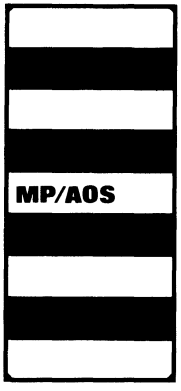
Chapter 6

THE MP/AOS OPERATING SYSTEM

An Advanced Operating System for
Real-Time, Multitasking, Multiprocessing
Execution and Development

David C. Cline
Data General Corporation



6

The MP/AOS operating system is the most recently introduced member of Data General Corporation's 16-bit operating system family. The system provides a real-time multitasking, multiprocessing environment for demanding, high-performance applications on ECLIPSE computers based on the 16-bit microECLIPSE microprocessor (as well as larger ECLIPSE systems). The system's design and comprehensive set of program development, run-time, and file management utilities make MP/AOS both an efficient real-time system and an extremely powerful and easy-to-use general-purpose system.

MP/AOS provides fast switching of tasks and processes, deterministic scheduling of tasks and processes by priority, low interrupt latency, and the ability to access more than 32K words (64K bytes) of memory from within a user process. In addition, the system supports bi-directional interprocess communication and memory segment sharing between multiple processes. Processes are memory resident and can be created and terminated dynamically. Each program can swap or chain to another program within each process. MP/AOS supports a wide range of application languages and utilities. Its file structure uses redundant pointers to ensure recoverability of data in all cases with the possible exception of the most catastrophic logical or physical damage.

Upward and downward compatibility with the large base of microNOVA, NOVA, and ECLIPSE software and hardware is a key feature. MP/AOS is a superset of MP/OS (a single process, multitasking system). MP/OS programs can be rebound to execute under MP/AOS. In addition, MP/OS programs may be developed under MP/AOS by means of a set of cross-development utilities. Many MP/AOS programs can be rebound to execute under AOS on the larger ECLIPSE computers; MP/AOS programs may be developed under AOS through an AOS-to-MP/AOS cross-development package. MP/OS programs developed under MP/AOS run on a wide range of microNOVA-based microcomputers and intelligent workstations, as well as on the NOVA 4 line of minicomputers.

Overview

MP/AOS extends the MP/OS operating system into multiprogramming environments. MP/AOS provides user management of memory beyond the 32K-word user address space, user access of data channel and Burst Multiplexer Channel maps, and I/O to unmapped memory. In this manner, the system provides powerful facilities that raise microprocessor-based systems to a new level of functionality. MP/AOS supports I/O and file system functions that are identical to the MP/OS functions (see Volume I for more details). Briefly, these functions include:

1) Device independent I/O operations. The I/O command set is flexible and optimized to reduce disk accessing limitations. (Often, accessing limitations can decrease an operating system's usefulness for real-time and general-purpose applications.)

2) A hierarchical file structure that allows access to a maximum of 512M bytes of data in a single seek.

3) A set of file access methods that includes multikey ISAM.

4) A file reliability system that allows all vital data structures to be rebuilt (unless the most catastrophic logical or physical damage occurs). This system is especially useful in commercial and real-time file-oriented environments.

MP/AOS runs on ECLIPSE systems based on the microECLIPSE chip set, the ECLIPSE S/20 (available in a 7 X 9 inch form-factor – compatible with existing MP/100 and MP/200 systems), and the ECLIPSE S/120 single board system (packaged in the Data General standard 15-inch NOVA/ECLIPSE chassis). The operating system also runs on the mid-to-large-size ECLIPSE S/130, S/140, S/250, and C350. The total MP/AOS environment is shown in Figure 6-1. MP/AOS requires the floating point and character set extensions to the ECLIPSE instruction set.

Optional in the S/20 ECLIPSE, and supported by MP/AOS, is a hardware floating point accelerator, which increases floating point performance by at least 300 percent. This accelerator achieves a Whetstone performance of 238K operations per second – roughly half the performance of an ECLIPSE S/140 with a hardware floating point unit.

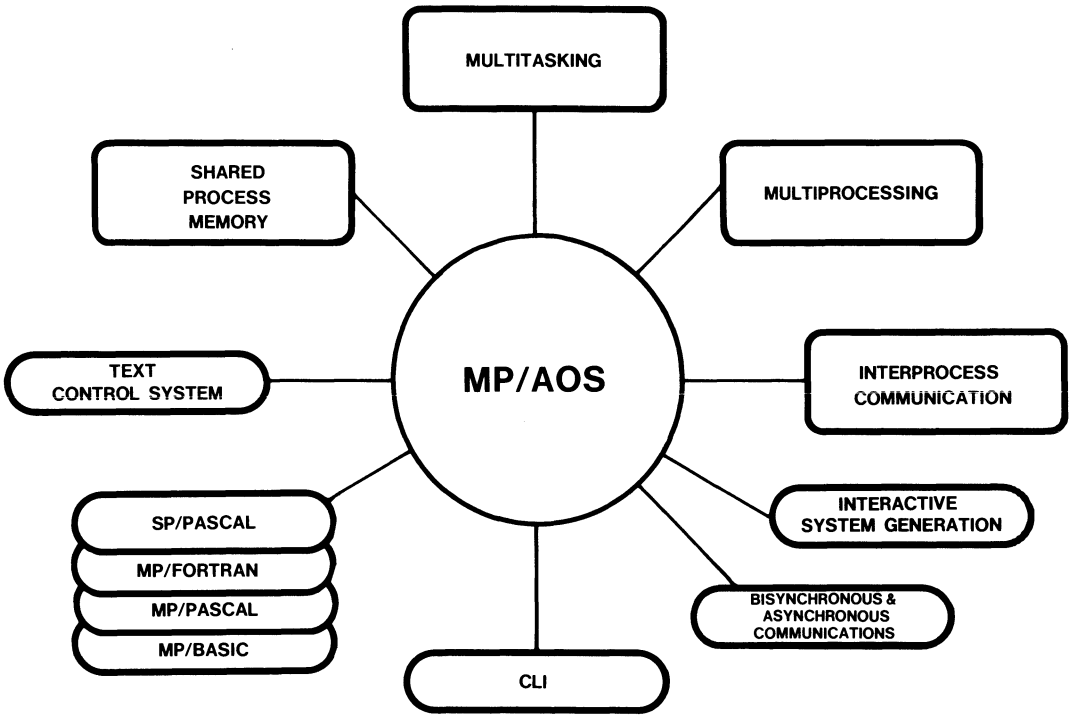


Figure 6-1 System Functionality of Data General's MP/AOS Operating System.

System Services and Support

MP/AOS supports the full ECLIPSE instruction set. This instruction set rivals the instruction sets of many mainframe computers by providing comprehensive bit, byte, and word manipulations as well as 32-bit (single-precision) and 64-bit (double-precision) floating point operations.

MP/AOS supports the ECLIPSE system real-time clock, asynchronous communications interface, and virtual console. MP/AOS also supports three standard Data General I/O protocols:

1) The microNOVA I/O bus - A 300K byte/second differentially driven bus that can be extended up to 100 feet by means of a simple cable.

2) The NOVA/ECLIPSE I/O bus - A bus that supports transfer rates up to 2M bytes/second.

3) A Burst Multiplexer Channel (BMC) - An I/O channel on larger ECLIPSE systems with transfer rates up to 10M bytes/second.

The microNOVA bus may be configured with any mix of the following disk drives:

1) 12.5M-byte and 25M-byte Winchester disk drives.

2) 315K-byte and 1.26M-byte diskette drives (one or two per controller). The diskette drives can also share a controller with the Winchester drives.

3) A cartridge disk drive with 5M bytes of removable storage and 5M bytes of fixed storage.

In addition, the system supports parallel printer ports, single line serial ports, and multiplexers (with 4 to 16 serial lines). MP/AOS serial interface drivers support dual asynchronous/synchronous ports, standard serial asynchronous devices (such as video display terminals and printers), and Xon/Xoff terminal protocols.

ECLIPSE Data Channel I/O supports all the previously described devices plus SMD-type disks (to 190M bytes each) and cartridge disks with 10M bytes of removable storage and 10M bytes of fixed storage.

Library support is available for both 800 and 1600 bpi tape drives, 600 and 900 lpm data channel printers, and synchronous communication channels. Library support allows programs to use these facilities without incurring memory overhead for device drivers that are not in use.

IDEF Facility

The system IDEF facility dynamically declares user-defined or library-defined device drivers. This facility permits user programs to handle interrupts from peripheral devices. The IDEF facility allows users to easily provide support for almost any non-standard device. This support is critical in many real-time environments. Additionally, users may assume control of a single I/O device on a multiplexer (e.g., one port of a 4-line mux can be used for synchronous communications while the remaining three ports are used for asynchronous communications).

Software Support

The following languages are supported under MP/AOS:

1) MP/Pascal - A threaded compiler implementation with conditional loading for minimal run-time overhead. MP/Pascal's language extensions include separate compilation modules for shareable routines and data, PROMable applications support, assembly language interface capabilities, dynamic string data types for text manipulation, multitasking support, and full access to MP/AOS system functions.

2) SP/Pascal - An extended system development version of Pascal. SP/Pascal contains all the features of MP/Pascal and additional extensions. These extensions include optimized machine code for the ECLIPSE architecture, exception handling and routine signaling, floating point hardware support (single- and double-precision), bit-level packed data storage, hardware character instruction set support for byte manipulation, structured constants for use as constants or literals, assembler-callable SP/Pascal procedures, and scalar expressions (in most constant contexts). SP/Pascal also provides a **whole** type with a 0 to 65535 subrange for full-word integers, ex-

tends the WITH statement for easier record copying, and provides a new **bytesize** function to set record and array sizes.

3) MP/FORTRAN IV - A multitasking and reentrant ANSI 1966 implementation that is transportable across all 16-bit Data General hardware and operating systems. MP/FORTRAN's formatted I/O capabilities are not generally available with other high-level languages. This compiler provides floating point hardware/firmware support and comprehensive file I/O facilities that interface to system file I/O and file management commands. MP/FORTRAN also provides extensive library functions, run-time format specification, task scheduling and coordination, quoted string and Hollerith constants, and full MP/AOS support for swapping, chaining, and overlaying portions of large programs.

4) MP/BASIC - An extended ANSI-standard implementation that includes integer data types, string dimensioning and concatenation operations, extensive string and mathematical functions, fixed and variable length file manipulation support, swapping, exception handling, and an assembly language interface.

5) Macroassembler (MASM) - A translator that converts one or more assemblylanguage source files into machine code. MASM supports relocatable code – both pure and impure. The assembler also includes a highly advanced macro capability that complements the diversity and flexibility of the ECLIPSE instruction set. MASM's sophisticated macro capabilities can be used to aid structured programming and to speed program development.

Utilities

MP/AOS provides a rich environment for sophisticated program development and for run-time file management. The system contains a comprehensive set of utilities; all MP/AOS utilities are compatible with the utilities that operate under the AOS and MP/OS operating systems. These utilities include:

1) A Relocating Linking Binder.

2) The Speed Text Editor.

3) The Library Editor.

The Relocating Linking Binder automatically relocates assembler object files and Pascal/FORTRAN compilation modules. The binder resolves external linkages, conditionally loads the proper system- and language-library routines, and structures and builds the appropriate overlay files. The utility also provides a set of useful tables and maps that detail the structure and location of the resulting modules.

The Speed text editor is a character-oriented text editor that provides file management, program development, and command line interpreter execution capabilities. Speed editing commands insert, delete and change characters, lines, or blocks of text. Commands are also available to search for specific words, strings, or patterns. A continuous display of the most recent updates is maintained. Command sequences and macros may be saved in buffers or files for re-execution. Thirty-six buffers provide workspaces for creating, modifying, and merging text files. In addition, cross-directory file operations are supported.

The Library Editor creates an object module library file consisting of one or more object modules or libraries. Individual modules may be programs supplied with the system (e.g., mathematical subroutines) or user-written modules processed by one of the MP/AOS compilers or by the macroassembler. The replace, insert, and delete functions are also supported.

Text Control System

MP/AOS includes a state-of-the-art Text Control System (TCS) that provides a comprehensive and integrated set of text processing facilities. These facilities have never before been available on a system of this size. The TCS facilities support source file management in a manner that guarantees maximum productivity and control. To support file management, TCS maintains historical records of code modifications to program source modules. These records include the name of the user who made each text modification, when the modifications were made, and why the modifications were made. This control system allows only one update per source file version to be in progress at a given time. In addition, the system significantly reduces space requirements by logging only the source file changes when a

module is checked back in. In this manner, any revision can be re-created.

The **Build** facility uses file system information as well as the information maintained by TCS to choose the correct modules for a particular software revision and to create a complete user software package. Build recognizes when individual modules have not been changed since the last compilation/assembly. By recognizing unchanged modules, Build minimizes the work required to construct the software package. Build tracks the relationship between modules as a directed graph. If only one source module changes, then only the product segments that depend on the changed source module are rebuilt. As a project management tool, Build can supply reports that detail the steps required to construct a software package – without actually generating the package.

The TCS screen-oriented editor (**Slate**) encompasses many text processing and programming functions in one consistent user interface that is similar to the Command Line Interpreter. Slate includes numerous user-oriented functions such as an integral "help" facility (to supply command and usage information) and variable tab/margin settings. Slate provides cursor-controlled screen manipulation and accepts command sequences from a command file or from the terminal. Editing functions are easily tailored through function key definitions. Also, multiple files/buffers can be manipulated concurrently for sophisticated editing and merging of files.

TCS also includes a **Find** facility that scans text modules for occurrences of complex, user-specified patterns.

File Management Utilities

The file management utilities running under MP/AOS are extremely powerful. In addition to an ISAM file library for managing multikey indices and data record files, these utilities include:

1) A file editor (FEDIT) - FEDIT allows a user to examine and modify the files of an MP/AOS system – especially program files and data files. Addresses as well as data values may be displayed and modified.

2) A file display and comparison utility (FDISP) - FDISP lists the contents of a file (including addresses and data) in readable form. When used to compare files, this

utility lists only those portions of the files that are at variance.

3) A source compare program (SCMP) - SCMP compares two source programs (line-by-line) and enumerates the changes necessary to convert the first file into the second file.

Other MP/AOS utilities provide a file sort/merge package, a printer spooler, and a copy facility that transfers files from one directory to another (or to tape). The utilities also support file transfers over inexpensive asynchronous communication links from an MP/AOS system to another MP/AOS system, to an MP/OS system, or to an AOS system. In addition, utilities are available that support manipulation of MP/AOS and MP/OS disk files on an AOS system.

User Interface

The Command Line Interpreter (CLI) is an English language, free-form interpreter, with an on-line "help" feature that provides general documentation. The CLI provides direct access to many MP/AOS facilities and utilities. Moreover, the CLI provides macro and textual substitution facilities in addition to built-in commands and pseudo-macros. These facilities simplify and customize the operator and system environment. CLI commands provide control of I/O and disk functions (type, copy, locate, delete, mount, and dismount) and support full access to the disk file structure (disk status, file status, directory access).

The Command Line Interpreter allows commands to be abbreviated to their shortest, unique subset. The CLI provides macro facilities for such purposes as creating program pipelines (typically load-and-go sequences). In addition to controlling I/O and peripheral functions, the CLI also allows switches and parameters for sorting, file searches, and the other options previously described.

When booted, MP/AOS normally executes the Command Line Interpreter, which contains a user-defined log-on macro. This macro may contain a user-generated password/security checking program that ensures authorized system use. The log-on macro may also be used to initialize the user environment (including protection attributes) and to automatically execute user programs. In this manner, the log-on macro can be used to eliminate operator intervention at system start-up.

Diagnostics

In addition to the extensive system diagnostics provided by ECLIPSE firmware to ensure memory, I/O, and processor integrity, MP/AOS offers many built-in diagnostic features. MP/AOS interfaces with the hardware error detection facilities that monitor memory, disk, and other errors. The system can display these errors on the primary console and log the errors to the system disk. The Error Logger utility can subsequently be used to review these errors. In combination, the diagnostics help to isolate hardware modules that may require field service. In this manner, the diagnostics guarantee maximum system availability and prevent module failure in critical environments. Because all applications do not require this diagnostic information, the diagnostic facility may be discontinued at the user's option.

Debugging and Process Optimization Tools

MP/AOS also supports an extensive set of utilities, some running as separate processes, to debug and optimize real-time processes. These utilities are described in the following paragraphs.

The **Process Debugger** (FLIT) is an interactive system utility that provides access to a task's memory locations, registers, and status information. Requiring no special program handling or modifications, FLIT allows users to:

1) Examine task status and modify task control statements.

2) Set conditional and unconditional breakpoints.

3) Alter display and addressing modes.

4) Define macros, text files, and single keys for tracing, and returning from, pointer chains.

The **Profiler** determines a probabilistic execution profile of a program in order to identify possible bottlenecks and optimize program execution speed. This utility measures a program's run-time performance and reports total execution time, system time, and process time. The profiler can also identify partial process elapsed time by specified symbolic location ranges, rather than by octal memory addresses.

The **Process Monitor** (OPM) produces a dynamic graphic display of

system resource allocation and status. This display includes a bar graph that shows the percentage of processor time used by user processes, by the operating system, and by the system idle loop. OPM also provides information on active and current processes, I/O block allocation, and individual user processes. This information includes process status, priority, processor time, and I/O block count, as well as performance ratios that are useful for maximizing system throughput.

The **Error Logger** processes the system log file to produce a history of time-and-date information for any hardware errors or soft main memory errors.

System Generation

MP/AOS's interactive system generation is designed for tight control of the system environment, minimal system space, and maximum process speed. The MP/AOS Sysgen procedure defines the system's peripheral configuration, the maximum number of processes, tasks, and interprocess communications buffers, and other system parameters. (For example, if only four processes are required, space is reserved for only four process control tables.) By allocating space during system generation, no execution time is required to allocate and initialize data structures. This static allocation technique allows faster process execution and more efficient memory use. In addition, no processes are required for "garbage collection" since the Sysgen facility maintains memory allocation integrity and efficiency.

Interrupts

MP/AOS is optimized for the real-time handling of interrupts. System resources are allocated at system generation (to avoid run-time delays), base I/O drivers are located in unmapped memory, and all user processes and tasks are scheduled by user priorities. MP/AOS also provides unpended I/O, which allows a task to continue processing concurrently with that task's I/O operations. (This capability can be a useful variation of multitasking support.)

The real-time response latency of MP/AOS depends on the previously mentioned system characteristics and on the worst case "interrupt/reschedule" disable path in a user's program. This latency is easily estimated and measured.

Communications

In addition to the bisynchronous communications library, MP/RJE80 (2780/3780) and MP/3270 are available.

Process Management

Due to the emphasis placed on real-time environments, MP/AOS does not treat multiple processes in a hierarchical manner. Processes created in parallel execute in separate address spaces and, once initiated, bear no further relationship to the processes that created them. All concurrent processes are memory-resident to increase execution speed.

Process priority is strictly user-defined. Processes, each referenced by their own unique system-assigned process identifier, can communicate with each other and control scheduling. Program nesting within the address space of a single process is supported to eight levels. (A parent program is reactivated when the next-lower program terminates.) A process can terminate itself and terminate other processes. A break file can be created by a terminated program (for later examination). In addition, a message can be passed by a terminated program to its parent program.

Task Management

A user process consists of one or more tasks that reside in the process address space. Since MP/AOS uses some memory for task control information, a maximum number of tasks must be specified when a process is initialized. Each process can support a maximum of 255 tasks (including system tasks).

As with process priority, task priority is strictly user-defined and each task is referenced by a unique task identifier. Task priority is subordinate to process priority. In this manner, tasks within low priority processes are run only if no higher priority process/task is ready to run. Tasks can modify their own priorities and the priorities of other tasks. Tasks can also control each other's activities and synchronize activities (to respond to external interrupts and to exchange messages).

Memory and Resource Management

MP/AOS divides system memory into unmapped and mapped portions in order to optimize system and interrupt performance (Figure 6-2). The unmapped portion is the kernel, which occupies the lower portion of memory. The kernel may occupy up to 31K words of memory, depending on the type of system generated. The kernel contains the interrupt drivers, the scheduler, and the major system data structures and associated support routines. The mapped portions consist of the supervisor and the disk buffer and overlay buffer areas. The supervisor program is divided into overlays to further reduce the total physical memory requirements. The supervisor frequently calls the kernel to manipulate kernel-managed data structures.

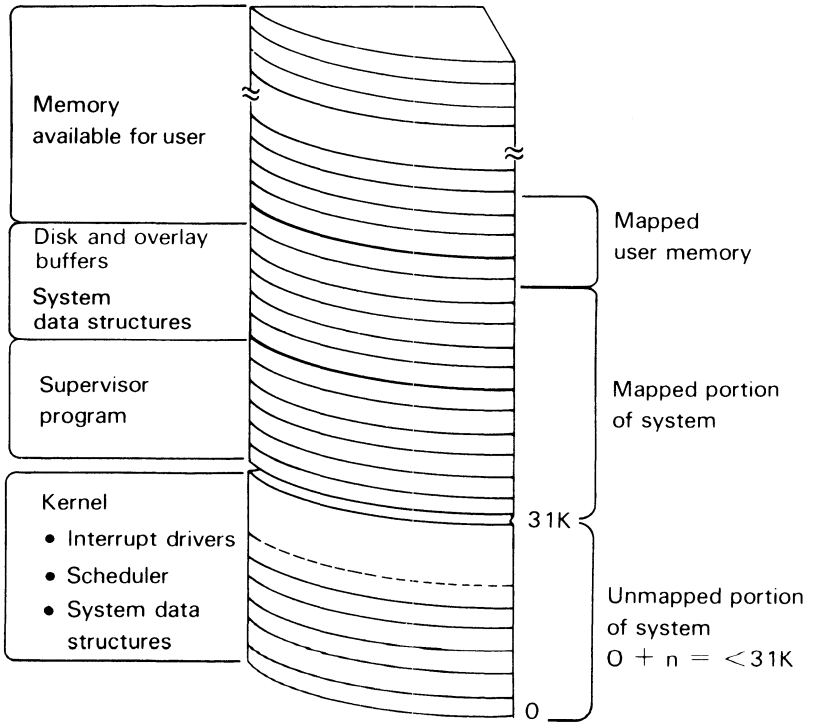


Figure 6-2 System Memory Partitioning.

The disk and overlay buffer areas are dynamically managed by the operating system. The disk buffer cache increases the efficiency of data transfers and minimizes disk accesses. The buffer cache is maintained by means of a least-recently-used (LRU) replacement algorithm. Normally, disk data currently in memory are not restored to disk until the cache is needed for a different block of data. Optionally, the buffers may be flushed before returning to the user program. This action guarantees that all file system and user data associated with an I/O channel are written to disk before the user program resumes execution. The buffering mechanism is bypassed for block-aligned data, which is moved directly from the device to the user address space.

In general, MP/AOS allocates resources statically whenever possible. With this allocation philosophy, resource needs can be tailored to specific applications and, once tailored, the system can be optimized for execution speed.

Memory Segments

An MP/AOS user manages memory by means of system calls. These system calls manipulate regions called **segments.** A segment is one or more 1024K-word pages of memory (up to the maximum memory accessible by a user process). Each user process has a maximum logical address space of 32K words, consisting of **impure** and **pure** memory (Figure 6-3). The pure memory area may be divided into shared and overlay memory. The actual amount of logical address space for each process is specified by the user during creation of the process. This memory need not be physically contiguous.

Process memory is allocated in three segments, which are locally numbered:

1) Segment 0 - Impure area.

2) Segment 1 - Shared area (Pure).

3) Segment 2 - Overlay area.

The shared segment is automatically shared in a manner that is transparent to the user. Shared and overlay segments are also write protected. The impure area may grow and shrink. The maximum allowable

size of the impure area is determined by subtracting the fixed-size shared and overlay segments from the 32K-word logical address space.

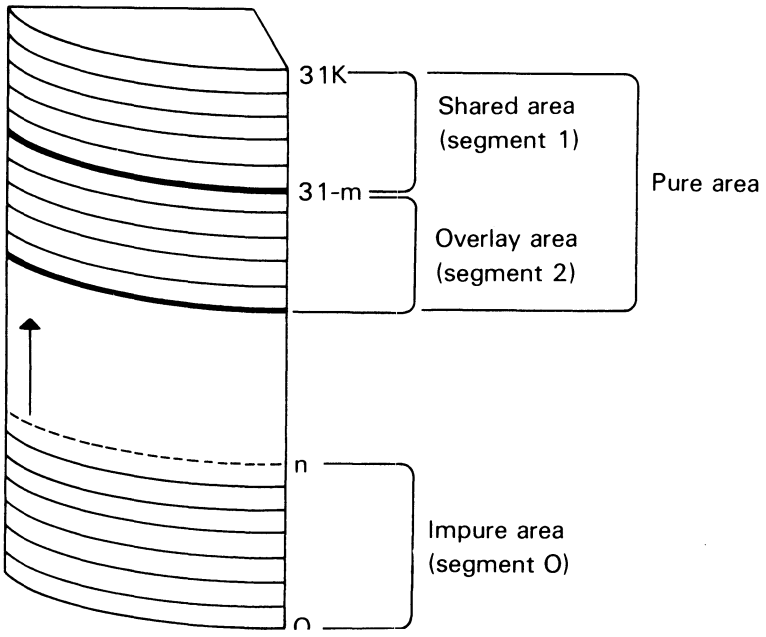


Figure 6-3 Pure and Impure Memory Segments.

User processes can "window" their logical address space anywhere within a valid memory segment. This windowing ability vastly enlarges the memory resources available to a user process and is subject only to physical memory limitations. A process may, for example, define one or more additional memory segments of any size, and attach these segments to the process' address space. The process can subsequently release the segments at any time (Figure 6-4). Since any number of processes may attach a segment once the segment has been created, data may be passed between processes in an efficient manner. A second useful feature is the ability to directly transfer data between a device and a memory segment regardless of whether the segment is mapped to a user address space.

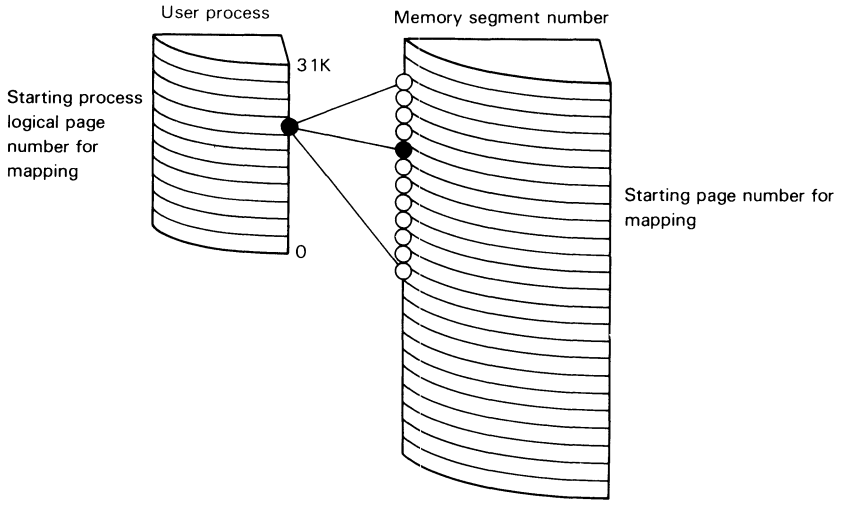


Figure 6-4 Attaching Segments to a Process' Address Space.

Memory Translation and Protection

The ECLIPSE hardware map feature translates a 15-bit address into a 20-bit address. To perform this translation, the address translation hardware uses the logical-to-physical translation functions set up by the supervisor. MP/AOS supports seven maps: two for transparent user address translation, four for the data channel, and one for the BMC (burst multiplexer channel in larger ECLIPSE systems).

Beyond translating addresses, the mapping function performs five protection functions:

1) Validity protection - Protects unused portions of the user process' logical address space.
2) Write protection - Protects against unauthorized data changes.
3) Indirect protection - Limits an indirection chain to 16 levels.
4) I/O protection - Controls access to I/O devices.
5) Data channel protection - Write protects a channel's logical address space.

I/O Devices and the File System

MP/AOS provides simple, yet efficient, methods by which programs can communicate with input/output devices and store/retrieve file data. All disk devices, character devices, and files are handled by the same system calls. This operating system support facilitates the writing of transportable, device-independent programs. The set of I/O commands includes **open, close, read, write,** and **position.** The open command associates one of up to 256 logical user channels with a physical device or with a disk file. Further read/write commands reference these logical channels – eliminating file name look-ups. Channels, and hence files, can be passed transparently across program invocations. For a more detailed discussion of this technique's flexibility, refer to the discussion of MP/OS in Volume I.

MP/AOS I/O capabilities include dynamic, block-aligned, and data sensitive transfers, as well as direct segment I/O. Dynamic I/O permits a specified number of bytes to be read or written. Data sensitive I/O transfers data until a specified maximum byte count is reached or until a predefined delimiter character is encountered. The delimiter may be one of the system defaults (CR, NL, FF, NUL) or user-defined. Direct segment I/O allows either of the previous modes to be used (i.e., data bytes may be transferred by count or until a delimiter is met). With direct segment I/O, however, the memory segment does not have to be mapped into the user address space.

MP/AOS divides I/O devices into block-structured devices (e.g., disks) and non-block-structured devices (e.g., consoles and printers). Disks are initialized and released by user calls to permit the orderly introduction of new media. MP/AOS runs consistency checks on block-structured devices during initialization and automatically executes the FIXUP program, if necessary. Opening a disk device without first initializing the device causes the disk to be accessed as a file (with all elements contiguous).

Terminals are treated as two different devices: the keyboard (for input) and the CRT or printer (for output). The console interface allows console characteristics to be displayed and modified by system calls. These characteristics allow users to make use of standard I/O facilities without explicit programming. For example, characters received from the keyboard may be automatically echoed to the CRT. Also, a form feed, which causes many CRTs to erase the screen, can be echoed as another character (e.g., a line feed). MP/AOS handles line printers in the same manner as console output devices.

MP/AOS supports a number of predefined control sequences that allow terminals to pass an interrupt to a program, terminate a program, and save the program state in a save file. In addition, a control sequence is available that interrupts a program's execution and transfers control to the debugger.

File Structure

The major features of the MP/AOS file system are identical to those of MP/OS (described in Volume I). These features include:

1) Multiple levels of directory nesting (for easy file organization) and directory search by pathname.
2) Contiguous file elements combined with a hierarchal index structure for random access within files of varying sizes.
3) Dynamic file creation and deletion.
4) Link files to simplify file referencing.
5) Search lists to facilitate and streamline access to multiple directories.
6) File protection through user-defined and system-defined attributes, including write- and access-protection.

The MP/AOS I/O and file structure is organized as a hierarchy of directories. The root node of the file structure is a table of system devices. These devices include disk and diskette devices that may contain further directory structures. Directories may contain either data or program files and may be nested to any level. The full name of a file is its **path name.** This directory structure supports separate development and operation directories. With this structure, two versions of a program can coexist and be protected from each other.

The primitive file allocation unit is an **element** that ranges from 512 bytes to 16M bytes in length. An element is a contiguous segment of disk space. A file may consist of multiple elements and an index block of element pointers. These element pointers allow the elements of a file to be allocated anywhere on a disk. Moreover, the index blocks may also be indexed, resulting in file structures as large as 2048M bytes. With this scheme, there is virtually no limit to the physical size of the disks that are supported under the file structure. Furthermore, since file allocation occurs at run-time, files are easily extended on a dynamic basis, with minimal dependence on locating contiguous disk sectors.

The advantage of this contiguous file structure is the ability to rapidly access a random position in a file with a single seek (no indices need be searched and no linked lists are traversed). The file descriptor maintaining a set of file elements may be kept in memory in order to eliminate a seek operation when accessing data elements or index blocks. This technique allows access to a maximum of 512M bytes in a single seek.

File access is controlled with a standard read call and a random file position call. The random position call allows any byte within a 32-bit address space to be directly accessed.

File Reliability

Beyond the file organization and the ability to assign access control attributes (read-only, write-only, permanent, and attribute protect), a number of advanced reliability concepts are employed by MP/AOS. First, key disk structures are located through indirect label vectors. This technique ensures minimal dependence on the error-free quality of specific disk sectors. Critical structures are also duplicated and checksummed so that errors are easily detected and corrected. Second, the directory structure is a logical association, not a physical tree. If a directory index is destroyed, the referenced files are not lost, and the directory can be rebuilt from file descriptor information. Third, the allocation of sectors is based on a bit map of free sectors and this bit map can be recreated by the FIXUP utility. Finally, a mechanism is provided for correcting disk structures in the event of a system failure. The FIXUP utility is run automatically if the bootstrap program determines that the disk was not properly dismounted. This utility ensures that the system will access only correct disks. This feature is especially useful in the fixed-disk environment of most microcomputers. The FIXUP utility automatically:

1) Validates all key structures and rebuilds damaged ones (even if both copies are damaged).

2) Checks all directories and rebuilds damaged directories.

3) Checks all file index structures to ensure that the structures point to valid (allocated) blocks and to ensure that these blocks have not been allocated in duplicate.

4) Rebuilds the bit map if necessary.

Summary

MP/AOS provides one of the most complete program development and execution environments available on systems of any size. MP/AOS is a real-time multiprocessing, multitasking system that runs on a range of systems from microprocessor-based single board computers to high-end minicomputers supporting 16 users. The operating system design and the comprehensive set of run-time, program development, and file management utilities make MP/AOS both an efficient real-time system and an extremely powerful and easy-to-use general-purpose system.

David C. Cline is manager of Data General Corporation's Small Systems Software Department, which is responsible for the design of high-performance real-time operating systems, languages, and utilities. Previously, he worked at Chi Corporation. He has a B.S. and M.S. degree in Computer Science from Bowling Green University, Bowling Green, Ohio.