EXTENDED OPERATING SYSTEM (EOS)

OVERVIEW

SOFTWARE SUBSYSTEM DESCRIPTION

3A PROCESSOR EXTENDED OPERATING SYSTEM

CONTENTS PAGE

1. GENERAL 1
A. Operating System General Concepts . 2
B. Extended Operating System Structure 2
C. EOS Kernel 3
D. EOS Tasks 3
E. Functional Structure 3
2. PROCESSOR/PROCESS MANAGEMENT . 4
A. Process/Task Creation 4
B. Processor Management 4
C. Event Management 5
3. MEMORY MANAGEMENT 6
A. Memory Protection 6
B. Memory Organization 6
C. Paging 7
4. DEVICE MANAGEMENT 7
A. Channel Management 7
B. Device Management 7
C. Terminal Administration 8
D. EOS Commands 8

CONTENTS PAGE

E. Data Administration 8
5. INFORMATION MANAGEMENT 8
A. File System 8
B. Tape Cartridge Management . . 9
C. Patching and Overwrite 9
6. MAINTENANCE 9
A. General 9
B. EOS Audits 9
C. Utilities 10
7. EOS FUNCTIONAL INTERFACE 11
8. GLOSSARY 11

Figures

1. EOS Functional Structure 12
2. EOS Kernel Hierarchy 13
3. EOS Call Structure 13
4. EOS Process States 14

1. GENERAL

1.01 This section provides a system overview of the Extended Operating System (EOS) which

executes on the 3A Central Control (3A CC). The EOS is a general purpose operating system designed to support real-time, multitasking applications. It supports applications by acting as a resource manager of the processor, the peripheral devices, memory, and data base. The EOS supports system reliability through duplication and provides the basic maintenance structure in the form of automatic reconfiguration, initialization, audits, and diagnostics.

1.02 Whenever this section is reissued, the reason(s) for reissue will be listed in this paragraph.

1.03 The following series of Bell System Practices provide the functional descriptions of the EOS:

(a) Memory Management, Section 254-340-014

(b) Processor/Process Management, Sections 254-340-030 and 254-340-031

(c) Data Management, Sections 254-340-052 through 254-340-064

(d) Maintenance, Sections 254-340-080 through 254-340-090

(e) Support, Sections 254-340-100 through 254-340-106.

1.04 The program listings (PRs) for the EOS are numbered according to the following assignment:

(a) Kernel Functions—4C104, 4C113, 4C129, 4C153, and 4C141 through 4C158

(b) Input/Output (I/O)—4C201 through 4C234

(c) Message and Commands—4C301 through 4C313

(d) Resident Application TTY Tables—4C401 and 4C402

(e) System Utilities—4C501 through 4C506

(f) System Maintenance—4C601 through 4C626

(g) Diagnostics—4C701 through 4C709

(h) Overwrite Function—4C801 and 4C802

(i) Common Systems—1C910 through 1C935.

The input and output manuals for the EOS are numbered IM-4C001-01 and OM-4C001-01. The trouble locating manuals are TLM 4C702 through 4C709.

A. Operating System General Concepts

1.05 A general purpose operating system provides a set of procedures that enables users to efficiently share a processor installation. Users compete for and cooperate in using physical resources such as processor time, storage space, and peripheral devices. The operating system frees users from machine details such as physical characteristics and protocol of peripheral devices, memory structure, etc. An operating system provides services which are a level above the physical machine and its basic instruction set. These services include memory management, process scheduling, interrupt handling, file system management, and system maintenance.

1.06 The various operating system services can be provided either by the system or through the medium of system calls from a user. Thus an operating system provides a set of services to other software that can be used as if the services are actually provided by the basic machine. This type of structure therefore augments a virtual machine on the basic physical machine. The EOS can thus be viewed as providing a series of virtual machines, each of which provides some basic management function.

B. Extended Operating System Structure

1.07 The extended operating system supports applications by acting as a resource manager of the processor, the peripheral devices, the memory, and the information data base. It is controlled by the application user through the medium of system calls from the user to the EOS. System calls are initiated through use of EOS system macros which are listed and described in Section 254-340-106.

1.08 The extended operating system is structured (Fig. 1) in two divisions: the kernel level and the system process/task level. The kernel performs the real-time or function critical elements of the operating system. Process/task level system functions are asynchronous cooperating processes/tasks. The user structures the application system as a series of processes/tasks; thus, the EOS and applications processes/tasks are processed exactly

the same by the kernel. The only difference is that the EOS processes/tasks are functionally part of the EOS. There is also a structural and functional distinction between processes and tasks.

(a) A process is the execution of a series of programs whose sequence of execution is specified by a file of commands. Each program contains all instructions and data necessary for the performance of the command. On completion of one command, the command interpreter reads and initiates execution of the next command. When the last command has been executed, the process is terminated and all resources are freed. Processes are defined in EOS by process descriptors.

(b) A task is an execution module with all necessary information to allow asynchronous execution. Like a process, a task is defined in EOS by a task descriptor which specifies the resources to be allocated to the task. Unlike a process, a task has no command file. When a task terminates, all resources are freed and the task ceases to exist until required again, at which time it is rebuilt from the task descriptor.

Processes and tasks are defined by an application through appropriate EOS system macros and are listed in an operating system table which is in program listing form and becomes a part of the application generic. Details of process and task management and control are provided in Section 254-340-030.

C. EOS Kernel

1.09 The kernel contains all real-time functions of the operating system and all functions required for the implementation of tasks. It also contains some maintenance functions to control restart of a task and some initialization functions. The main functions of the kernel are:

- Dispatcher
- Interrupter Handler
- Timer
- Intertask Communicator.

The functions are arranged in a hierarchy (Fig. 2) with the functions toward the center representing the highest level. The structure is such that calls from one level to another can only be made inward. For example, the interrupt handler can call the dispatcher but the dispatcher cannot call the interrupt handler. A call may skip levels, eg, the intertask communicator can call the dispatcher directly, bypassing both the timer and interrupt handler levels.

1.10 The kernel functions are requested by EOS or application processes/tasks via appropriate EOS system calls which generate various service requests for the kernel functions.

D. EOS Tasks

1.11 The EOS provides other services which are implemented as asynchronous tasks.

- File system/device handlers
- Open/close functions
- Attach/detach functions
- Access methods
- Terminal administration
- Process creation/termination
- Command interpretation
- Maintenance.

These tasks call upon the kernel and communicate through the kernel as do any tasks running under the operating system.

E. Functional Structure

1.12 The basic function of EOS is to act as a resource manager and utilize the available resources in the most effective manner. It accomplishes this by a systematic scheduling of the various EOS and application processes and/or tasks which comprise a particular application system. In this context, the EOS processing can be described by a functional organization:

- Processor/process management
- Memory management/organization

- Device management
- Information management
- Maintenance
- Interrupt management
- System utilities.

1.13 Since most of these EOS functions and all applications functions are accomplished by processes/tasks, timely scheduling of the proper processes/tasks is the basic operation of the EOS. The scheduling process is accomplished by the kernel which responds to requests for service from either the EOS or application tasks (Fig. 3). The requests can be in the form of interrupts, either timed or demand, or system calls from the various processes/tasks which request a service of some type. The response to these calls basically requires scheduling and changing state of a process/task and is predicated on several basic features of the EOS.

(a) Each process or task is assigned a priority between 0 through 255, with 255 being highest. The application user assigns the priorities at system generation, assigning task priority in order of importance.

(b) An ordered list of readied processes/tasks (ready list) is dynamically maintained in priority order.

(c) The highest priority process/task on the ready list is always dispatched first.

(d) Once a process/task is executing in the RUNNING state, it will run until:

(1) It goes into the WAIT state

(2) It completes and goes into the COMPLETE state

(3) It is interrupted by a higher priority task and placed in the INTERRUPTED state.

A general description of the major functions is given in subsequent parts of this section with more detailed descriptions in the appropriate sections referenced in paragraph 1.03.

2. PROCESSOR/PROCESS MANAGEMENT

2.01 Process/task management consists of several primary functions. These are the creation and termination of tasks, allocating the processor to tasks which are ready to execute, and coordinating tasks and intertask communication (event management).

A. Process/Task Creation

2.02 Tasks are initially created, as specified by the application, at system generation time and stored in memory in the form of process or task descriptors. Tasks may be activated either at system generation or at some later time by another executing task or process.

2.03 When a command file initiation request is received (process) or some interrupt or event occurs (task), the process or task is constructed from information contained in the descriptor. The descriptor describes the task identification, priority, storage, and device requirements, etc, and resources are allocated based on this data. This activity, in turn, requires the services of other tasks. The descriptor information is then retrieved from dynamic memory and linked into the ready list in order of task priority.

2.04 When the task or process completes, a reverse process takes place, the resources are freed, and the description is cleared from dynamic memory. The task in a sense ceases to exist until another request is received.

B. Processor Management

2.05 Processor management is the procedure which determines which process or task will have control of the processor. The basic mechanism is based on process state transition.

2.06 Only one process/task can be executing (in the RUNNING state) at one time. All other tasks in the system are in one of several states (Fig. 4). These states are: INACTIVE, HOLD, READY, RUNNING, SUSPEND, WAIT, INTERRUPTED, or COMPLETE. One function of the operating system is to handle the logical transition of a task from state to state. The various states are defined as follows:

- INACTIVE—The state of a nonexecuting process.

- HOLD—Upon activation, a process enters this state while attempting to reserve all required resources before execution starts.

- READY—All conditions necessary for execution have been satisfied but the processor is not available. READY state processes are queued by priority for assignment by the dispatcher.

- RUNNING—The process state having processor allocation to execute the running task.

- SUSPEND—The state when a process or task has been suspended by its parent process.

- WAIT—The process is suspended pending the completion of some asynchronous event.

- INTERRUPTED—When an interrupt occurs, the currently executing process is halted temporarily and placed in this state while the interrupt handler is executed. Upon completion of the interrupt, the interrupted process will either be returned to the RUNNING state or, if the interrupt readied a higher priority process, will be returned to the ready queue.

- COMPLETE—Upon process completion, the operating system must delete various table entries and perform any necessary post-processing, eg, statistics recording. During this period, the process will be in this state.

2.07 Various mechanisms which can initiate state transition are as follows:

- Hardware interrupts which may require some immediate action on the part of the system and precipitate the preemption of the currently executing task and creation of a new task.

- The event mechanism is used to provide synchronization of tasks. A task can wait for the occurrence of an event and, when the event does occur, the task will resume execution.

- For tasks which must be run at specified time intervals, a timer interrupt will cause movement from state to state.

- A task may be moved to the WAIT state for an I/O completion or be suspended by another task.

- The normal activation and termination of a task.

2.08 The major state transition involved in processor management is the movement of a task from the READY state to the RUNNING state. When a task is moved to the READY state, it is placed on the ready list in order of its priority. The determination of the task to be moved to the RUNNING state is based solely on which task on the ready list has the highest priority. The priorities are determined statistically and assigned upon creation of the task; however, the priority of a task may be altered dynamically by the system or the task itself during operation.

C. Event Management

2.09 An event is a signal to a process or task that a previously specified state change has occurred. Each process and task has a unique set of 32 event flags, 32 event mask bits, and the ability to link a subroutine, called an event routine, to each event. Events are used to:

(1) Signal the completion of another process or task

(2) Signal the completion of an I/O operation

(3) Signal the completion of a specific time interval

(4) Signal the receipt of an interprocess message

(5) Signal the occurrence of certain maintenance actions.

2.10 Events permit EOS to coordinate the execution of a number of processes or tasks and to overlap input or output operations with the execution of other statements in the process or task that initiated the operations. Events are defined in each task requiring them, based on certain criteria: 12 of the 32 event flags are reserved for EOS use, thus 20 event flags are available for each

process and task. The event flags reserved for EOS use are:

Flag(s)	Use
0, 1, 2, 3	Maintenance
4	File System
5	Reply Received
6	Request Received
9	I/O Error
7, 8, 10, 11	Unassigned.

2.11 The EOS is event-driven and, once a process relinquishes control of the processor, it can be rescheduled for execution only as a result of the occurrence of an event. When an event occurs, its associated process is readied for execution. Next, the application process indicates whether it requires notification by EOS that an event has occurred or whether it will schedule its own detection of events.

2.12 The EOS notifies the process that an event has occurred via the use of event routines. Event routines are subroutines which are linked to event flags. When a readied process is dispatched by EOS, the local dispatcher assumes control if any event flags are set.

2.13 Each process or task also has a 32-bit MASK register which permits the enabling or disabling of event routines. A mask bit of zero means the corresponding event routine is disabled, and a mask bit of one means the corresponding event routine is enabled.

2.14 After all event flags have been processed by the local dispatcher, the EOS returns control to the point in the program where the preemption occurred in the process or task. The local dispatcher controls the flow of control to the event routines. The functional details and major program interface for processor/process management are in Section 254-340-030.

3. MEMORY MANAGEMENT

3.01 The EOS memory management is the function which is concerned with controlling the resources of the processor main memory main store (MAS). The function is composed of three basic activities: protection, organization, and paging.

A. Memory Protection

3.02 Main memory can be write-protected in blocks of 4096 (4K) words. Registers in the memory controller are used to designate which 4K blocks of memory are to be write-protected. The EOS specifies write-protection based on the write-protect control table which is located in the operating system tables. The control table provides a map over main memory when complete. Each word of the table represents two 32K-word memory modules, with each bit representing a 4K block of memory. A block of memory is write-protected when its corresponding bit is set to "1". For example, bit 0 of word 0 represents the first 4K block; when this bit is set, words 0 through 4095 in MAS are write-protected.

3.03 Which blocks to write-protect are left to the application with the following exceptions.

(a) Because the interrupt transfer vector originates at address 0, the first 4K block of memory must always be protected.

(b) The second 4K block is never write-protected since it contains the location of the paging buffer and other variable data areas.

B. Memory Organization

3.04 The basic EOS memory organization consists of two primary types: static (or fixed) and dynamic.

3.05 The fixed structure is located in the first 8K of memory and is divided between the write-protected first 4K and writable second 4K.

(a) The first 4K contains the interrupt transfer vector, maintenance transfer vector, system transfer vector, and the read-only kernel programs.

(b) The second 4K is writable and contains the hold-get stack, system variables, and the paging buffer.

These areas are defined in the operating system table, which represents the basic source of information for system generation.

3.06 The dynamic area can be located anywhere within the first 64K block of memory. All EOS data structures are allocated in this area, and the area may be dynamically allocated during run time. Dynamic memory contains various types of data items, eg, task descriptors, local file tables, timers, etc, allocated by specific block types.

C. Paging

3.07 Certain programs and data tables which are infrequently used are stored on the cartridge tape. These programs are normally those which are not used at any specific interval but must be available on demand. In the EOS, all diagnostic programs are of this type and are stored off-line.

3.08 When these programs are required (normally via a manual request), they must be read from the tape into the processor in order to execute. The process of reading in the programs is called paging and is a combination of activation between memory management and tape operation programs.

3.09 The programs are read into and then executed in the paging buffer. This buffer is a location in fixed store (paragraph 3.05) whose size is dictated by application usage requirements. Paging saves a significant amount of storage. The functional details pertaining to memory management are in Section 254-340-064.

4. DEVICE MANAGEMENT

4.01 The basic functions of device management are:

- Keeping track of the status (busy, idle, out-of-service) of each device
- Determining which requesting process or task may use an I/O device, when, and for how long
- Assigning an I/O device and the associated control units and channel
- Releasing the I/O device and making it available to another requesting process/task.

These functions encompass both channel management and device management.

A. Channel Management

4.02 The EOS maintains a data base which provides initialization information for all peripheral channels, the direct memory access (DMA), and all assigned I/O devices. The data base contains device identification and characteristics. This includes assigned parallel channels and associated subparallel channels, interrupt levels, and number of duplex bus selectors on a parallel channel.

4.03 The EOS is responsible for controlling the DMA access during an initialization sequence. Whenever the processor goes through an initialization or switches the off-line and on-line 3A CC units, EOS sets the DMA control bits so that the off-line processor ***cannot*** access the on-line processor main memory. The purpose is to isolate any erroneous data and preserve the integrity of the backup memory contents.

B. Device Management

4.04 The EOS currently has the capability to manage the following I/O devices:

(a) Cartridge Tape Unit

(b) TELETYPE®

(c) Programmable Magnetic Tape System (PROMATS)

(d) RSI-232 Remote Interface Unit

(e) Direct Memory Access

(f) System Status Panel

(g) DATASPEED® 40.

4.05 Each I/O device supported by EOS has an entry in a "device equipment table" which is specified in the operating system table and is maintained as part of the system configuration. This consists of the device name, location, control, channel identification, and interface data:

(a) Type of channel required

(b) Device address

(c) Device type

(d) Device software name

(e) Device TTY message name

(f) Device I/O functions, ie, read only, write only, or read and write

(g) Assigned interrupt level.

4.06 The mechanism of channel and device management is via EOS system tasks. These tasks are created as required by service calls from application tasks or other EOS tasks when channel or I/O activities are required. Service calls are initiated by EOS system macros from the application or EOS tasks.

C. Terminal Administration

Note: Applicable to generic issues prior to Issue G2A.

4.07 The terminal administrator task provides the interface between a process or task that wants to send or receive a TTY message and teletypewriter controllers (TTYCs). The program provides for eight software defined channels. Each channel is assigned a specific activity. For example, channel 0 is defined as the maintenance channel and will receive all maintenance messages.

4.08 The terminal administrator (TERMAD) is an EOS task which processes binary coded client output messges into the code which the TTY utilizes, directs it to the proper channel, and issues the write command when the TTYC is available. The TERMAD also receives input messages and passes them to client programs. Details of device management and terminal administration are in Sections 254-340-054 and 254-340-062.

D. EOS Commands

4.09 The EOS provides several direct TTY commands to facilitate its use. Processes implemented by EOS commands are low priority and execute in the background without interfering with real-time task execution.

4.10 Examples of some of the EOS commands currently being supported are:

- Set or read the system clock
- Set or change a process or task state
- Load from the PROMATS tape
- Enter the interactive debugging facility.

E. Data Administration

Note: The following is applicable to Issue G2A and subsequent issues of the generic.

4.11 The EOS data administration programs provide a centralized package for controlling and distributing I/O messages between devices and EOS or application programs. The programs pass messages to clients and receives messages from clients in an encoded format that relieves clients of the tasks of manipulating ASCII strings. All formatting is performed for the client by the central routines, thus ensuring consistency and proper system states for message processing.

4.12 Data administration performs basic I/O functions such as backup device selection, duplication of data on more than one device, and determining which device to use in a centralized administration package. This permits the acceptance of data from a client and proper redirection to the appropriate device(s). The package also reads data from a device and directs it to the appropriate device. All reads and writes are via the EOS file system. See Section 254-340-040 for details.

5. INFORMATION MANAGEMENT

A. File System

5.01 The EOS provides information management in the form of a file system with cataloging facilities and management of the cartridge data base.

5.02 The file system provides the common user interface with information residing on any of the supported peripheral devices. The application program defines the file to be accessed in the form of a User File Block. It then requests operations on the file through macro calls. It is the job of the file system to locate files and translate user requests into calls to the appropriate device handler, etc, to execute these requests.

5.03 The primary information data base for EOS is contained on the data cartridge. This

includes the generic, nonresident programs, office data base, patch files, etc. Much of this information must be positioned and formatted on the cartridge is very precise and well-defined positions due to bootstrapping considerations. The EOS provides mechanisms for building this cartridge, accessing data on the cartridge, and maintaining the information. Modifying information on the cartridge is supported by EOS facilities such as patching and by file system general access to the cartridge.

B. Tape Cartridge Management

5.04 The tape cartridge supported by EOS is employed as backup for the primary system memory and data base. The EOS allows the application to specify one of three EOS supported access methods (sequential, basic, or random) to control how data is physically stored on the cartridge.

- ***Sequential Access*** treats the entire cartridge as one continuous medium and any read or write operation addresses the tape record following the one previously read or written.

- ***Basic Access*** treats the entire cartridge as one continuous medium and allows records to be accessed in a random manner based on location information contained in the record.

- ***Random Access*** treats the cartridge as a collection of files accessed independently and allows records to be randomly accessed within a file based upon location information contained in a precreated cartridge resident directory.

C. Patching and Overwrite

5.05 The EOS provides capability to patch and overwrite programs. Overwriting is an operation which allows replacing existing data or instruction code in a particular memory location with new data or instruction code. Patching replaces a program instruction with a branch instruction to a modified or new section of code in a special patch area with a return to the original code sequence.

5.06 A method is also provided to change the data and programs that reside only on the tape cartridge. This method employs a dedicated tape file (patch file) which identifies the locations which have been patched plus their new contents. The EOS arranges that all changes in the patch file are applied to backup and nonresident programs and data when they are loaded into the processor main store.

6. MAINTENANCE

A. General

6.01 The EOS maintenance services are available to the user via a straightforward interface based on system maintenance control macros. The EOS maintenance services encompass three basic functions:

(a) Initialization and recovery

(b) Duplex processor operation and resident maintenance

(c) Common diagnostics.

These functions contain the essential maintenance features provided by EOS which are used by an application to meet its system requirements. A general description of these basic functions is in Section 254-340-080.

6.02 The EOS furnishes additional maintenance support to an application in the form of audits and system utilities.

B. EOS Audits

6.03 Audit programs verify and maintain the integrity of the system and provide for an orderly recovery from a system initialization. Audit programs detect both when the system is malfunctioning and ensure that the data base is maintained during and after initialization. Audit programs perform the following functions:

- Verify the integrity of the data base

- Repair erroneous data, if possible

- Collect resources that are tied up in error

- Notify maintenance programs when system must be reinitialized.

6.04 The two basic types of EOS audits are background and initialization audits. Audits

that run in the background are run at low priority to monitor the system and check data which is not dynamically changing. Background audits cause system initializations when uncorrectable inconsistencies are discovered. Since background audits will be interrupted frequently because of low priority, most dynamic data cannot be audited without blocking interrupts during a particular test. Thus, audits of the EOS kernel and file system tables are only run during system initialization.

6.05 The EOS main store audit programs complement correct main store locations in both memories so that if the processor cannot be operated in the duplex mode, the functioning 3A CC has an improved chance of maintaining system integrity. The main store audit performs the following functions.

(1) Can be invoked manually via TTY or automatically.

(2) Runs entirely at the lowest priority level (0) and uses up real-time left over from application functions.

(3) Is invoked whenever a successful update of the off-line 3A CC has occurred. It is normally stopped whenever the off-line 3A CC is removed from service.

(4) Scans both main stores for words having single parity errors.

(5) Double parity errors will cause the 3A CC in which the error occurs to be taken off-line; the erroneous word is then complement corrected.

C. Utilities

6.06 The utilities programs are called into service via the maintenence TTY with message formats as specified in the Input/Output Message Manual. Both resident and nonresident general purpose utility functions are implemented for the EOS.

6.07 Examples of the resident utility functions are as follows:

(a) Load Store—Used to modify the contents of a temporary store word.

(b) Monitor Store—Used to monitor four consecutive 16-bit store words for changes.

(c) Load Register—Used to modify the contents of one of the general purpose registers (R0 through R15).

(d) Monitor Register—Used to monitor the contents of four consecutive general purpose registers for changes. The output message from the monitor utilities may be directed to either the office maintenance TTY or the display buffer. The monitoring occurs once per base loop.

(e) Load Indirect (Store).

(f) Monitor Indirect (Register)—The monitor and load indirect TTY message calling the utilities will contain the register numbers of a pair of registers which contain the 20-bit main memory address of the word to be loaded or monitored. Otherwise, the action is the same as load store and monitor register.

(g) Dump Store

(1) The contents of a specified block of on-line store locations are printed on the maintenance TTY.

(2) The contents of a specified block of off-line store locations are printed on the maintenance TTY.

(3) Off-line store locations are printed when a system initialization occurs.

6.08 Examples of the nonresident utilities are as follows:

(a) Off-line Register Dump—Prints the contents of specified general purpose registers in the off-line control unit on the maintenance TTY.

(b) Overwrite Utility—This utility provides the means of making authorized changes to the system software.

A description of the utility functions is in Section 254-340-082.

7. EOS FUNCTIONAL INTERFACE

7.01 The EOS system calls provide the interface between the application programs and the operating system and are designed for a wide variety of uses including:

- Timer control
- Event control
- Current process control
- External process control
- Interprocess communication
- Storage control
- Input/output control
- Maintenance control
- General purpose
- File system.

7.02 Each system call can be classified as either an active or declarative macro. Generally, active macros will expand into executable code, while declarative EOS macros generate data structures. See Section 254-340-106 for the description of EOS system macros.

8. GLOSSARY

8.01 The following terms and definitions are used in this document.

Application (Programs)—Programs that run under control of EOS and are designed to support a specific system application on the 3A Processor.

ASCII—American Standard Code for Information Interchange.

Bit—An abbreviation of the term "binary digit", a bit constitutes a single position in a binary number and has value 0 or 1.

Buffer—A temporary storage area, generally a memory word(s) used to temporarily hold data and isolate circuits or program segments.

Control Unit (CU)—Consists of a 3A CC, main memory, cartridge tape device, and supporting equipments.

EOS—Extended Operating System.

Input/Output (I/O)—A peripheral data transfer device such as a TTY or tape drive, or a data read or write which causes some data media to change state.

Kernel—The basic software functions of EOS as compared to those that operate as processes.

Macro—A frequently used set of instructions identified by a name and parameter values. Used to simplify programming and to generate more uniform code.

MAS—Main store.

Memory—Synonymous with storage—consists of devices arranged so that large units of information may be held (written) and, at a later time, retrieved (read).

Operating System—A collection of software which provides management of system resources.

Paging—An organized method of moving large quantities of program code or data from one storage medium to another under program control.

Parity—A data validation check which consists of comparing the odd or even status of bits of a data word against a bit indicating that status.

Priority—The assigned level of importance by which various entities will be processed, if more than one requests service at the same time.

Process (EOS)—The execution of a series of programs whose sequence of execution is specified by a file of commands.

PROMATS—An acronym for "Programmable Magnetic Tape System", a magnetic tape recorder.

Store—Same as memory.

Task (EOS)—An execution module which contains all of the necessary information to allow asynchronous execution.

TTYC—Teletypewriter controller.

3A CC—3A Central Control.

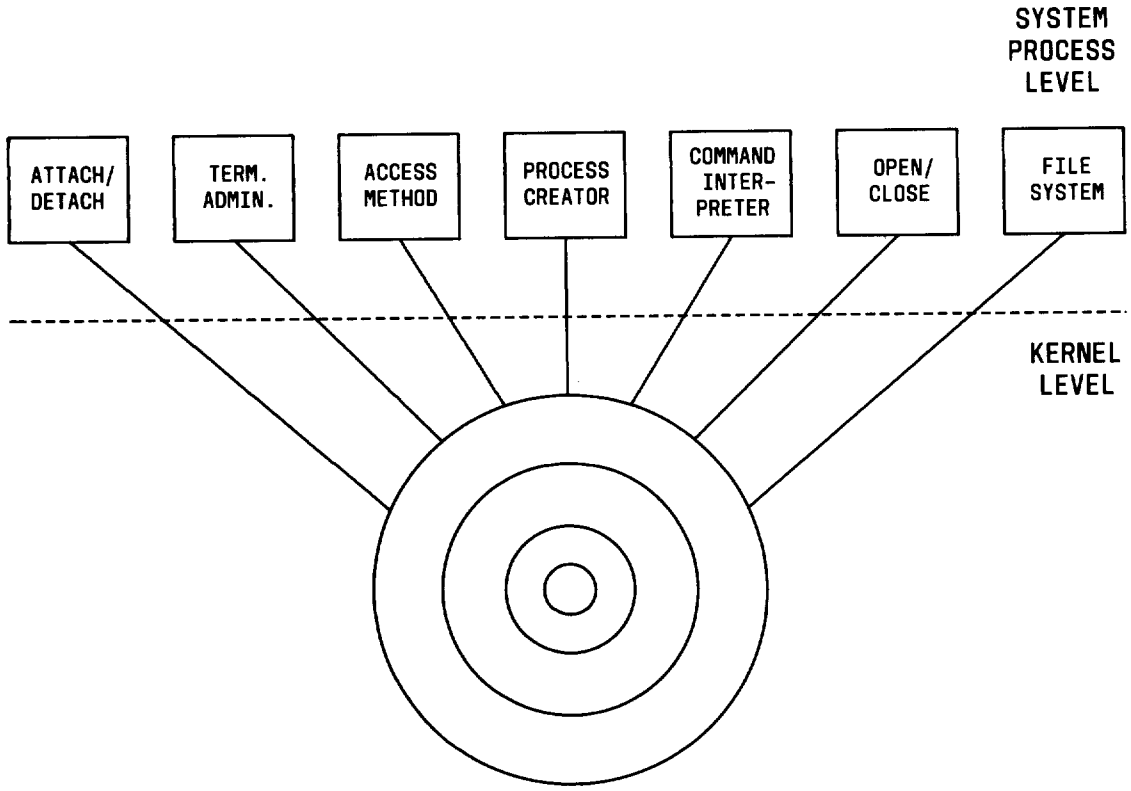

Fig. 1—EOS Functional Structure

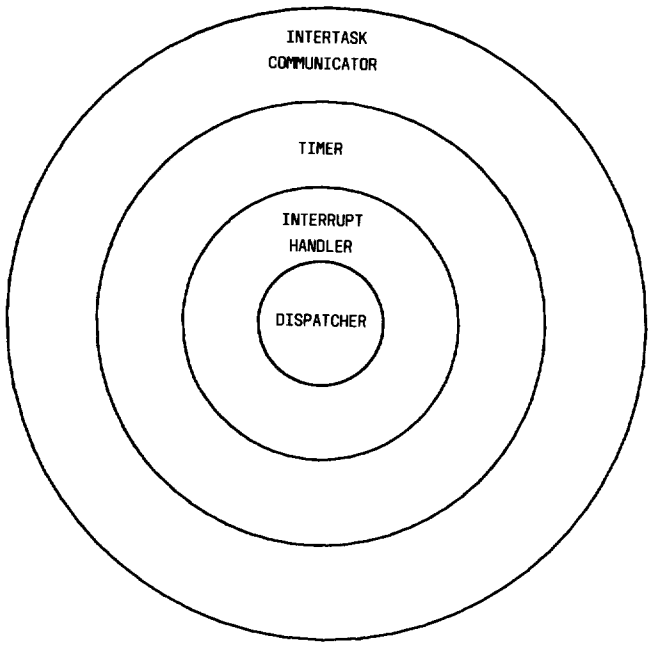

Fig. 2—EOS Kernel Hierarchy

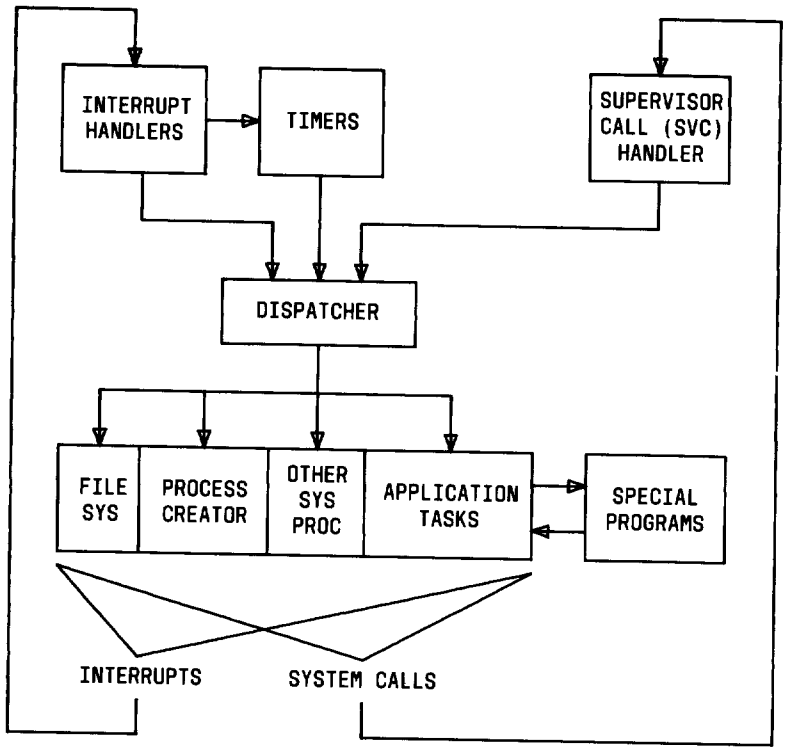

Fig. 3—EOS Call Structure

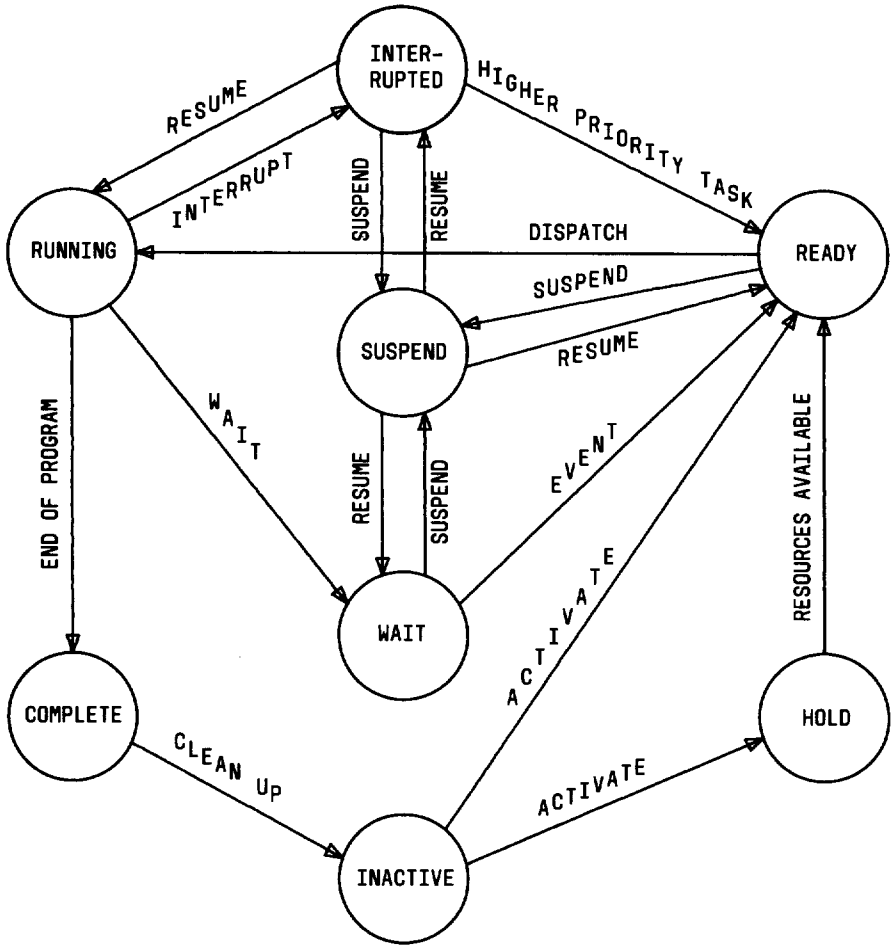

Fig. 4—EOS Process States