provides a private disc volume facility which allows you to create and access files on removable disc volumes. Private volumes consist of removable disc packs which, when mounted on a disc drive, can be accessed by MPE through the file system. 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.

System Account Structure

The primary responsibility of the system manager is to create and maintain the organizational structure of accounts, groups, and users under which access to MPE occurs. The account structure provides maximum security and control by permitting the system manager to assign specific access and system usage capabilities to each user.

"Accounts" are collections of users and groups. Each account has a unique name and an optional password assigned to it when the system manager creates the account. Each account also has its own file domain or unique set of files. The system manager may define resource-use limits for an account. MPE maintains a running count of each resource that the account uses. MPE also stores a list of user name and group names recognized by the account, the maximum job priority at which jobs in the account may be scheduled, and limits established on the account's usage of disc file space, CPU time, and connect time.

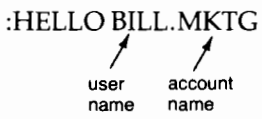
"Groups" are used to partition the file domain of an account. Files must be assigned to a group, and each group has a unique name (within the account) and optional password. Limits may be established on the permanent disc space, CPU time, and connect time used by a group. MPE maintains running counts of resource usage for each group and the sum of these group counts always equals that of the account in total.

"Users" are individuals who access the HP 3000. Each user is assigned a unique name and optional password, and is assigned to a specific account. Each user may have a specified home group of files, and may access any other file groups in the account. A maximum job priority may be assigned to each user.

Each account "owns" a unique set of files separate and distinct from every other account. This ownership is indirect in that only groups may own files directly. Thus, every file belongs to a group, every group belongs to an account, and every account belongs to the system.

To illustrate how accounts, groups, and users interrelate, consider the following example. Figure 2-8 represents a system which includes interactive terminals dispersed throughout a company. The system manager has assigned three accounts: Marketing, Engineering, and Finance. The marketing account manager has defined two users who can access the system: Bill and Dave. Each user has his private group (assigned as his home group) where he stores his private programs and files. Bill and Dave can also access programs and files stored in the other groups in the account. A group named PROJ1 was created to contain programs and data files related to current projects. An administrative group was also created to contain administrative work such as schedules and budgets. The public group, to which no password was assigned, contains general purpose utility programs for use by all.

Bill can log on to the HP 3000 from a terminal with the command:



By default, Bill now has access to all programs and data files in his home group: BILLSGRP. Bill can gain access to a file in the PROJ1 group by using the fully qualified file name which specifies both the account and group under which the file was created. The file Bill wants is data file 1 so he enters:

Alternatively, Bill could have logged on to the HP 3000 and requested access to all programs and files in the PROJ1 group by appending the group name to his log on request, as follows:

Bill now has access to all files in the PROJ1 group.

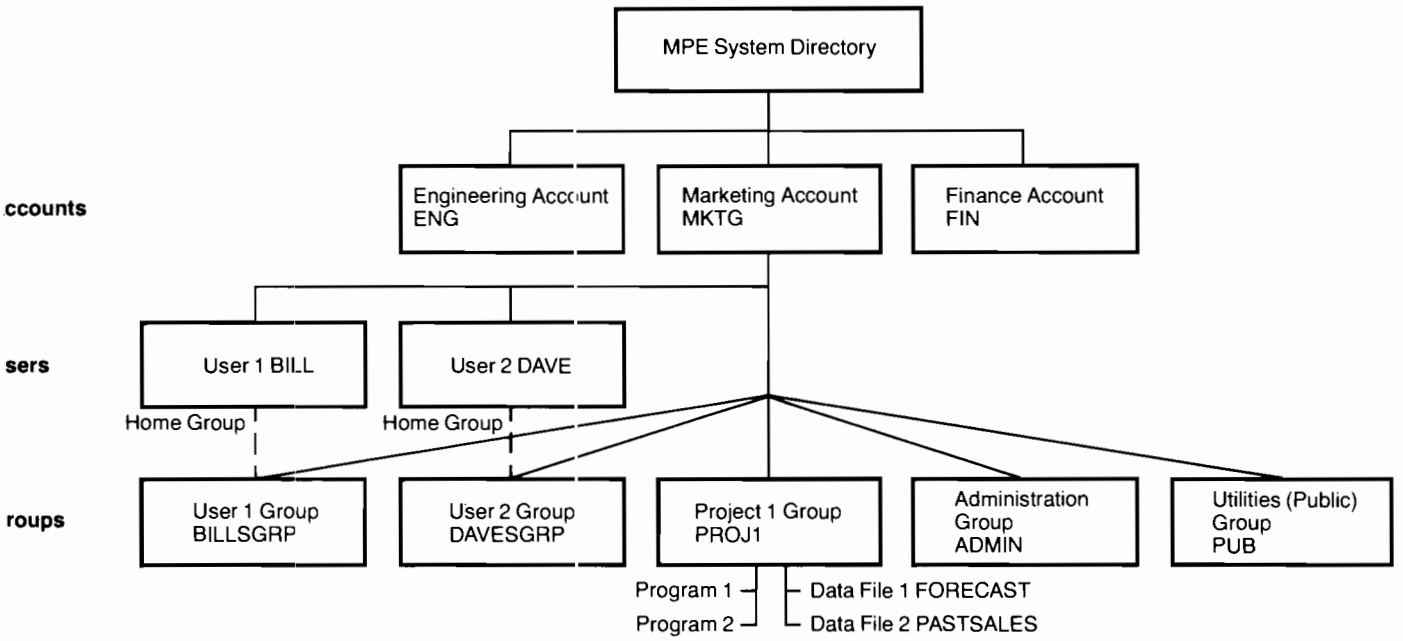


Figure 2-8 MPE account structure

o summarize, you can log on the system using only our assigned user and account names, in which case ou are automatically given access to your home group.)r, you can log on specifying your user name, account ame, and a group name which gives you access to the roup you specify whether or not it is your home group.

o access a file in a group other than the one specified /hen you logged on, you use the fully qualified name of ne file which can consist of the file name, the group name, the account name, and the appropriate passwords when required.

As you can see, the account structure provides both control and security over file use. Access to the system is granted only to individuals with a valid log-on identification consisting of account, group, and user names, each of which may require a password. Figure 2-9 illustrates both an unsuccessful and successful log-on procedure where passwords are required.

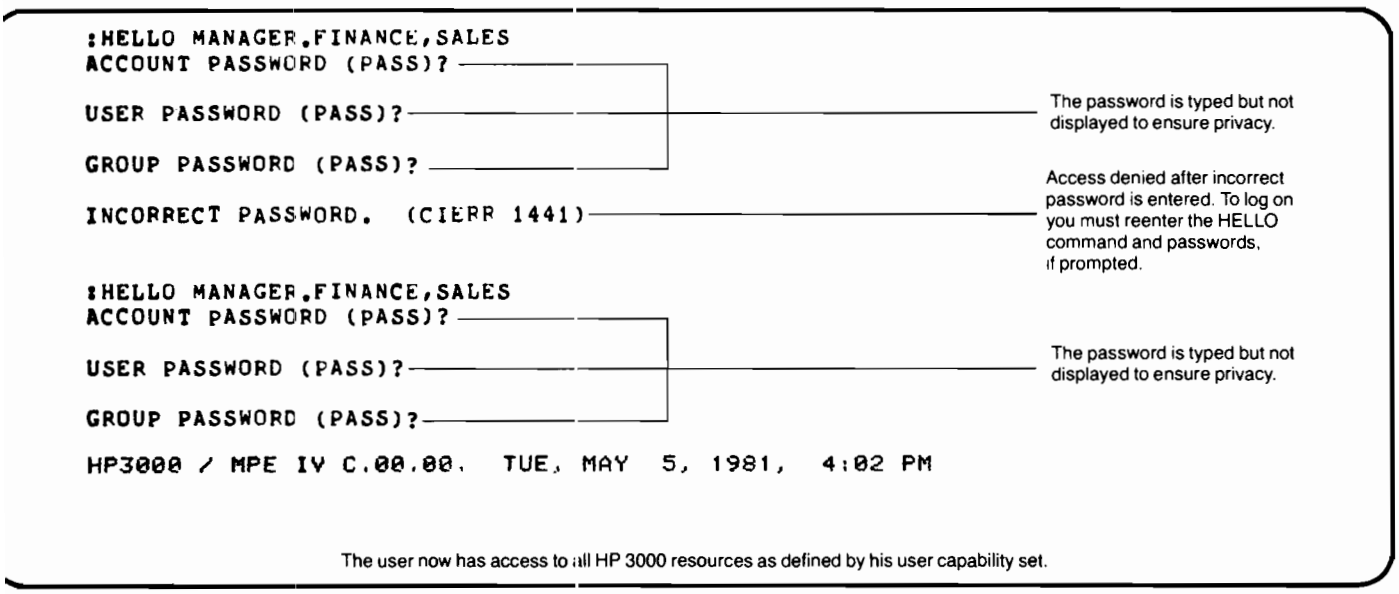


Figure 2-9 MPE system access security.

System Supervisor

The user with system supervisor capabilities can use commands which enable him to control the master scheduling queue, permanently allocate or deallocate a procedure or program to virtual memory, and manage the logging facility.

He can control the master scheduling queue in two ways: first by entering the SHOWQ command which displays information about the scheduling of processes and the contents of the master queue and various subqueues; and second, by entering the TUNE command which allows changes to the time quantum or limits defining the bounds of any subqueue.

Two important commands, ALLOCATE and DEALLOCATE, allow him to tune the system by permanently allocating a procedure or program in the HP 3000 to virtual memory. This significantly decreases the time needed for an operation with large, frequently used routines.

System Logging Facility
The MPE logging facility records details of system resource requests in a series of log files on disc and can be used to monitor system resource usage. The system supervisor selects those system and user events that are to be recorded. Log records are provided for job and session initiation or termination, program termination, file closing, file spooling completion, and system shutdown. Log files can be used in the generation of precise billings based on accurate system usage records.

I/O device failures on any device are also recorded in log files which can then be used to detect problems before they begin to interfere significantly with overall system operation.

Accounting Facility
The MPE accounting facility provides a flexible and powerful means of coordinating access to the system and disc file usage. To coordinate system access, system administrators can devise a structure of accounts and users which reflects the functional organization of the people who use the system. The accounting facility maintains running totals on the amounts of system resources that each account consumes, including disc space used, cumulative CPU time consumed, and cumulative terminal connect time for sessions. The current totals can be displayed at any time and can be used for billing purposes.

File usage can be coordinated also because 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.

Users may create files only in the account and group under which they are currently running. By using fully-qualified file names users also have access to any file present in the system, provided that existing security provisions allow them access.

Interprocess Communication Facility
The MPE interprocess communication facility allows independent executing processes to pass information back and forth efficiently via disc files. The facility uses standard file system intrinsics as well as additional intrinsics to handle software interrupts. The facility provides security for the passed data via the MPE security system.

Performance Measurement

On-Line Performance Tool/3000 (OPT/3000)

The On-line Performance Tool/3000 (OPT/3000) is a performance measurement package which consists of two interrelated products: OPT/3000 Performance Measurement Software and the OPT/3000 System Performance Training Course. This Package is designed to help HP users gather and utilize performance data for system management, capacity planning, and application development activities.

On-Line Performance Software

On-line Performance Tool/3000 is an interactive performance measurement software product for the HP 3000 that provides information to the system analyst. OPT/3000 can be used to characterize the current system workload, CPU utilization, memory management activity, I/O traffic, program and process activity, and system table usage to help the user isolate bottlenecks and improve system performance. Performance data provided by OPT/3000 is continuously updated at regular intervals and can be presented in charts, graphic displays, or summary reports. Information can be displayed on an H terminal and a hard copy of any display can be generated on a line printer with a single keystroke. Although OPT/3000 is primarily designed for interactive use, it can be executed in batch mode to collect snapshots of system activity over a period of time.

Display Contexts
OPT/3000 can generate 23 unique displays containing system performance information. These displays are grouped into six categories called display contexts. Each context is associated with a different type of system resource.

These six display contexts are:

- Global
- Memory
- CPU-Memory Manager
- I/O
- Process
- System Tables

Within each context, displays are available at successively greater levels of detail. This structure allows the user to progress from summary level information to more detailed information as required.

Features

- Interactive Terminal Reporting of Performance-Related Data: OPT/3000 provides summary and detailed CPU, memory, I/O, and process information in dynamically updated terminal displays. The displays are updated under the control of the user and may be updated automatically or by simply pressing the return key.
- Graphical Presentation of Information: OPT/3000 utilizes the features of the HP 26xx series of terminals to generate displays with a graphical format. The terminal video enhancements used include blinking, inverse video, underlining, and half-bright. The line drawing character set and the cursor addressing capabilities are also used.
- Multiple Display Levels: The displays associated with a particular context are layered such that each level provides the user with more detailed information than the previous level.
- Summary Reporting of Resource Usage: Although OPT/3000 is primarily designed for interactive use, it can be executed in batch mode to collect summary information about system activity. These reports can also be generated interactively and can be used to provide data for capacity planning activities.
- Hard Copy Capability: A hard copy of any display can be generated on the line printer with a single keystroke.
- Logging Capability: The information used to generate the summary reports can also be logged into a disc file. This data can be accessed by user programs for reformatting and reporting at a later time.
- Low Overhead: The performance impact of OPT/3000 running on an HP 3000 is low, varying from one to three percent of available CPU time. As a result, multiple users of OPT/3000 on a system can be executing simultaneously with little impact on system performance.
- Extensive On-Line Help Facility: With this integrated facility, documentation explaining any command or context display can be quickly and easily displayed. In many cases, interpretation guidelines are provided to aid in the identification of performance problems.

System Performance Training Course

The OPT/3000 System Performance Training Course teaches users how to use OPT/3000 software to generate performance related data and interpret the results. The course shows system analysts how to identify performance bottlenecks, characterize workloads, collect information for capacity planning, analyze system table configurations, and tune the performance of individual programs. This training is required with every initial installation of OPT/3000.

Flexible Disccopy

Flexible Disccopy/3000

Flexible Disccopy/3000 provides a method of converting IBM 3741 format flexible disc data sets (files) to HP 3000 disc files and translating EBCDIC character code to ASCII. It can operate in either an interactive environment or in batch mode, and can convert either single or multiple data sets and volumes. (Each IBM 3741 flexible disc is called a volume.) A complete error, warning, and status message file is included, and with the MPE Operating System provides the user and console operator with messages about program status, user prompts, and error conditions.

System Architecture

Stack Architecture
Separation of Code and Data
Processes
Variable-Length Segmentation
Code Segmentation
Data Stack
Registers
Virtual Memory
Microprocessor
Microcode
Instructions

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 3-1. 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 the top two stack data values are implicitly used.

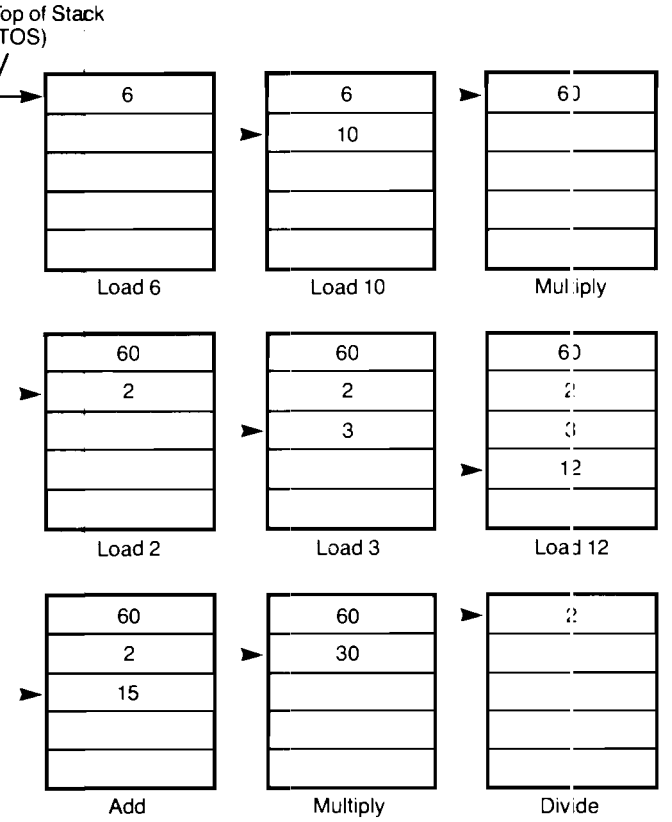


Figure 3-1 Stack Evaluation of (6*10) ÷ [2*(3+12)]

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 do not 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 principal 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 environment 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 intermixing 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 3-2.

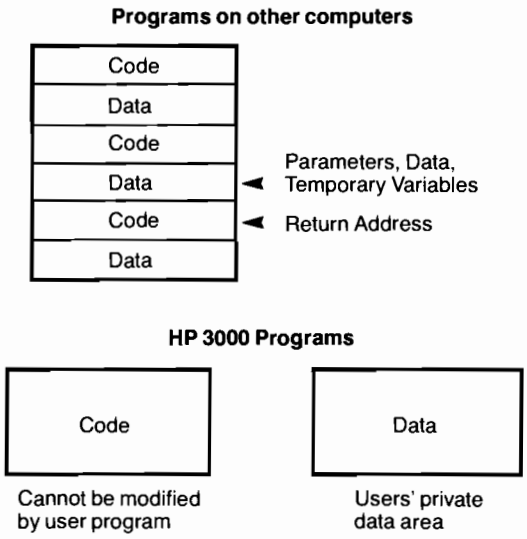


Figure 3-2 Separation of Code and Data

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 nonmodifiable 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-entrant code allows one copy of heavily used programs to be shared by many users concurrently, main memory can be used with optimum efficiency. Re-entrant code and stack-structured data together make possible subprogram recursion (a subprogram calling itself), a capability 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 t 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 t 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 b MPE in a code segment table while the location and size of all associated data segments is maintained by MPE ir 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 seg-

ents required for current execution are present in ain memory. Code segments may be up to 32,760 ytes in length. Data segments may be up to 65,528 ytes in length.

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

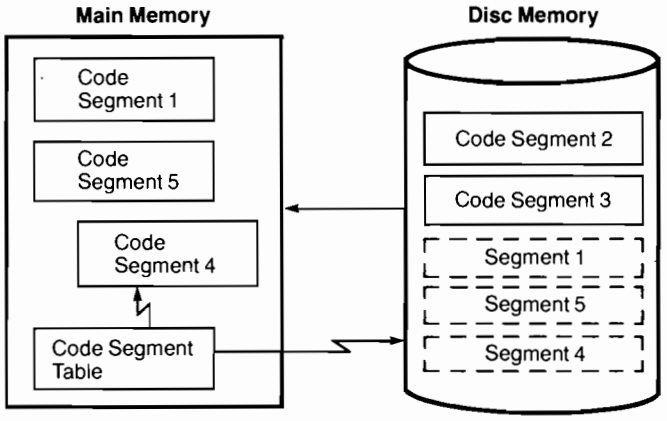


Figure 3-3 Code Segmentation

Code segmentation

ode segmentation allows you to divide your program to several segments using control statements to the mpiler at compilation time or commands to the MPE perating system. Then the operating system takes over e management of these segments. That is, MPE deter- ines whether or not a code segment should be in main emory or in disc memory. Thus you can write pro- ams much larger than the available main memory of ur HP 3000. In the example in Figure 3-3, a program as been divided into five code segments. Code seg- ents 1, 4 and 5 are in main memory while code seg- ents 2 and 3 are in disc memory. The code segment ble (CST), which is resident in main memory, points to e active code segments. If the code segment is in main emory, the code segment table points to the beginning that code segment. If the code segment is on disc, the de segment table points to its disc address. The man- gement of code segments and the code segment table completely transparent to your program. When a sub- utine call is made from one code segment to the next, e instruction that makes the call examines the code gment table to determine whether or not the new de segment is in main memory. If it is, control is trans- rred immediately to that code segment. If the code gment table indicates that the required segment is on sc, then the instruction interrupts the operating sys- m and informs it that a required code segment is not memory. It is then up to the operating system to ake space for that code segment in memory so execu- on of the process can continue.

u have control over segmentation; that is, you deter- ine where the program is divided. The result is that gmentation can be tailored to your program's logic. e size of segments can be optimized to memory size d you can avoid thrashing (unnecessary swapping of de segments due to poor segmentation—across a DO op, 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 can be cut back to make use of the same memory space for the next subroutine. The end result is that less memory is required for program execution.

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.

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 3-4 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 condition, 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 manipulat-

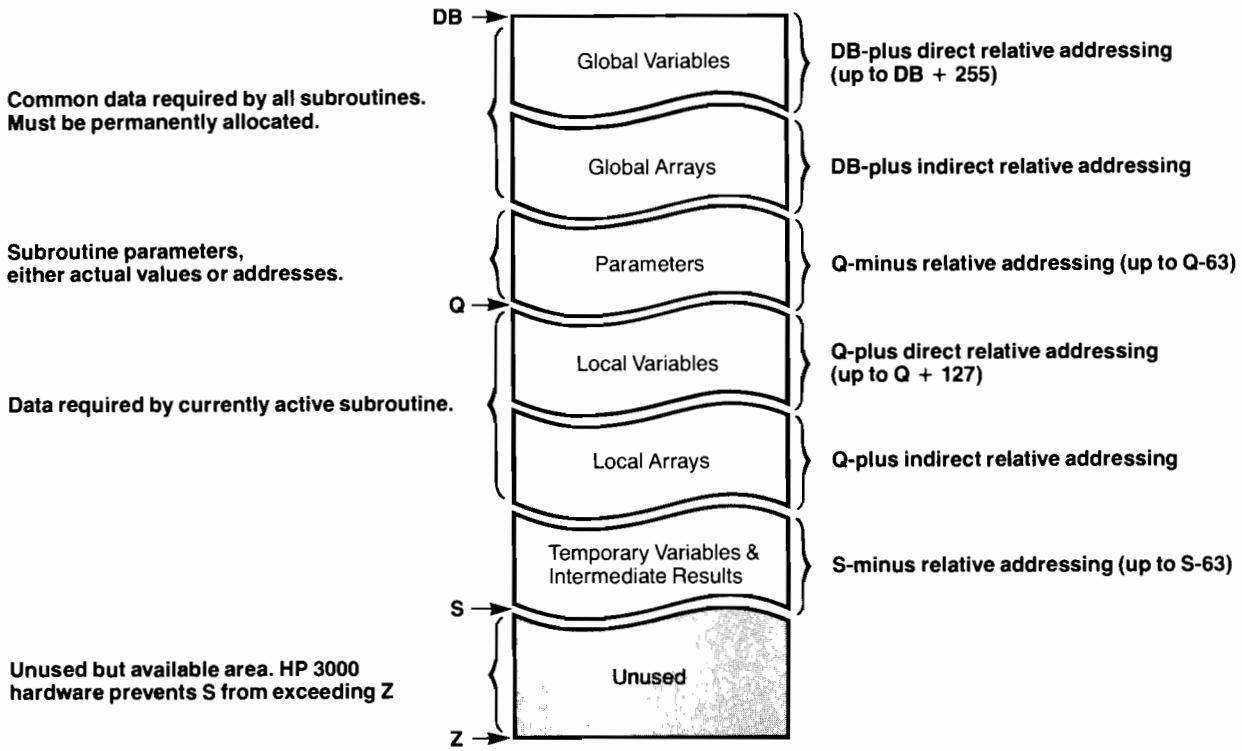


Figure 3-4 Data Stack Contents

g data at or near the top of the stack. These registers e implicitly accessed by many of the machine instructons and whenever the top stack locations are specifilly referenced. Data stacks are automatically xpanded by the operating system during execution up a maximum size of 64 Kbytes (K = 1024).

Registers

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

ll addressing of code and data is done relative to hardare address registers. Thus by simply changing the ase addresses in these registers, segments are dynamilly relocatable in memory. The few instances where bsolute addresses are required are privileged operaons handled by the operating system.

pproximately one-half of the HP 3000 registers are accessible to user programs and/or the operating system nd its related software. The remaining registers are sed, for example, by the interrupt system and for miroprogram processing. The registers for the Series 64 re summarized in Appendix D, the registers for the Seies 44 are summarized in Appendix F, and the registers or Series 40 are summarized in Appendix G of this Genral Information Manual.

Virtual memory

irtual memory is a very efficient memory management cheme which, in addition to main (semiconductor) iemory, uses disc storage as secondary memory. Users' rogram code and data are divided into variable-length egments which reside in secondary memory. As a proram is being executed, only those segments of code nd data which are required at a particular time actually eside in main memory; the other related segments remain on disc until they in turn are required. When a articular code segment is no longer needed, it is simply verlayed by another segment.

his can be done because in the HP 3000, code segments re nonmodifiable and re-entrant. When the segment is eeded again, it is simply copied from the disc on which ne program resides. Since programs are copied into ain memory directly from disc memory, they need not e copied prior to execution to a special "swapping isc." Data segments, however, are dynamic, and their ontents can change during execution. Therefore, when particular segment is no longer needed, it is automatially 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.

Microcoded Processor Architecture

The HP 3000 Central Processing Unit is microcoded. This means the HP 3000 instruction set is broken down further into a series of microinstructions a group of which represents each macroinstruction. This microcode is stored in the Control Store of each CPU. On the Series 40 and 44 this Control Store is implemented in permanent Read Only Memory (ROM) which is part of the CPU hardware. On the Series 64 the Control Store is implemented as Writeable Control Store (WCS) and is loaded from the Cold Load Storage device. Control Store never changes during the operation of the machine.

Microcode

On the Series 40 and 44 the microcode is organized as a series of 32-bit microinstructions. On the Series 64 the microcode width is 64-bits because it controls two closely-coupled processors.

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.

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.

Instructions

There are over 200 unique HP 3000 instructions. 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. Complete machine instructions are provided in Appendix E for the Series 64 and Appendix H for the Series 40 and 44.

Appendix A:
System Documentation

Overview

Guide to a Successful Installation 30000-90135	General Information Manual 30000-90008	HP 3000 Specification Guide 5953-0588

HP 3000 Hardware

Site Preparation Set 30000-60029	Site Planning and Preparation Guide 30000-90206	Site Planning Workbook 30000-90207	HP Computer Supplies 5953-2450

Machine Instruction Set 30000-90022	Instruction Decoding Pocket Guide 30000-90057

System Usage

MPE Commands 30000-90009	MPE Intrinsics 30000-90010	Segmenter 30000-90011	System Utilities 30000-90044	Debug/Stack Dump 30000-90012
EDIT/3000 03000-90012	FCOPY/3000 03000-90064	Error Messages 30000-90015	Index to MPE 30000-90045	Software Pocket Guide 30000-90049
Using files 30000-90102	System Tables Manual 32002-90003	SORT/3000 32214-90001		

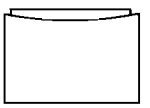
System Administration

System Manager/ System Supervisor 30000-90014	HP 3000 Console Operator's Guide 32002-90004

Additional System Software

OPT/3000 32238-90001	Flexible Disccopy/3000 32199-90001

Denotes Pocket Guide



Appendix B:
MPE Commands

All MPE 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**
:ALTLOG	Alters the attributes of an existing logging identifier.	**J,S,B,P**
:ALTSEC	Changes security provisions for a file	**J,S,B,P**
:APL	Accesses the APL subsystem.	**J,S**
:ASSOCIATE	Gives a user operator control of a device	**J,S,B,P**
: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**
:DISASSOCIATE	Removes the control of a device from a user	**J,S,B,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 DSN/DS.	**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**
:FCOPY	Invokes the FCOPY facility.	**J,S**
: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**

Command	Function	When issued
:FREERIN	Deallocates a global RIN, and returns it to RIN pool.	J,S
:GETLOG	Establishes a logging identifier on the system	J,S,B,P
:GETRIN	Acquires a global RIN.	J,S,B,P
:HELLO	Initiates a session. Acceptable for device recognition. Requires 1A capability class.	S
:HELP	Access the HELP subsystem.	J,S,B
:IF	Used to control the execution sequence of a job.	J,S,B
:IML	Initiate DSN/IMF Pass-Through Mode	
:JOB	Initiates a batch job. Requires BA capability class.	J,S
:LISTF	Lists descriptions of files	J,S,B,P
:LISTLOG	Lists active logging identifiers	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 DSN/Multi-leaving Remote Job Entry (MRJE) facility.	
: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
:RECALL	Displays all pending console REPLY messages.	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
:RELLOG	Removes a logging identifier from the system	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 DSN/RJE Subsystem.	J,S
:RPG	Compiles an RPG/3000 program.	J,S
: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

Command	Function	When issued
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
SHOWLOG-STATUS	Displays status information about currently opened log files.	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
:VINIT	Accesses the VINIT subsystem to perform on-line conditioning and formatting of serial discs and private volumes.	J,S
: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
-CLEANSL	Copies the currently managed SL to a new SL file, removing inactive segments	J,S
-CLEANUSL	Copies the currently managed USL to a new USL file, removing inactive segments	J,S
-COPY	Copies an RBM or segment from one USL to another.	J,S

Command	Function	When issued
-COPYSL	Same as CLEANSL except allows user to expand SL space by a given percentage	J,S
-COPYUSL	Same as CLEANUSL except allows user to expand USL space by a given percentage	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	List specific segments in the SL.	J,S
-LISTUSL	List specific segments in 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

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	Loads a program or procedure.	J,S
DEALLOCATE	Removes a program or procedure from virtual memory.	J,S,P
IMLMGR	Starts interactive IML Managers' program	J,S
JOBPRI	Sets or changes the priority for batch jobs or sessions.	J,S,B,P
RESTORE	Returns files to the system.	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 into 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
TUNE	Changes scheduling parameters for processes	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.
ALLOW	Grants a user access to a specific operator command
ALTJOB	Alters attributes of waiting job or session.
ALTSPOOLFILE	Alters attributes of output spooling files.
BREAKJOB	Suspends an executing job.

Console Operator Commands

Command	Function
:CONSOLE	Changes the system console from its current device to another job-accepting (non-DS) terminal
:DELETESPOOLFILE	Deletes a spooled device file.
:DISALLOW	Prohibits a user access to a specified operator command.
:DOWN	Removes a device from normal system use.
:DOWNLOAD	Downloads information to an output device.
=DSLINE	Enables or disables the data communications link under control of the DS/3000 subsystem.
: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.
:JOBSECURITY	Controls the availability of certain job commands to a user.
:LDISMOUNT	Logically dismounts a private volume set/class (UV capability required).
:LIMIT	Limits the number of concurrently running jobs/sessions.
:LMOUNT	Logically mounts a private volume/class on a non-system domain disc drive.
:LOG	Starts, restarts, stops User Logging (LG capability required).
=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.
=MPLINE	Enables or disables the data communications link under control of the DSN/MTS subsystem.
=MRJE	Enables or disables the data communications link under control of the DSN/MRJE subsystem.

Command	Function
:OUTFENCE	Defines priorities for output spooled files.
:REFUSE	Disallows jobs/sessions and/or data on a designated device.
:REPLY	Replies to a pending console request.
=REPLY	Same as :REPLY
:RESUMEJOB	Resumes a suspended job.
:RESUMESPOOL	Resumes a spooled device.
=SHUTDOWN	Closes down the operating system.
:STARTSPOOL	Initiates spooling of a device.
:STOPSPOOL	Terminates spooling of a device.
:STREAMS	Enables or disables the users' ability to submit job/session and/or data streams.
:SUSPENDSPOOL	Causes a spooled device to stop operation.
:TAKE	De-assigns a device that was GIVEn to the diagnostics.
:UP	Allows a DOWNed device to function again.
:VMOUNT	Enables or disables the private volumes facility.
:WARN	Sends an urgent message to jobs and sessions.
:WELCOME	Defines the message users receive when they log on the system.

Appendix C: MPE Intrinsics

ll MPE intrinsics (system procedures) are summarized elow, listed alphabetically. Any special capabilities required to call a particular intrinsic are noted.

trinsic	Function
CCEPT	Accepts (and completes) the request received by the preceding GET intrinsic call.
CTIVATE	Activates a process. (Requires PH capability).
DJUSTUSLF	Adjust directory space in a USL file.
LTDSEG	Changes the size of an extra data segment. (Requires DS capability.)
RITRAP	Enables or disables internal interrupt signals from all hardware arithmetic traps.
SCII	Converts a number from binary to ASCII code.
INARY	Converts a number from ASCII to binary code.
ALENDAR	Returns the calendar date.
AUSEBREAK	Requests a session break.
LOCK	Returns the time of day.
LOSELOG	Closes access to the logging facility.
OMMAND	Executes an MPE command programmatically.
REATE	Creates a process. (Requires PH capability.)
TRANSLATE	Converts a string of characters from EBCDIC to ASCII or from ASCII to EBCDIC.
ASCII	Converts a value from double-word binary to ASCII code.
BINARY	Converts a number from ASCII to double-word binary value.
EBUG	Sets breakpoints and modifies or displays stack or register contents.
LSIZE	Changes size of DL-DB area.
MOVIN	Copies block from data segment to stack. (Requires DS capability.)
MOVOUT	Copies block from stack to data segment. (Requires DS capability.)
XPANDUSLF	Changes length of a USL file.

Intrinsic	Function
FATHER	Requests PIN of father process. (Requires PH capability.)
FCARD	Drives the HP 7260A Optional Mark Reader.
FCHECK	Requests details about the file input/output errors.
FCLOSE	Closes a file.
FCONTROL	Performs various control operations on a file or terminal device.
FDEVICECONTROL	Performs various control operations on a spoolfile associated with a 2680 Laser Printer.
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.
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.
FREAD-BACKWARD	Reads a logical record on a tape drive from a point in front of the current record pointer.
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.

Intrinsic	Function
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.
FWRITELABLE	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. (Continue execution if I/O not completed.)

Intrinsic	Function
IOWAIT	Initiates completion operations for an I/O request. (Wait for I/O to complete.)
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.
PRINTOP	Prints a character string on operator's console.
PRINTOREPLY	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.

Intrinsic	Function
UTJCW	Puts the value of a particular job control word (JCW) in the JCW table.
WRITE	Sends a block of data to the remote slave program.
PENLOG	Provides access to a logging facility.
UIT	Aborts a process.
UITPROG	Aborts a program.
EAD	Reads an ASCII string from an input device.
EADX	Reads an ASCII string from an input device.
ECEIVEMAIL	Receives mail from another process. (Requires PH capability.)
EJECT	Rejects a request received by the preceding GET intrinsic call and returns an optional tag field back to a remote master program.
ESETCONTROL	Resets a terminal to accept a CONTROL-Y signal.
ESETDUMP	Disables the abort stack analysis facility.
EARCH	Searches an array for a specified entry or name.
ENDMAIL	Sends mail to another process. (Requires PH capability.)

Intrinsic	Function
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.
WRITELOG	Writes a record to a logging file.
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 64
Hardware Features

his appendix provides a summary of hardware fea-
res of the HP 3000 Series 64 computer system. For
ore details on the HP 3000 architecture, refer to Chap-
r 1; System Introduction, and Chapter 3; System
rchitecture.

he HP 3000 Series 64 is designed as a full function com-
uter, providing the processing power and capabilities
handle a full range of EDP and Distributed Data Pro-
ssing. Based on a modular concept, the HP 3000 Series
allows independent elements to be interconnected
rough a newly designed, powerful multiple I/O bus
ructure. These elements consist of a CPU module with
ual arithmetic logic units (ALUs), cache memory, main
emory, I/O Adaptors, General I/O Channels, and
SN/Advanced Terminal Processors. Communication
etween modules is accomplished using a high speed
entral System Bus and up to two Intermodule Busses.
he system also includes a system console, system dis-
ay panel, and a Diagnostic Control Unit (DCU). Pe-
pheral devices are connected to the system through
he General I/O Channels. Interactive terminals are at-
tached to the system through the DSN/Advanced Termi-
nal Processor or DSN/Intelligent Network Processors
running the DSN/Multipoint Terminal Software. Data
communication links are established via DSN/Intelligent
Network Processors.

The CPU is centered around a Hewlett-Packard de-
signed microcoded processor using high speed Emitter
Coupled Logic (ECL) technology and a dual arithmetic
logic unit (ALU). This implementation provides the
highest performance ever offered in an HP 3000 while
allowing flexibility in the machine instruction set. The
Series 64 also employs high speed, semiconductor, ran-
dom access main memory modules which use automatic
fault detection and correction.

The hardware design of the HP 3000 Series 64 provides
the ability to expand the system as the user's needs
grow and applications change. The system software
accommodates these changing needs by allowing
additional hardware modules and peripheral devices
to be easily configured on the system.

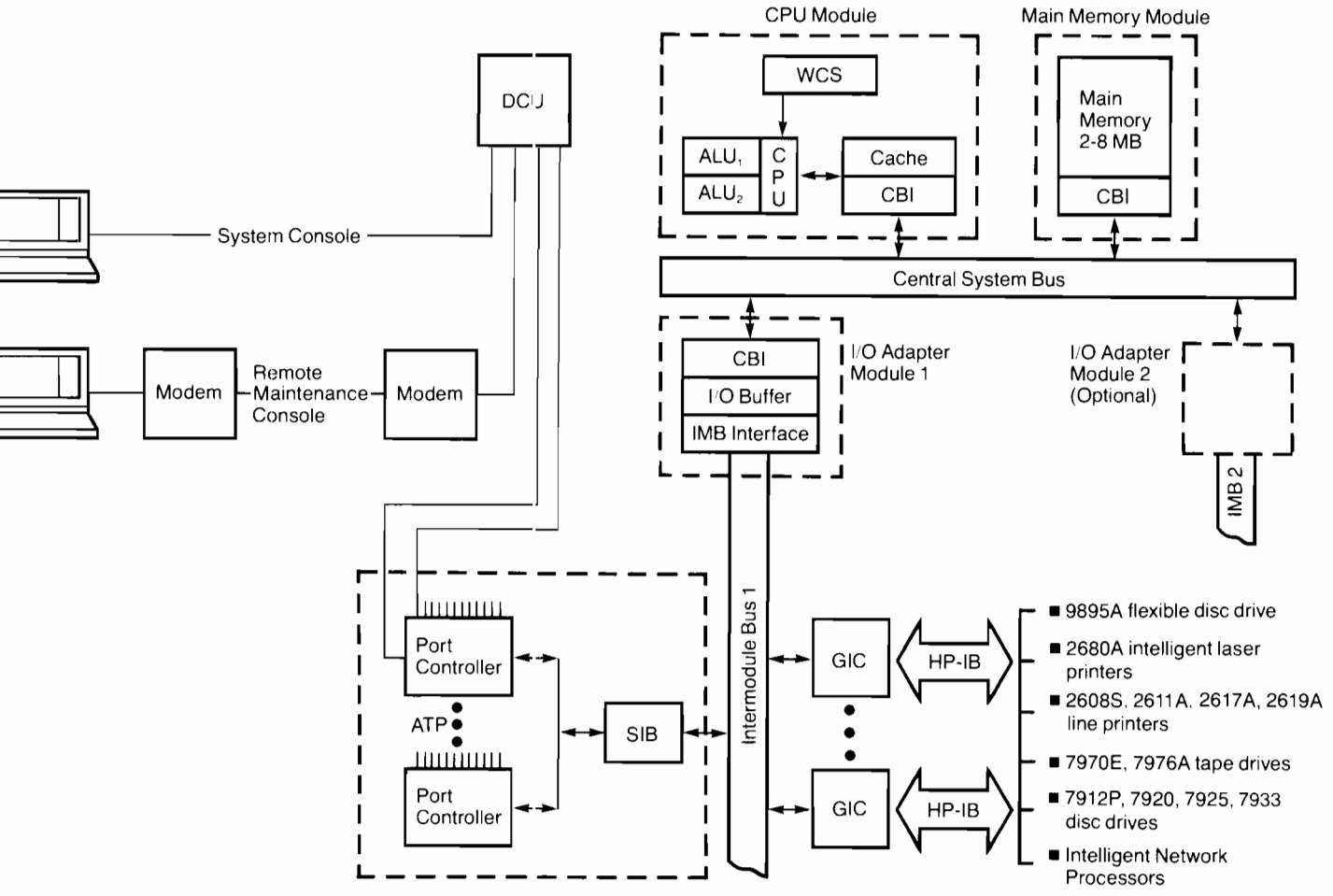


Figure D-1. HP 3000 Series 64 Hardware Organization

Central Processing Unit (CPU)

The significant features of the HP 3000 Series 64 central processing unit (CPU) are listed in Table D-1.

Table D-1. HP 3000 Series 64 CPU Features

Architecture

- Hardware-implemented stack
- Separation of code and data
- Variable-length code segmentation
- Non-modifiable re-entrant code
- Virtual memory for code
- Dynamically relocatable programs

Implementation

- Microprogrammed Emitter Coupled Logic (ECL) CPU
- 75 nanosecond microinstruction cycle time
- 8 Kbyte cache memory
- Dual 16-bit arithmetic logic units (ALUs)
- 40 Kbyte Writeable Control Store (WCS)
- Automatic restart after power failure
- Central System Bus
- Overlapping CPU and I/O operations

Instructions

- Over 214 powerful instructions
- Instructions are 8, 16, and 32 bits in length
- 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

The CPU converts an instruction in the current instruction register (CIR) into a starting address for the microcode contained in the Writeable Control Store (WCS) and determines various initial conditions required for executing the instruction. As the current instruction is being executed, the next instruction is fetched and placed in 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 Series 64 instruction set is presented in Appendix E. Instructions are 16 or 32 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 extended to 28 digits in precision. In addition, there are a number of instructions designed to aid in creating the multi-programming 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 WCS, and associated logic control. All of the microcode except for a 256 word bootstrap is contained in the WCS. At initial power on, the bootstrap PROM in the Diagnostic Control Unit (DCU) is enabled, and the microcode contained in it is executed. Under operator control from the system console, this microcode loads the system microcode into the WCS and then branches to the starting point in the WCS. The WCS contains all of the microcode to implement the instruction set. Microdiagnostics can also be loaded into the WCS and run by the DCU. It is possible to load microdiagnostics before loading the instruction set to verify the integrity of the machine before the system begins execution. Microcode routines control the operation of the instruction decoder and the hardware processor in order to create the HP 3000 operating environment. The microinstruction cycle time is 75 nanoseconds.

The hardware processor consists of two arithmetic logic units, a shift register, specific purpose registers, and related data manipulating and testing logic. A unique dual ALU design is used to increase arithmetic processing power. Using this design, the Series 64 CPU is able to perform two 16-bit, or a single 32-bit operation in a single CPU cycle. Performance is increased as fewer microinstruction cycles are required. Since the HP 3000 architecture (see chapter 3) 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. Table D-2 lists all registers and their associated functions.

The eight top of stack registers are of special interest. In order to improve execution speed, up to seven 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 Series 64 Hardware Registers

Register	Function
PB	Code Segment Pointers
P	
PL	
BNKP	
BNKD	Stack Pointers
BNKS	
DL	
DB	
Q	
SM	
Z	
RA	Top of Stack Registers
RB	
RC	
RD	
RE	
RF	
RG	
RH	
X	Index Register
STAT	Status Register
PERF	Performance Register
CIR	Current Instruction Register
NIR	Next Instruction Register
OPA	Cache Operand Register
OPB	
PDB	Cache Memory Address and Data Registers
CAR	
NAMER	Top of Stack Address Register
ENAMR	
CSAR	Firmware Registers
RAC	
RAR	
RRGA	ALU Registers
SRGA	
RRGB	
SRGB	
BKX3-BKX7	Scratch Pad, Flag, Interrupt Registers, and Counters
SP0-SP4	
CPX1-CPX2	
F1-F5	
SR	
ESR	
CTRS	
XTRA0-XTRA127	
XTRB0-XTRB127	

ache memory

he Series 64 uses a hierarchical memory to reduce the me required to access memory. The hierarchical mem- ry consists of a main memory module and an 8 kbyte ache memory.

ache memory, acting as a high speed buffer between ie CPU and main memory, increases system perfor- iance by decreasing memory access time. High speed CL random access memories (RAMs) used in the cache low it to supply a word to the CPU in 75 nsec, or one PU cycle. However, only a fraction of main memory in be in the cache at one time. With the design of the eries 64 memory system, the CPU will on average find the word in the cache 95% of the time. The other 5% of the time, the word required by the CPU resides in main memory and the CPU must wait while it is read. The net result is the fastest average memory access time of any HP 3000, 145 nsec.

The cache is responsible for supplying data to the CPU upon request. CPU requests are made by passing the address of the word desired to the cache. This address is compared with the addresses currently stored in the cache and if the word is found, it is passed to the CPU. However, if not found, the word is loaded into the cache from main memory and passed to the CPU.

Data is transferred from cache to main memory and from main memory to cache in eight word blocks. Since the CPU is likely to request adjacent words of main memory, this method reduces the number of times the CPU cannot find words in the cache. Whether or not the cache has the word is transparent to the CPU, except in how long it takes to service the request. At any time there can be up to two requests pending to the cache, one for each ALU.

Main memory

The significant features of the HP 3000 Series 64 main memory are listed in Table D-3.

Table D-3 HP 3000 Series 64 Main Memory Features

- High-speed dynamic NMOS random access memory (64 K RAMs)
- Automatic fault detection and correction
- Memory sizes ranging from 2 to 8 megabytes
- 8 word block memory access
- Average memory access time (with cache): 145 nsec
- Write cycle time: 975 nsec minimum (8 word block)
- Read cycle time: 900 nsec (8 word block)
- 15 minute (minimum) rechargeable battery pack to maintain memory data during power failure. Total amount of backup time depends on memory size and battery condition (age and level of charge)

The HP 3000 Series 64 uses high speed random access memory (64 Kb NMOS RAM chips). The memory subsystem has the capability of detecting single bit and double bit errors and correcting single bit errors.

The modular design of the HP 3000 Series 64 allows the system to have its main memory expanded from 2 megabytes to 8 megabytes in 1 megabyte increments.

At any time, the latest updated copy of a block of memory may reside in either main memory, the I/O buffer of an I/O Adapter, or in cache memory. To assure that the latest updated copy is accessed, cache memory and I/O buffers will supply data for any memory request if they have the latest updated copy.

Central System Bus

The Central System Bus (CSB) is the communication link between the CPU Module, Main Memory Module, and Input/Output Adapter Modules. A 53 Mbyte/second overall bandwidth allows the CSB to support multiple IMBs. Over the CSB, no module has implied control. Modules operate independently of each other except when it is necessary to transfer data or send commands. When this is necessary, the initiating module asks for and receives control of the CSB. All transfers to and from memory are in 8 word blocks.

The Common Bus Interface (CBI) serves as the interface to the CSB. Each of the three modules; cache, I/O Adapter, and main memory; require a CBI to communicate across the CSB to another module.

I/O Adapter

Providing an interface between the Central System Bus and Intermodule Busses, Input/Output Adapters (IOAs) allow communication between the I/O system and the main memory and CPU. IOAs synchronize the relatively slow speed IMB with the CSB. The IOA controls direct memory access (DMA) between main memory, which is on the CSB, and I/O channels on the IMB. A 64 byte buffer cache memory included with each IOA buffers communication between the 16-bit IMB and the 32-bit CSB. To the devices on the IMB, the IOA appears as an effective memory by responding to IMB requests for read, write, and read/write. These requests are generated by the I/O controllers on behalf of the devices.

Series 64 system serviceability

The Series 64 is designed to be extremely reliable and easy to service. New hardware which enhances Series 64 serviceability (both hardware and software) includes the Diagnostic Control Unit (DCU) and the Power-Supply Control (PSC) Unit. They help reduce system downtime and allow the Series 64 to have low monthly maintenance costs. Table D-4 lists the features provided by the DCU.

Table D-4 HP 3000 Series 64 Diagnostic Control Unit Features

- Friendly user interface—MPE-like commands
- Remote diagnostic capability
- Operator access and modification of CPU registers
- Control for loading and running system diagnostics
- Intelligence that controls system power through the Power Supply Controller (PSC)
- Over temperature warning preceding temperature shutdown

The microprocessor controlled DCU can interrogate, modify, and diagnose all of the registers and most of the data paths in the CPU. It can also run diagnostic tests on the Main Memory, Cache Memory, I/O Adapter (IOA), and 80% of the General I/O Channel.

The DCU selftest is resident on the DCU in read only memory. Other microdiagnostics can be loaded into the WCS. Commands are MPE-like and error indications are given in simple English statements. The self-test takes approximately 30 seconds to execute and was designed to be simple enough to be initiated by the customer prior to requesting service from HP.

A unique logic design in the Series 64 allows the contents of any register in the CPU and many other system boards to be serially streamed into the DCU. This design requires no operational hardware except for the DCU. Each board has a single serial line running directly to the DCU. All of the registers and many other elements defining the machine state are connected in a serial shift string with this line. Under DCU control, the contents of any register can be examined or modified by appropriately shifting the serial shift string. Using this feature, DCU fault locating microdiagnostics can isolate most failures to a three board set and list that set on the system console. It can also be used by the system technician to examine register contents during system troubleshooting.

The DCU also protects main memory, user files, spool files and executing processes upon occurrences of a power failure. Similarly, when the DCU detects an overtemperature condition, it first generates a warning to the system console, and if the condition persists, will simulate a power failure condition shutting the system down and protecting circuits from being temperature stressed beyond normal operating specifications.

Most of the functions performed by the DCU are available remotely. By connecting a remote terminal to the DCU through a modem, a remote console may be operated in parallel with the system console. With this facility, an HP Customer Engineer can utilize DCU diagnostic features to identify and solve hardware problems while at a remote location.

The Series 64 Power Supply Controller (PSC) provides an interface to the power system. The PSC monitors and controls the power supply system to ensure the computer operates with valid power supplies. Also, it aids in the diagnosis and troubleshooting of power supply system faults and failures through status reporting to the DCU. Control of the PSC is accomplished through a control program running on the DCU.

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. All devices can be operated concurrently (within system bandwidth). Peripherals that fail are taken off-line from the operating system by operator command.

Peripheral I/O hardware

HP 3000 Series 64 peripheral I/O hardware consists of the General I/O Channels (GIC), DSN/Advanced Terminal Processors (ATP), DSN/Intelligent Network Processors (INP), and the peripheral units.

When an I/O request is issued, the device driver in the CPU performs different actions depending on whether or not a terminal is involved. With devices other than terminals, the CPU assembles a channel program, then issues a Start I/O Program (SIOP) instruction to the General I/O Channel over which the device communicates. For terminals, the host software driver receives calls from any of multiple sources for terminal actions and instructs the DSN/Advanced Terminal Processor to initiate the action. The DSN/Advanced Terminal Processor will interrupt the CPU when the action is completed.

The General I/O Channel is the hardware I/O channel which provides the electrical interface between the computer system via the IMB and peripheral devices connected to the computer system Hewlett-Packard Interface Bus (HP-IB). The HP-IB is HP's computer system implementation of the IEEE standard 488-1975 interface, used on the Series 64 to connect peripheral devices to the channel. The HP-IB consists of eight data lines and eight control lines. The ATP provides an intelligent interface between computer system via the IMB and terminals.

General I/O Channel

The General I/O Channel (GIC) is the primary channel for communication between the CPU and the I/O devices other than terminals. Each GIC controls a computer system Hewlett-Packard Interface Bus (HP-IB) and translates I/O commands from the CPU into the proper HP-IB protocol. Nearly all transactions with I/O devices are accomplished without software interrupts, since I/O is achieved with channel programs. Software is responsible for setting up a channel program, but the execution of this program is performed by the CPU's channel microcode. The CPU's channel microcode is devoted to I/O tasks and implements the necessary algorithms for decoding the channel instructions and effecting the required I/O operations. Once the channel program is running, device control and data flow are normally carried to completion with no software intervention and without altering the system environment.

Several devices may simultaneously need service, and the CPU must decide which one will receive attention. First, all channels are polled, and the highest priority channel with a device request pending is chosen. The CPU then obtains from that channel the number of the highest priority device needing service. Once the device number is determined, execution of the channel program will begin. The CPU fetches each channel instruction and breaks it down into several IMB commands addressed to the proper GIC. The GIC interprets these commands and directs them onto the HP-IB device.

The GIC contains Direct Memory Access (DMA) hardware which allows large records of data to be transferred at the maximum speed of the HP-IB (about 1 Mb/second). The channel microcode enables the device and then initiates the DMA hardware on the GIC. After initial addressing of a device to talk or listen, the CPU relinquishes control of the IMB and allows the GIC to perform its function through DMA operation. During this time the GIC becomes the master of the IMB and the IOA and controls traffic flow. On a read operation the DMA hardware will read the bytes, pack them into words and place them directly into memory, all without assistance from the CPU. The CPU is free to service other devices while DMA is in progress. Upon completion of a DMA transfer, the GIC returns to a slave condition and awaits the next operation.

Device controller

The device controller is the hardware linkage between a peripheral device and the computer system. Its primary function is to translate I/O commands from a General I/O Channel (GIC) 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 GIC. The device controller also generates interrupts when required by some device condition or by channel command.

Device reference table (DRT)

Device controllers are identified by a logical 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. Each device controller connects to a general I/O channel (GIC). Certain device controllers may control several logical devices. In such cases, each logical 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 logical device number, there are two other characteristic numbers associated with each device controller. These are the data service priority and interrupt priority. In the Series 64 both of these are determined by the logical device number in the DRT: the lower the number, the higher the priority.

DSN/Advanced Terminal Processor

An intelligent interface between terminals and the CPU is provided by the DSN/Advanced Terminal Processor (ATP). The significant features of the DSN/Advanced Terminal Processor are listed in Table D-5.

Table D-5 DSN/Advanced Terminal Processor Features

- Data transfer rates up to 9600 bits/second
- Handles character processing, eliminates CPU interrupts
- Expandable from 12 to 96 terminals
- RS-232-C and RS-422 support (local terminals up to 15m or 1220m, respectively from the CPU)
- Full-duplex asynchronous modem support
- Direct memory access of user data

An ATP is composed of one System Interface Board (SIB) and from one to eight Port Controllers. The SIB provides a hardware interface to the Intermodule Bus (IMB) and, under microprocessor control, performs byte packing and unpacking and controls direct memory access (DMA) of user data. Port Controllers provide the hardware interface for terminal/workstation devices to the Series 64. With a microprocessor dedicated to each terminal port, the Port Controller handles all handshaking between the system and the connected devices, character echoing, speed sensing, and input character buffering. The Series 64 may use multiple ATPs.

The ATP allows terminals to transmit and receive data on either a character-by-character basis or a block-at-a-time basis. For both types of operations, the ATP transfers data directly to and from memory. Because this eliminates the need for character processing by the CPU, the ATP significantly reduces CPU utilization.

A flexible set of physical attachment interfaces are available to allow asynchronous terminals to be attached directly, for local (intrafacility) use, or through full-duplex modems (Bell type 103, 202T, 212A, and CCITT V.24 type modems) for remote installations.

SN/Intelligent Network Processor

e DSN/Intelligent Network Processor (INP) allows HP
)0 computers to be linked to other computers in a dis-
)uted data processing environment and supports mul-
oint terminals. The significant features of the DSN/
elligent Network Processor are listed in Table D-6.

Table D-6 DSN/Intelligent Network Processor Features

- 16-bit SOS microprocessor
- Data communications protocol handling
- Character handling and 32 kbytes buffer storage
- Modem and hardwired interfaces up to 56,000 bits/second
- Full- and half-duplex asynchronous modem support
- Bisync and HDLC/SDLC protocol compatible
- RS-232-C, RS-422, CCITT V.24 and V.35 interfacing
- Direct memory access for data
- Auto call capability

e INP microprocessor performs all of the commu nica-
n data link protocol management, including: ser ali-
ion, protocol management, frame/block management,
d data buffering. This reduces CPU utilization and
es it to perform other tasks. Throughput is incre ased
overlapping the transfer of data already received
m the communication channel with the processing
d buffering of new data coming from the communica-
n channel. The protocol driver may be dynamically
anged. This allows the INP to be easily reconfigured
m one data link protocol to another and permits sever-
subsystems to use a single INP. An auto call capability
ncluded with the INP. It allows a remote connec ion
)ccur in a dial-up environment without the interven-
n of a human operator.

Interrupt system

The interrupt system provides for up to 105 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 in 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 20 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

The peripheral devices used on the HP 3000 Series 64 are connected primarily to GICs, while the ATP is reserved solely for terminals. Peripherals attached to GICs through the HP-IB include disc drives, line printers, magnetic tape drives, card readers and DSN/Intelligent Network Processors. For a complete configuration of the supported peripherals on the HP 3000 Series 64, refer to the current HP 3000 Configuration Guide.

Automatic restart after power failure

An integral part of the HP 3000 Series 64 is a power fail/automatic restart capability. When the system AC line voltage falls below 90% of rated voltage, the system initiates a powerfail warning (PFW). During PFW the system (hardware and MPE) writes all register contents to a reserved section of main memory, critical activities in the system are completed, and then the power down signal is generated to shut the system down. The battery back-up power supply refreshes main memory and ensures its validity for at least 15 minutes, depending on memory size and battery condition.

The system is automatically restarted when power supply voltages reach 90% of their rated values and all register values are automatically restored and processing resumes.

Appendix E:
HP 3000 Series 64
Machine Instructions

Stack Op Instructions

)AX	Add A to X
)BX	Add B to X
)D	Add A to B
)XA	Add X to A
)XB	Add X to B
ND	Logical AND of A and B
'ST	Test byte on TOS and set CC
AB	Rotate A-B-C
MP	Integer compare B, A and set CC
ADD	Double integer add D, C, + B, A
CMP	Double integer compare and set CC
)EL	Double delete TOS
)IV	Double integer divide
)UP	Double duplicate TOS
ECA	Decrement A
ECB	Decrement B
ECX	Decrement X
EL	Delete TOS
ELB	Delete B
FLT	Float a double integer
IV	Integer divide B by A
IVL	Divide long integer C, B ÷ A
MUL	Double integer multiply
NEG	Double integer negate
SUB	Double integer subtract D, C − B, A
TST	Test double word on TOS and set CC
UP	Duplicate TOS
XCH	Double exchange
ZRO	Push double zero onto stack
ADD	Floating point add, D, C + B, A
CMP	Floating point compare and set CC
)IV	Floating point divide D, C ÷ B, A
XR	Fix and round
FIXT	Fix and truncate
FLT	Float an integer
FMPY	Floating point multiply
FNEG	Floating point negate
FSUB	Floating point subtract D,C − B,A
INCA	Increment A
INCB	Increment B
INCX	Increment X
LADD	Logical add A + B
LCMP	Logical compare B, A and set CC
LDIV	Logical divide C, B ÷ A
LDXA	Load X into A
LDXB	Load X into B
LMPY	Logical multiply B × A
LSUB	Logical subtract B − A
MPY	Multiply integers, integer product
MPYL	Multiply integers, long integer product
NEG	Integer negate
NOP	No operation
NOT	Logical complement TOS
OR	Logical OR of A, B
STAX	Store A into X
STBX	Store B into X
SUB	Integer subtract B − A
TEST	Test TOS and set CC
XAX	Exchange A and X
XBX	Exchange B and X
XCH	Exchange A and B
XOR	Logical exclusive OR of A, B
ZERO	Push integer zero onto stack
ZROB	Zero B
ZROX	Zero X

Shift Instructions

SL	Arithmetic shift left
SR	Arithmetic shift right
SL	Circular shift left
SR	Circular shift right
ASL	Double arithmetic shift left
ASR	Double arithmetic shift right
CSL	Double circular shift left
CSR	Double circular shift right
LSL	Double logical shift left
DLSR	Double logical shift right
LSL	Logical shift left
LSR	Logical shift right
QASL	Quadruple arithmetic shift left
QASR	Quadruple arithmetic shift right
TASL	Triple arithmetic shift left
TASR	Triple arithmetic shift right
TNSL	Triple normalizing 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

Program Control and Special Instructions

DISP	Dispatch	**PCN**	Push CPU code (% 4)
EXIT	Exit from procedure	**PSDB**	Pseudo interrupt disable
HALT	Halt	**PSEB**	Pseudo interrupt enable
IXIT	Interrupt exit	**RSW**	Push cold load chan/dev
LLBL	Load label	**SCAL**	Subroutine call
LLSH	Linked list search	**SXIT**	Exit from subroutine
PAUS	Pause, interruptable	**XEQ**	Execute stack word
PCAL	Procedure call		

Machine and I/O Instructions

HIOP	Halt I/O program	**SCLR**	Set system clock limit
INIT	Initialize I/O channel	**SED**	Set enable/disable external interrupts
RCCR	Read system clock	**SMSK**	Set device mask
RIOA	Read I/O Adapter	**TOFF**	Hardware timer off
RMSK	Read device mask	**TON**	Hardware timer on
STRT	Programmatic warm start	**WIOA**	Write I/O Adapter
SIOP	Start I/O channel program	**MCMD**	Message command
SINC	Set System Clock Interrupt	**DUMP**	Programmatic dump
FLSH	Flush CPU and I/O Cache		

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

EADD	ADD
ECMP	Compare
EDIV	Divide
EMPY	Multiply
ENEG	Negate
ESUB	Subtract

Decimal Arithmetic

ADDD	Decimal add
CMPD	Decimal compare
CVAD	ASCII to decimal conversion
CVBD	Binary to decimal conversion
CVDA	Decimal to ASCII conversion
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

Field and Bit Instructions

F	Deposit field, A bits to B	**TCBC**	Test and complement bit, set CC
F	Extract specified field, right-justify	**TRBC**	Test and reset bit, set CC
AN	Scan bits	**TSBC**	Test and set bit, set CC
C	Test specified bit and set CC		

Branch Instructions

C	Branch on specified CC	**BRO**	Branch on TOS odd (bit 15 = 1)
Y	Branch on carry	**CPRB**	Compare range and branch
NCY	Branch on no carry	**DABZ**	Decrement A, branch if zero
NOV	Branch on no overflow	**DXBZ**	Decrement X, branch if zero
OV	Branch on overflow	**IABZ**	Increment A, branch if zero
R	Branch unconditionally	**IXBZ**	Increment X, branch if zero
RE	Branch on TOS even (bit 15 = 0)		

Move Instructions

MPB	Compare bytes in two memory blocks	**MVB**	Move bytes in memory, addresses +/−
ABS	Move using absolute addresses	**MVBL**	Move words from DB+ to DL+ area
DS	Move using data segments	**MVBW**	Move bytes while of specified type
FDS	Move from data segment	**MVLB**	Move words from DL+ to DB+ area
OVE	Move words in memory, addresses +/−	**SCU**	Scan bytes until test or terminal byte
TDS	Move to data segment	**SCW**	Scan bytes while equal to test byte

Privileged Memory Reference Instructions

DEA	Load double word from extended address	**PSTA**	Privileged store into absolute address
EA	Load single word from extended address	**SDEA**	Store double word into extended address
T	Load from system table	**SSEA**	Store single word into extended address
DA	Privileged load from absolute address	**SST**	Store into system table

Immediate Instructions

ADI	Add immediate to integer in A	**LDXI**	Load X immediate
AXI	Add immediate to X	**LDXN**	Load X negative immediate
NDI	Logical AND immediate with A	**MPYI**	Multiply immediate with A
MPI	Compare A with immediate, set CC	**ORI**	Logical OR immediate with A
MPN	Compare A with negative immediate	**SBXI**	Subtract immediate from X
VI	Divide immediate into A	**SUBI**	Subtract immediate from A
DI	Load immediate to TOS	**XORI**	Logical exclusive OR immediate
DNI	Load negative immediate to TOS		

Register Control Instructions

ADS	Add operand to stack pointer	**SETR**	Set specified registers from stack
HR	Push specified registers onto stack	**SUBS**	Subtract operand from stack pointer
LK	Read clock	**XCHD**	Exchange DB and TOS
LK	Store clock		

Appendix F:
HP 3000 Series 44
Hardware Features

This appendix provides a summary of hardware features of the HP 3000 Series 44 computer system. For more details on the HP 3000 architecture, refer to Chapter 1—System Introduction and Chapter 3—System Architecture.

The HP 3000 Series 44 design is based on a modular concept which allows independent elements to be interconnected through a central system bus structure. These elements consist of a central processing unit which controls memory via a controller, General I/O Channels, DSN/Advanced Terminal Processor, DSN/Asynchronous Data Communications Controllers, and the bus system which allows communication between the I/O devices. Also, the system includes a system console and a Control and Maintenance Processor (CMP). Peripheral devices are connected to the system through the General I/O Channels. Interactive terminals are attached to the system through either the DSN/Advanced Terminal Processor, DSN/Asynchronous Data Communications Controllers, or DSN/Intelligent Network Processors running the DSN/Multipoint Terminal Software. Data communication links are established via DSN/Intelligent Network Processors.

The CPU is centered around a Hewlett-Packard designed microcoded processor using Schottky TTL technology. Implementation using this technology provides high speed execution of instructions while allowing flexibility in the machine instruction set. The Series 44 also employs high speed, semiconductor, random access memory modules which use automatic fault detection and correction.

The hardware design of the HP 3000 Series 44 provides the ability to expand the system as the users needs grow and applications change. The system software accommodates these changing needs by allowing additional hardware modules and peripheral devices to be easily configured on the system.

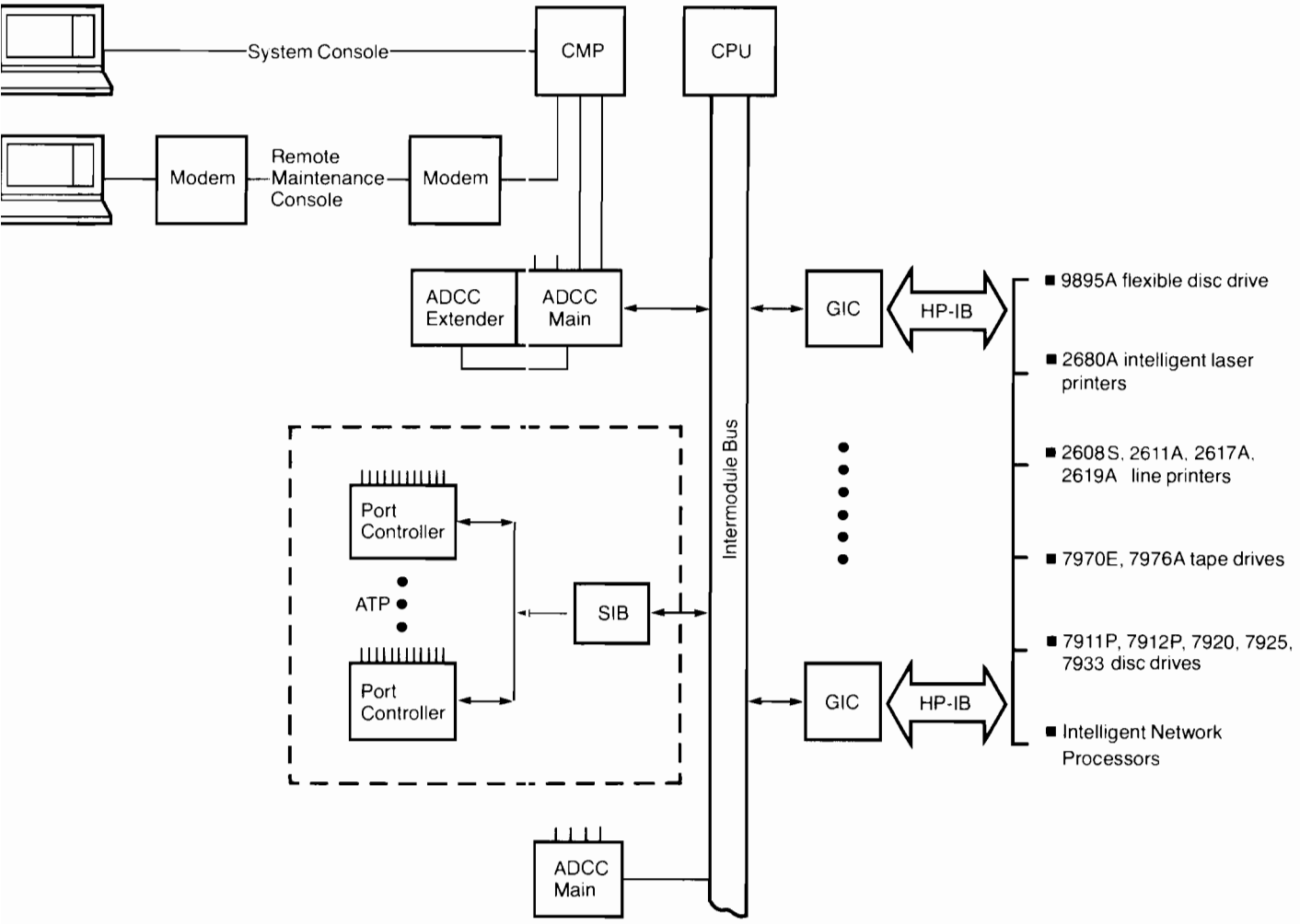


Figure F-1. HP 3000 Series 44 Hardware Organization

The HP 3000 Series 44 instruction set is presented in Appendix H. Instructions are 16 or 32 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 extended to 28 digits in precision. In addition, there are a number of instructions designed to aid in creating the multi-programming 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 logic control. Microcode routines control the operation of the instruction decoder and the hardware processor in order to create the HP operating environment. The microinstruction cycle time is 105 nanoseconds.

The hardware processor consists of an arithmetic-logic unit, shifting network, 72 specific purpose registers—18 of which are accessible to user programs—and related data manipulating and testing logic. Since the HP 3000 architecture (see chapter 3) 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 F-2, three of the CPU registers point to locations in a code segment defined as the current code segment. Five of the registers point to locations in a data segment defined as the current data segment. Table F-2 lists all 72 registers and their associated functions.

Central Processing Unit (CPU)

The significant features of the HP 3000 Series 44 central processing unit (CPU) are listed in Table F-1.

The CPU converts an instruction in the current instruction register (CIR) into a starting address for the microcode contained in the 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 in 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.

Table F-1 HP 3000 Series 44 CPU Features

Architecture

- Hardware-implemented stack
- Separation of code and data
- Non-modifiable re-entrant code
- Variable-length code segmentation
- Virtual memory for code
- Dynamically relocatable programs

Implementation

- Microprogrammed Schottky TTL CPU
- 105 nanosecond microinstruction cycle time
- Automatic restart after power failure
- Intermodule bus
- Overlapping CPU and I/O operations

Instructions

- 214 powerful instructions
- Instructions are 8, 16, and 32 bits in length
- 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

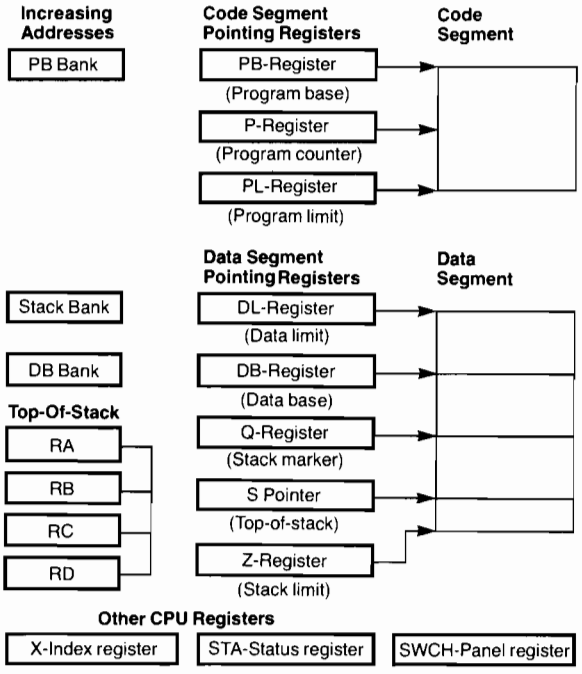


Figure F-2. HP 3000 CPU REGISTERS

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 F-2 Series 44 Hardware Registers

Registers accessible to user programmers

Register	Function
PB, P, PL, PBNK	Code Segment Pointers
DBNK, SBNK, DL, DB, Q, SM, Z	Stack Pointers
RA, RB, RC, RD	Top of Stack Registers
X	Index Register
STA	Status Register
SWCH	Front Panel Switch Register

Registers dedicated for system use

CIR	Current Instruction Register
NIR	Next Instruction Register
R4-R13, R15-R39, SIR, SP0, CTR, ABNK, BBNK, F1-F4, NF1-NF4	Scratch Pad, Flag, and Interrupt Registers
OPND, UBUS	Memory Address and Data Registers
CSAR	Firmware Address Register

Main memory

The significant features of the HP 3000 Series 44 main memory are listed in Table F-3.

Table F-3 HP 3000 Series 44 Main Memory Features

- High-speed dynamic NMOS random access memory
- Automatic fault detection and correction
- Memory sizes ranging from 1 megabyte to 4 megabytes
- Write cycle time: 555 nsec minimum
- Read access time: 300 nsec
- Read cycle time: 430 nsec
- 15 minute (minimum) rechargeable battery pack to maintain memory data during power failure. Total amount of backup time depends on memory size and battery condition (age and level of charge)

The HP 3000 Series 44 uses high speed random access memory. The memory subsystem has the capability of detecting single bit and double bit errors and correcting single bit errors.

The modular design of the HP 3000 Series 44 allows the system to have its main memory expanded from 1 megabyte to 4 megabytes.

The word length transmitted over the intermodule bus is 16 bits. In the memory modules, the word length is expanded to 39 bits; 32 bits of data and 7 bits of automatic fault detection and correction.

Series 44 system serviceability

The Series 44 is designed to be extremely reliable and easy to service. Hardware which enhances Series 44 serviceability (both hardware and software) is the Control and Maintenance Processor (CMP). The microprocessor based CMP provides diagnostic and control functions for identifying hardware problems. It helps reduce system downtime and allows the Series 44 to have low monthly maintenance costs. Table F-4 lists the features provided by the CMP.

Table F-4. HP 3000 Series 44 Control and Maintenance Processor

- Friendly user interface—MPE-like commands
- System Console Flexibility-any 262x, 264x, 2382A, or 2635B terminal may be used as a system console
- System selftest
- Remote diagnostic capability
- Over temperature shutdown

The diagnostics are resident on the CMP in read only memory. Commands are MPE-like and error indications are given in simple English statements. The self-test takes approximately 30 seconds to execute and was designed to be simple enough to be initiated by the customer prior to requesting service from HP. The CMP also protects main memory, user files, spool files and executing processes upon occurences of a power failure.

All the functions performed by the CMP are available remotely. By connecting a remote terminal to the CMP through a modem, a remote console may be operated in parallel with the system console. With this facility, an HP Customer Engineer can utilize CMP diagnostic features to identify and solve hardware problems while at a remote location.

I/O System

Communication between the CPU, memory and I/O modules is carried over the Intermodule Bus (IMB). Because the CPU generates greater than 90 percent of the bus activity, it is given continuous access to the bus and relinquishes control to the I/O channels only on request.

The IMB has separate address and data paths, each with handshake controls that operate in a master/slave mode to transfer data between modules. The CPU talks to memory and to the I/O system and always functions as a master. The I/O channels function as masters to memory but become slaves when talking with the CPU. To access memory, the I/O channels must request the bus through a priority structure. Any channel request will cause the CPU to relinquish control of the IMB so that the request can be serviced.

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. All devices can be operated concurrently (within system bandwidth). Peripherals that fail are taken off-line from the operating system by operator command.

Peripheral I/O hardware

HP 3000 Series 44 peripheral I/O hardware consists of the General I/O Channels (GIC), DSN/Advanced Terminal Processors, DSN/Asynchronous Data Communication Controllers, DSN/Intelligent Network Processors, and the peripheral units. DSN/Advanced Terminal Processors and DSN/Asynchronous Data Communications Controllers interface log-on and data-entry terminals.

When an I/O request is issued, the device driver in the CPU performs different actions depending on whether or not a terminal controlled by an DSN/Advanced Terminal Processor is involved. With devices other than terminals controlled by an ATP, the CPU assembles a channel program, then issues a Start I/O Program (SIOP) instruction to the General I/O Channel or DSN/Asynchronous Data Communications Controller over which the device communicates. For terminals controlled by an ATP, the host software driver receives calls from any of multiple sources for terminal actions and instructs the DSN/Advanced Terminal Processor to initiate the action. The DSN/Advanced Terminal Processor will interrupt the CPU when the action is completed.

The General I/O Channel is the hardware I/O channel which provides the electrical interface between the computer system via the IMB and peripheral devices connected to the computer system Hewlett-Packard Interface Bus (HP-IB). The HP-IB is HP's computer system implementation of the IEEE standard 488-1975 interface, used on the Series 44 to connect peripheral devices to the channel. The HP-IB consists of eight data lines and eight control lines. The ATP provides an intelligent interface between computer system and terminals.

General I/O Channel

The General I/O Channel (GIC) is the primary channel for communication between the CPU and the I/O devices other than terminals. Each GIC controls a computer system Hewlett-Packard Interface Bus (HP-IB) and translates I/O commands from the CPU into the proper HP-IB protocol. Nearly all transactions with I/O devices are accomplished without software interrupts, since I/O is achieved with channel programs. Software is responsible for setting up a channel program, but the execution of this program is performed by the CPU's channel microcode. The CPU's channel microcode is devoted to I/O tasks and implements the necessary algorithms for decoding the channel instructions and effecting the required I/O operations. Once the channel program is running, device control and data flow are normally carried to completion with no software intervention and without altering the system environment.

everal devices may simultaneously need service, and he CPU must decide which one will receive attention. irst, all channels are polled, and the highest priority hannel with a device request pending is chosen. The PU then obtains from that channel the number of the ighest priority device needing service. Once the device umber is determined, execution of the channel pro- ram will begin. The CPU fetches each channel instruc- ion and breaks it down into several IMB commands ddressed to the proper GIC. The GIC interprets these ommands and directs them onto the HP-IB device.

he GIC contains Direct Memory Access (DMA) hard- vare which allows large records of data to be trans- erred at the maximum speed of the HP-IB (abou: 1 Mb/ econd). The channel microcode enables the device and hen initiates the DMA hardware on the GIC. After ini- al addressing of a device to talk or listen, the CPU re- nquishes control of the IMB and allows the GIC to erform its function through DMA operation. During his time the GIC becomes the master of the bus and emory and controls traffic flow. On a read operation he DMA hardware will read the bytes, pack them into vords and place them directly into memory, all without ssistance from the CPU. The CPU is free to service ther devices while DMA is in progress. Upon comple- on of a DMA transfer, the GIC returns to a slave condi- on and awaits the next operation.

Device controller

he device controller is the hardware linkage between a eripheral device and the computer system. Its primary unction is to translate I/O commands from a general I/O hannel (GIC) to the unique signals required to control a articular device. When an I/O program is in execution, he device controller responds to and requests service rom the GIC. The device controller also generates inter- upts when required by some device condition or by hannel command.

Device reference table (DRT)

Device controllers are identified by a logical device num- er which is used to access the device reference table DRT). The DRT is known to both hardware and soft- vare, containing among other things, a pointer to the tart of the SIO program for each device controller. Each evice controller connects to a General I/O Channel GIC). Certain device controllers may control several ogical devices. In such cases, each logical device at- ached to the controller is addressed separately using a nit number assigned when the device is installed.

Data service and interrupt priorities

In addition to a logical device number, there are two other characteristic numbers associated with each device controller. These are the data service priority and interrupt priority. In the Series 44 both of these are determined by the logical device number in the DRT: the lower the number, the higher the priority.

DSN/Advanced Terminal Processor

An intelligent interface between terminals and the CPU is provided by the DSN/Advanced Terminal Processor (ATP). The significant features of the DSN/Advanced Terminal Processor are listed in Table F-5.

Table F-5 DSN/Advanced Terminal Processor Features

- Data transfer rates up to 9,600 bits/second
- Handles character processing, eliminating CPU interrupts
- Expandable from 12 to 60 terminals
- RS-232-C and RS-422 support (local terminals up to 15m or 1220m, respectively from the CPU).
- Direct memory access of user data

An ATP is composed of one System Interface Board (SIB) and from one to five Port Controllers. The SIB provides a hardware interface to the Intermodule Bus (IMB) and, under microprocessor control, performs byte packing and unpacking and controls direct memory access (DMA) of user data. Port Controllers provide the hardware interface for terminal/workstation devices to the Series 44. With a microprocessor dedicated to each terminal port, the Port Controller handles all handshaking between the system and the connected devices, character echoing, speed sensing, and input character buffering.

The ATP allows terminals to transmit and receive data on either a character-by-character basis or a block-at-a-time basis. For both types of operations, the ATP transfers data directly to and from memory. Because this eliminates the need for character processing by the CPU, the ATP significantly reduces CPU utilization.

A flexible set of physical attachment interfaces are available to allow asynchronous terminals to be attached directly, for local (intrafacility) use.

DSN/Asynchronous Data Communication Controller

The DSN/Asynchronous Data Communications Controller (ADCC), the second channel type used in the system, provides a way to connect modem connect terminals and an alternative way to connect direct connect terminals. The Series 44 requires at least one ADCC for the CMP.

This channel performs for terminals essentially the same functions as the GIC but not in the same manner. Data is transferred from memory to the ADCC in parallel form, then converted to a serial bit stream for transmission over the RS-232-C lines to the device. Information being read from a device is in serial form and is converted to eight-bit bytes for transfer to memory.

Two types of ADCC boards may be used, the Main ADCC and the Extender ADCC. Each board contains four ports for connections to devices through RS-232-C data communication lines. The Main ADCC supports full duplex (Bell type 103, 212, and 202T modems in the U.S.) only. The Main ADCC is used when four or fewer devices are connected to a channel. The Extender ADCC extends the device capability of the channel to eight. Most of the control circuitry is on the Main ADCC. For this reason, the Main ADCC is required for the Extender ADCC to function.

When more than eight devices are to be attached to an ADCC channel, additional Main ADCCs are required, since each ADCC can accommodate only one Extender.

Unlike the GIC, the ADCC does not have a DMA facility, and therefore, cannot be the master of an IMB or of memory. Also, terminals on the channel do not respond to a parallel poll. As a result, the ADCC is always a slave and must be directly controlled by the CPU through the use of channel programs. Circuitry on the ADCC decodes the address information relating to channels and devices, and selects the correct device for operation.

The ports on the ADCC (Main and Extender) may be either hardwired to devices or to full-duplex modems.

DSN/Intelligent Network Processor

The DSN/Intelligent Network Processor (INP) allows HP 3000 computers to be linked to other computers in a distributed data processing environment and supports multipoint terminals. The significant features of the DSN/Intelligent Network Processor are listed in Table F-6.

Table F-6 DSN/Intelligent Network Processor Features

- 16-bit SOS microprocessor
- Data communications protocol handling
- Character handling and 32 kbytes buffer storage
- Modem and hardwired interfaces up to 56,000 bits/second
- Full- and half-duplex asynchronous modem support
- Bisync and HDLC/SDLC protocol compatible
- RS-232-C, RS-422, CCITT V.24 and V.35 interfacing
- Direct memory access for data
- Auto call capability

The INP microprocessor performs all of the communication data link protocol management, including: serialization, protocol management, frame/block management, and data buffering. This reduces CPU utilization and frees it to perform other tasks. Throughput is increased by overlapping the transfer of data already received from the communication channel with the processing and buffering of new data coming from the communication channel. The protocol driver may be dynamically changed. This allows the INP to be easily reconfigured from one data link protocol to another and permits several subsystems to use a single INP. An auto call capability is included with the INP. It allows a remote connection to occur in a dial-up environment without the intervention of a human operator.

Interrupt system

The interrupt system provides for up to 105 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 in 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 20 internal interrupts (for user errors, system violations, hardware faults, and power fail/restart) plus fourteen traps for arithmetic errors and illegal use of instructions.

eripherals

ne peripheral devices used on the HP 3000 Series 44
e connected primarily to GICs, while the ATP and
DCC are reserved solely for terminals and remote
rinters. Peripherals attached to GICs through the HP-
include disc drives, line printers, and magnetic tape
ives. For a complete configuration of the supported
eripherals on the HP 3000 Series 44, refer to the current
P 3000 Configuration Guide.

utomatic restart after power failure

n integral part of the HP 3000 Series 44 is a power fail/
utomatic restart capability. When the system AC line
oltage falls below 90% of rated voltage, the system ini-
ates a powerfail warning (PFW). During PFW the sys-
m (hardware and MPE) writes all register contents to a
served section of main memory, activities in the sys-
m are successfully completed, and then the power
own signal is generated and the system is shut down.
ne battery back-up power supply refreshes main mem-
y and ensures its validity for up to 15 minutes, de-
ending on memory size and battery condition.

he system is automatically restarted when power sup-
y voltages reach 90% of their rated values and all reg-
ter values are automatically restored and processing
sumes.

Appendix G:
HP 3000 Series 40
Hardware Features

is appendix provides a summary of hardware fe a-
es of the HP 3000 Series 40 computer systems. For
ore details on the HP 3000 architecture, refer to Chap-
1—System Introduction and Chapter 3—System
chitecture.

e HP 3000 Series 40 design is based on a modular con-
ot which allows independent elements to be intercon-
cted through a central system bus structure. These
ements consist of a central processing unit which con-
ls memory via a memory controller, General I/O
annels, DSN/Asynchronous Data Communication
ntrollers and the bus system which allows communi-
ion between the I/O devices. Also, the system in-
ides a system console and a Control and Maintenance
ocessor (CMP). Peripheral devices are connected to
e system through the General I/O Channels. Interac-
e terminals are attached to the system through the
SN/Asynchronous Data Communications Controllers
DSN/Intelligent Network Processors running the
DSN/Multipoint Terminal Software. Data communication links are established via DSN/Intelligent Network Processors.

The CPU is centered around a Hewlett-Packard designed microcoded processor using Schottky TTL technology. Implementation using this technology provides high speed execution of instructions while allowing flexibility in the machine instruction set. The Series 40 also employs high speed, semiconductor, random access memory modules which use automatic fault detection and correction.

The hardware design of the HP 3000 Series 40 provides the ability to expand the system as the user's needs grow and applications change. The system software accommodates these changing needs by allowing additional hardware modules and peripheral devices to be easily configured on the system.

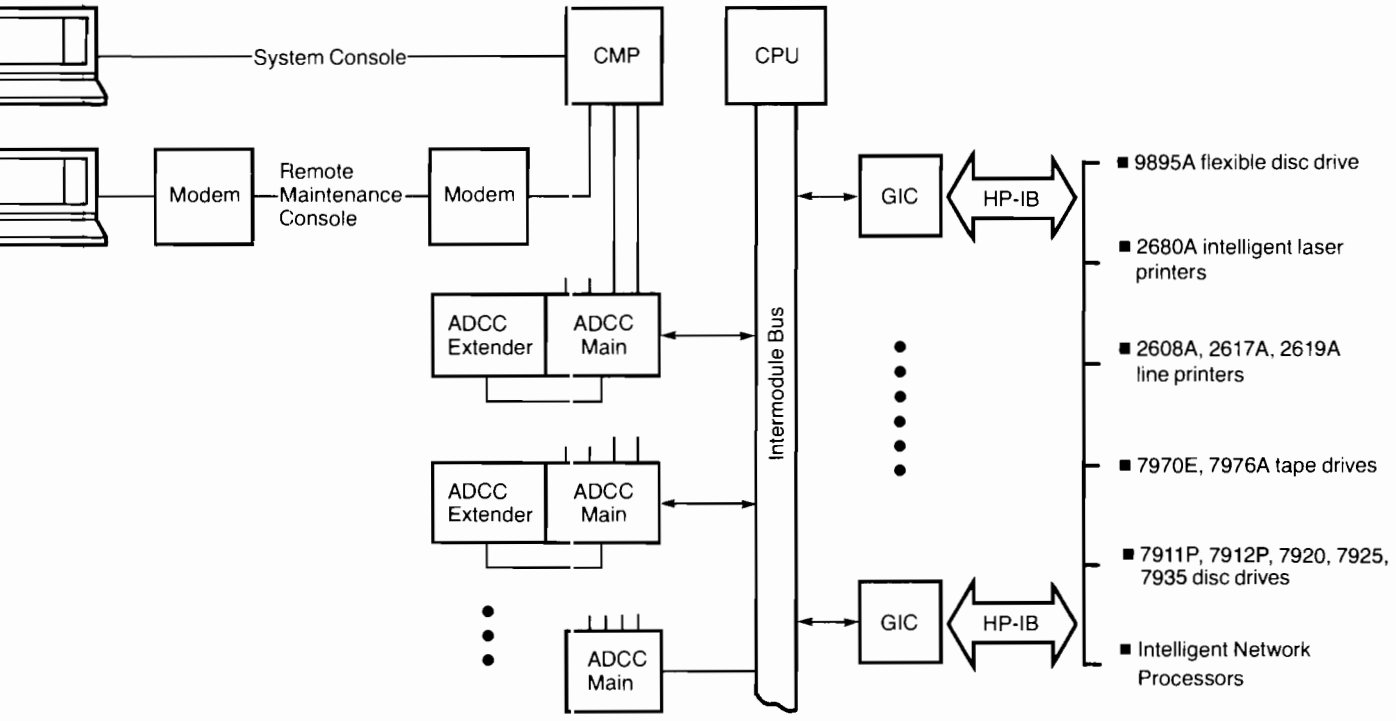


Figure G-1 HP 3000 Series 40 Hardware Organization

Central Processing Unit (CPU)

The significant features of the HP 3000 Series 40 central processing unit (CPU) are listed in Table G-1.

Table G-1 HP 3000 Series 40 CPU Features

Architecture

- Hardware-implemented stack
- Separation of code and data
- Non-modifiable re-entrant code
- Variable-length code segmentation
- Virtual memory for code
- Dynamically relocatable programs

Implementation

- Microprogrammed Schottky TTL CPU
- 105 nanosecond microinstruction cycle time
- Automatic restart after power failure
- Intermodule bus
- Overlapping CPU and I/O operations

Instructions

- 214 powerful instructions
- Instructions are 8, 16 and 32 bits in length
- 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.

The CPU converts an instruction in the current instruction register (CIR) into a starting address for the microcode contained in the 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 in 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 Series 40 instruction set is presented in Appendix H. Instructions are 16 or 32 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 sin gle precision (32 bits) or double precision (64 bits), inte ger arithmetic in 16-bit and 32-bit lengths, and packed decimal instructions extended to 28 digits in precision. In addition, there are a number of instructions designe to aid in creating the multi-programming environment of the system. These include procedure call and exit in structions and others which implement various operat ing system functions previously done in software.

Firmware storage and control consists of microcode stored in read-only memory (ROM), and associated logic control. Microcode routines control the operation of the instruction decoder and the hardware processor in order to create the HP operating environment. The microinstruction cycle time is 105 nanoseconds.

The hardware processor consists of an arithmetic-logic unit, shifting network, 72 specific purpose registers—1 of which are accessible to user programs—and related data manipulating and testing logic. Since the HP 3000 architecture (see Chapter 3) is structured on code segments and data segments, most of the CPU registers ar used for defining the segment limits and operating ele ments within the segments. As shown in Figure G-3, three of the CPU registers point to locations in a code segment defined as the current code segment. Five of the registers point to locations in a data segment defin as the current data segment. Table G-2 lists all 72 registers and their associated functions.

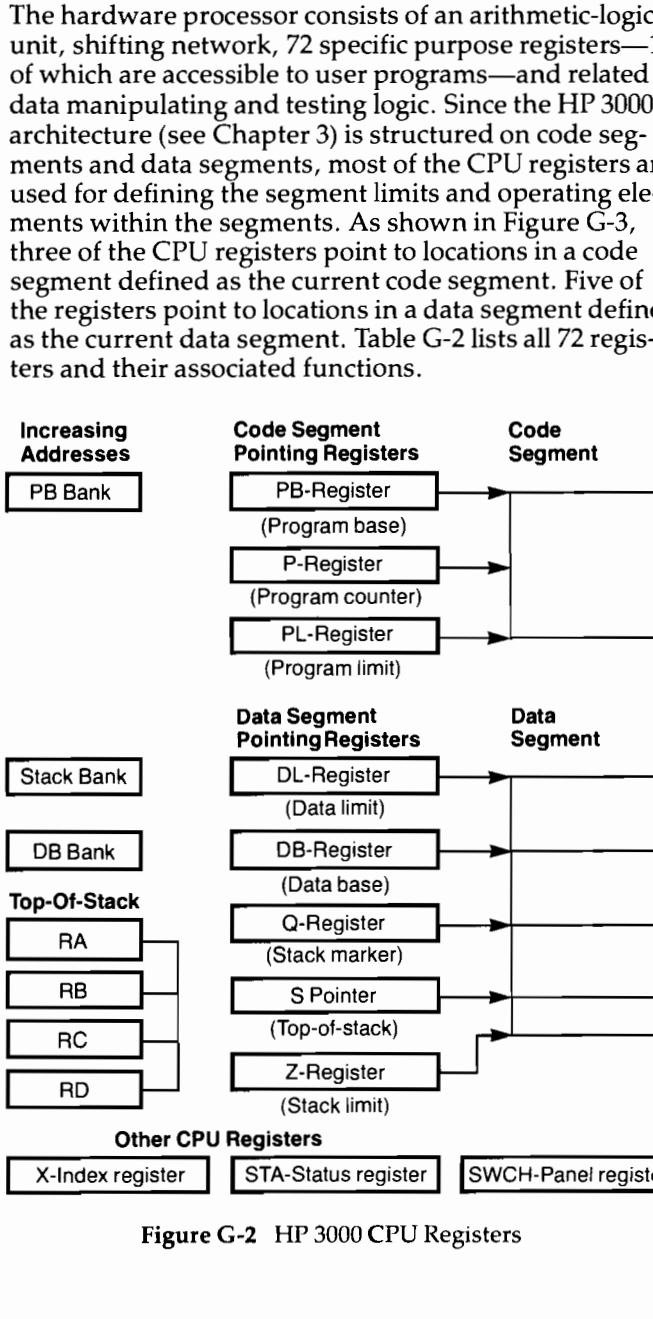


Figure G-2 HP 3000 CPU Registers

ne four top-of-stack registers are of special interest. In rder to improve execution speed, up to four elements om the top of a data stack may be contained in these gisters. This allows many functions to be treated as gister-to-register operations rather than the slower peed memory-to-register or register-to-memory type perations. These registers are manipulated by the PU, and their use is fully transparent.

Table G-2 Series 40 Hardware Registers

Registers accessible to user programmers

Register	Function
PB	Code Segment Pointers
P	
PL	
PBNK	
DBNK	Stack Pointers
SBNK	
DL	
DB	
Q	
SM	
Z	
RA	Top-of-Stack Registers
RB	
RC	
RD	
X	Index Register
STA	Status Register
SWCH	Front Panel Switch Register

Registers dedicated for system use

CIR	Current Instruction Register
NIR	Next Instruction Register
R4-R13	Scratch Pad, Flag, and Interrupt Registers
R15-R39	
SIR	
SP0	
CTR	
ABNK	
BBNK	
F1-F4	
NF1-NF4	
OPND	Memory Address and Data Registers
UBUS	
CSAR	Firmware Address Register

Main memory

The significant features of the HP 3000 Series 40 main memory are listed in Table G-3.

Table G-3 HP 3000 Series 40 Main Memory Features

- High-speed, dynamic NMOS random access memory
- Automatic fault detection and correction
- Memory sizes ranging from 256K bytes to 2 megabytes
- Write cycle time: 555 nsec minimum
- Read access time: 300 nsec
- Read cycle time: 430 nsec
- 15 minute (minimum) rechargeable battery pack to maintain memory data during power failure. Total amount of backup time depends on memory size and battery condition (age and level of charge).

The HP 3000 Series 40 uses high-speed random access memory. The memory subsystem has the capability of detecting single bit and double bit errors and correcting single bit errors.

The word length transmitted over the intermodule bus is 16 bits. In the memory modules, the word length is expanded to 39 bits; 32 bits of data and 7 bits of automatic fault detection and correction.

Series 40 system serviceability

The Series 40 is designed to be extremely reliable and easy to service. Hardware which enhances Series 40 serviceability (both hardware and software) is the Control and Maintenance Processor (CMP). The microprocessor based CMP provides diagnostic and control functions for identifying hardware problems. It helps reduce system downtime and allows the Series 40 to have low monthly maintenance costs. Table G-4 lists the features provided by the CMP.

Table G-4 HP 3000 Series 40 Control and Maintenance Processor

- Friendly user interface—MPE-like commands
- System Console Flexibility—any 262x, 264x, 2382A, or 2635B terminal may be used as a system console
- System selftest
- Remote diagnostic capability

All CMP programs are resident in read only memory. Commands are MPE-like and error indications are given in simple English statements. The selftest takes approximately 30 seconds to execute and was designed to be simple enough to be initiated by the customer prior to requesting service from HP.

All the functions performed by the CMP are available remotely. By connecting a remote terminal to the CMP through a modem, a remote console may be operated in parallel with the system console. With this facility, an HP Customer Engineer can utilize CMP diagnostic features to identify and solve hardware problems while at a remote location.

I/O System

Communication between the CPU, memory and I/O modules is carried over the Intermodule Bus (IMB). Because the CPU generates greater than 90 percent of the bus activity, it is given continuous access to the bus and relinquishes control to the I/O channels only on request.

The IMB has separate address and data paths, each with handshake controls that operate in a master/slave mode to transfer data between modules. The CPU talks to memory and to the I/O system and always functions as a master. The I/O channels function as masters to memory but become slaves when talking with the CPU. To access memory, the I/O channels must request the bus through a priority structure. Any channel request will cause the CPU to relinquish control of the IMB so that the request can be serviced.

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. All devices can be operated concurrently (within system bandwidth). Peripherals that fail are taken off-line from the operating system by operator command.

Peripheral I/O hardware

HP 3000 Series 40 peripheral I/O hardware consists of the General I/O Channels (GIC), DSN/Asynchronous Data Communication Controllers, DSN/Intelligent Network Processors, and the peripheral units. DSN/Asynchronous Data Communications Controllers interface log-on and data-entry terminals. Table G-5 lists the features which the peripheral I/O hardware offers.

Table G-5 HP 3000 Series 40 Peripheral I/O Hardware Features

- 4 ports per DSN/Asynchronous Data Communication Controller
- Up to 8 terminal controllers per system
- Bell type 103 and 212 modem support

When an I/O request is issued, the device driver in the CPU assembles the channel program, then issues a Start I/O Program (SIOP) instruction to one of two types of channels on the Intermodule Bus: the General I/O Channel (GIC) and the DSN/Asynchronous Data Communications Controller (ADCC). The GIC is the hardware I/O channel which provides the electrical interface between the computer system via the IMB and peripheral devices connected to the computer system Hewlett-Packard Interface Bus (HP-IB). The HP-IB is HP's computer system implementation of the IEEE standard 488-1975 interface used on the Series 40 to connect peripheral devices to the channel. The HP-IB consists of eight data lines and eight control lines. The ADCC provides a bit-serial data interface between computer system and terminals. The two channels operate in a similar manner; however, the GIC has a DMA facility to permit high-speed transfer of large blocks of data, while the ADCC can transfer data only one character at a time.

General I/O Channel

The General I/O Channel (GIC) is the primary channel for communication between the CPU and the I/O devices other than terminals. Each GIC controls a computer system Hewlett-Packard Interface Bus (HP-IB) and translates I/O commands from the CPU into the proper HP-IB protocol. Nearly all transactions with I/O devices are accomplished without software interrupts, since I/O is achieved with channel programs. Software is responsible for setting up a channel program, but the execution of this program is performed by the CPU's channel microcode. The CPU's channel microcode is devoted to I/O tasks and implements the necessary algorithms for decoding the channel instructions and effecting the required I/O operations. Once the channel program is running, device control and data flow are normally carried to completion with no software intervention and without altering the system environment.

Several devices may simultaneously need service, and the CPU must decide which one will receive attention. First, all channels are polled, and the highest priority channel with a device request pending is chosen. The CPU then obtains from that channel the number of the

ighest priority device needing service. Once the device umber is determined, execution of the channel pro-ram will begin. The CPU fetches each channel instruc-ion and breaks it down into several IMB commands ddressed to the proper GIC. The GIC interprets these ommands and directs them into the HP-IB device.

'he GIC contains Direct Memory Access (DMA) hard-vare which allows large records of data to be trans-erred at the maximum speed of the HP-IB (about 1 Mb/ econd). The channel microcode enables the device and nen initiates the DMA hardware on the GIC. After ini-al addressing of a device to talk or listen, the CPU re-nquishes control of the IMB and allows the GIC to erform its function through DMA operation. During his time the GIC becomes the master of the bus and nemory and controls traffic flow. On a read operation he DMA hardware will read the bytes, pack them into vords and place them directly into memory, all without ssistance from the CPU. The CPU is free to service ther devices while DMA is in progress. Upon comple-ion of a DMA transfer, the GIC returns to a slave condi-ion and awaits the next operation.

Device controller

'he device controller is the hardware linkage between a eripheral device and the computer system. Its primary unction is to translate I/O commands from a General 'O Channel (GIC) to the unique signals required to con-rol a particular device. When an I/O program is in exe-ution, the device controller responds to and requests ervice from the GIC. The device controller also gener-tes interrupts when required by some device condition r by channel command.

Device reference table (DRT)

)evice controllers are identified by a logical device num-er which is used to access the device reference table DRT). The DRT is known to both hardware and soft-vare, containing among other things, a pointer to the tart of the SIO program for each device controller. Each evice controller connects to a general I/O channel GIC). Certain device controllers may control several ogical devices. In such cases, each logical device at-ached to the controller is addressed separately using a nit number assigned when the device is installed.

Data service and interrupt priorities

n addition to a logical device number, there are two ther characteristic numbers associated with each de-ice controller. These are the data service priority and nterrupt priority. In the Series 40; both of these are de-ermined by the logical device number in the DRT: the ower the number, the higher the priority.

DSN/Asynchronous Data Communication Controller

The DSN/Asynchronous Data Communications Controller (ADCC) is the second channel type used in the system. This channel performs for terminals essentially the same functions as the GIC but not in the same manner. Data is transferred from memory to the ADCC in parallel form, then converted to a serial bit stream for transmission over the RS-232-C lines to the device. Information being read from a device is in serial form and is converted to eight-bit bytes for transfer to memory.

Two types of ADCC boards may be used, the Main ADCC and the Extender ADCC. Each board contains four ports for connections to devices through RS-232-C data communication lines. The Main ADCC supports full duplex (Bell type 103, 212, and 202T modems in the U.S.) only; Extender ADCC boards are required for European half duplex modem support. The Main ADCC is used when four or fewer devices are connected to a channel. The Extender ADCC extends the device capability of the channel to eight. Most of the control circuitry is on the Main ADCC. For this reason, the Main ADCC is required for the Extender ADCC to function.

When more than eight devices are to be attached to an ADCC channel, additional Main ADCCs are required, since each ADCC can accommodate only one Extender.

Unlike the GIC, the ADCC does not have a DMA facility, and therefore, cannot be the master of an IMB or of memory. Also, terminals on the channel do not respond to a parallel poll. As a result, the ADCC is always a slave and must be directly controlled by the CPU through the use of channel programs. Circuitry on the ADCC decodes the address information relating to channels and devices, and selects the correct device for operation.

The ports on the ADCC (Main and Extender) may be either hardwired to devices or to modems.

DSN/Intelligent Network Processor

The DSN/Intelligent Network Processor (INP) allows HP 3000 computers to be linked to other computers in a distributed data processing environment and supports multipoint terminals. The significant features of the DSN/ Intelligent Network Processor are listed in Table G-6.

Table G-6 DSN/Intelligent Network Processor Features

- 16-bit SOS microprocessor
- Data communications protocol handling
- Character handling and 32 kbytes buffer storage
- Modem and hardwired interfaces up to 56,000 bits/second
- Full- and half-duplex asynchronous modem support
- Bisync and HDLC/SDLC protocol compatible
- RS-232-C, RS-422, CCITT V.24 and V.35 interfacing
- Direct memory access for data
- Auto call capability

The INP microprocessor performs all of the communication data link protocol management, including: serialization, protocol management, frame/block management, and data buffering. This reduces CPU utilization and frees it to perform other tasks. Throughput is increased by overlapping the transfer of data already received from the communication channel with the processing and buffering of new data coming from the communication channel. The protocol driver may be dynamically changed. This allows the INP to be easily reconfigured from one data link protocol to another and permits several subsystems to use a single INP. An auto call capability is included with the INP. It allows a remote connection to occur in a dial-up environment without the intervention of a human operator.

Interrupt system

The interrupt system provides for up to 105 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 in the microprogram, which then sets up the interrupt processing environment and transfers control to the interrupt routine.

Interrupt routines operate on a common stack (interrup 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 20 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

The peripheral devices used on the HP 3000 Series 40 are connected primarily to GICs, while the ADCC is reserved solely for terminals and remote printers. Peripherals attached to GICs through the HP-IB include disc drives, line printers, and magnetic tape drives. For a complete configuration of the supported peripherals on the HP 3000 Series 40, refer to the current HP 3000 Configuration Guide.

Automatic restart after power failure

An integral part of the HP 3000 Series 40 is a power fail/ automatic restart capability. When the system AC line voltage falls below 90% of rated voltage, the system initiates a powerfail 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 up to 15 minutes, depending on memory size and battery condition.

The system is automatically restarted when power supply voltages reach 90% of their rated values and all register values are automatically restored and processing resumes.

Appendix H:
HP 3000 Series 40 and 44 Machine Instructions

Stack Op Instructions

ADAX	Add A to X
ADBX	Add B to X
ADD	Add A to B
ADXA	Add X to A
ADXB	Add X to B
AND	Logical AND of A and B
BTST	Test byte on TOS and set CC
CAB	Rotate A-B-C
CMP	Integer compare B, A and set CC
DADD	Double integer add D, C + B, A
DCMP	Double integer compare and set CC
DDEL	Double delete TOS
DDIV	Double integer divide
DDUP	Double duplicate TOS
DECA	Decrement A
DECB	Decrement B
DECX	Decrement X
DEL	Delete TOS
DELB	Delete B
DFLT	Float a double integer
DIV	Integer divide B by A
DIVL	Divide long integer C, B ÷ A
DMUL	Double integer multiply
DNEG	Double integer negate
DSUB	Double integer subtract D, C − B, A
DTST	Test double word on TOS and set CC
DUP	Duplicate TOS
DXCH	Double exchange
DZRO	Push double zero onto stack
FADD	Floating point add, D, C + B, A
FCMP	Floating point compare and set CC
FDIV	Floating point divide D, C ÷ B, A
FIXR	Fix and round
FIXT	Fix and truncate
FLT	Float an integer
FMPY	Floating point multiply
FNEG	Floating point negate
FSUB	Floating point subtract D,C − B,A
INCA	Increment A
INCB	Increment B
INCX	Increment X
LADD	Logical add A + B
LCMP	Logical compare B, A and set CC
LDIV	Logical divide C, B ÷ A
LDXA	Load X into A
LDXB	Load X into B
LMPY	Logical multiply B × A
LSUB	Logical subtract B − A
MPY	Multiply integers, integer product
MPYL	Multiply integers, long integer product
NEG	Integer negate
NOP	No operation
NOT	Logical complement TOS
OR	Logical OR of A, B
STAX	Store A into X
STBX	Store B into X
SUB	Integer subtract B − A
TEST	Test TOS and set CC
XAX	Exchange A and X
XBX	Exchange B and X
XCH	Exchange A and B
XOR	Logical exclusive OR of A, B
ZERO	Push integer zero onto stack
ZROB	Zero B
ZROX	Zero X

Shift Instructions

ASL	Arithmetic shift left
ASR	Arithmetic shift right
CSL	Circular shift left
CSR	Circular shift right
DASL	Double arithmetic shift left
DASR	Double arithmetic shift right
DCSL	Double circular shift left
DCSR	Double circular shift right
DLSL	Double logical shift left
DLSR	Double logical shift right
LSL	Logical shift left
LSR	Logical shift right
QASL	Quadruple arithmetic shift left
QASR	Quadruple arithmetic shift right
TASL	Triple arithmetic shift left
TASR	Triple arithmetic shift right
TNSL	Triple normalizing 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

Program Control and Special Instructions

DISP	Dispatch	**PCN**	Push CPU code (% 10)
EXIT	Exit from procedure	**PSDB**	Pseudo interrupt disable
HALT	Halt	**PSEB**	Pseudo interrupt enable
IXIT	Interrupt exit	**RSW**	Push cold load chan/dev
LLBL	Load label	***SBM**	Set Bank Mask
LLSH	Linked list search	**SCAL**	Subroutine call
PAUS	Pause, interruptable	**SXIT**	Exit from subroutine
PCAL	Procedure call	**XEQ**	Execute stack word

Machine and I/O Instructions

HIOP	Halt I/O program	**SCLR**	Set system clock limit
INIT	Initialize I/O channel	**SED**	Set enable/disable external interrupts
RCCR	Read system clock	**SMSK**	Set device mask
RIOC	Read I/O channel	**TOFF**	Hardware timer off
RMSK	Read device mask	**TON**	Hardware timer on
STRT	Programmatic warm start	**WIOC**	Write I/O channel
SIOP	Start I/O channel program	**MCS**	Memory controller read status
		DUMP	Programmatic dump

Loop Control Instructions

MTBA	Modify variable, text 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

*Series 44 only

Field and Bit Instructions

DPF	Deposit field, A bits to B	TCBC	Test and complement bit, set CC
EXF	Extract specified field, right-justify	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 add (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

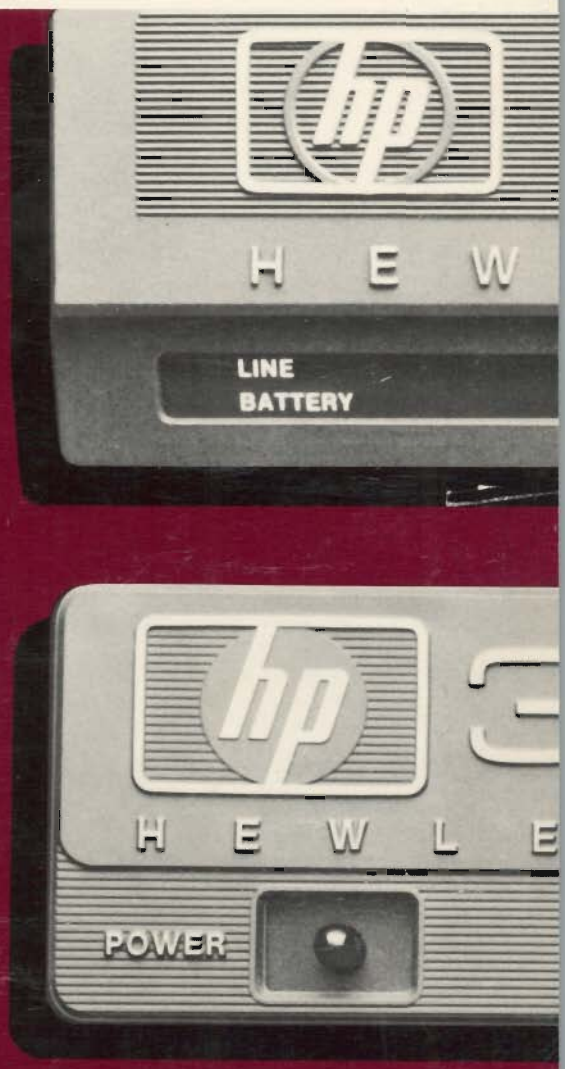
LDEA	Load double word from extended address	PSTA	Privileged store into absolute address
LSEA	Load single word from extended address	SDEA	Store double word into extended address
LST	Load from system table	SSEA	Store single word into extended address
PLDA	Privileged load from absolute address	SST	Store 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		



HEWLETT PACKARD

Corporate Headquarters:
Hewlett-Packard
1507 Page Mill Road
Palo Alto, CA 94304 USA

European Headquarters:
Hewlett-Packard S.A.
7, rue du Bois-du-Lan
P.O. Box
CH-1217 MEYRIN 2
Geneva, Switzerland

Intercontinental Headquarters:
Hewlett-Packard
3495 Deer Creek Rd.
Palo Alto, CA 94304 USA

Printed in USA 7/
5953-7462