The functions of disk and tape operating systems for SYSTEM/360 *configurations with as little as sixteen thousand bytes of main storage are discussed. The two related systems are designed to provide a range of services that include input/output control, stacked-job control, symbolic device assignments, and library maintenance. A set of language translators, a set of sort/merge programs, and various other programs go far toward minimizing the effort required of program preparation.*

Design objectives, system definitions, and functional capabilities are stressed. The design of the control program is discussed in some detail.

Function and design of DOS/360 and TOS/360

by G. Bender, D. N. Freeman, and J. D. Smith

Prior to third-generation computers, few systems programmers had tried to implement full-function operating systems for small-storage computers. True, most installations had many of the functional requirements that justified operating systems, but storage limitations made it impractical to meet these needs on an integrated basis. At present, however, two forces are clearly at work to change the trend. First, the new computers bring with them a degree of device modularity and flexibility that intensifies the need for integrated operating systems, and second, these same computers possess a degree of efficiency that makes operating systems practical for smaller main-storage areas.

The purpose of this paper is to discuss the Disk Operating System (DOS/360) and the Tape Operating System (TOS/360) for IBM's SYSTEM/360. Despite their frugal use of main storage, these systems offer functional services comparable to the IBM 1410 Operating System or IBM 7090 IBSYS.[1] The basic objectives of the systems are to systematize stacked-job processing, support a wide range of input/output devices, provide the full advantages of high-level languages, and ease upward extensibility in languages and operating environments.

Each of the systems is designed for the standard SYSTEM/360 instruction set[2] and requires approximately six thousand (6K) bytes of main storage for system residence. A minimum configuration consists of a 16K MODEL 30, an IBM 1052 printer keyboard, a card

reader (or equivalent magnetic tape), a printer (or tape), a punch (or tape), and either four IBM 2400-series magnetic tape units for TOS/360 or one IBM 2311 disk drive for DOS/360.

For the configuration with more main storage but a fairly straightforward set of operating needs, the systems are possible alternatives to the more modular and versatile OS/360 operating system.[3] For configurations with 32K or more bytes of main storage, the systems offer a form of fixed-priority multiprogramming in which one or two "foreground" programs can be executed concurrently with a low-priority background program. In DOS/360, a telecommunication application can also be executed as either a foreground or a background program.

Each system, intended to assist the user in program preparation, data management, and device control, consists of a control program and a set of supporting programs. The latter include translators for five programming languages; general-purpose programs for sorting and merging; a number of utility programs; a generalized program that assists in program debugging and correction; and service programs for creating and maintaining the system libraries.

The control program consists primarily of an Initial Program-Load (IPL) routine, a Supervisor, and a Job Control routine. IPL is called and used whenever system operation is initiated; among its functions is the loading of the Supervisor into main storage and the branching to the Supervisor. The Job Control routine, which is executed between jobs, is loaded by the Supervisor as needed. The Supervisor launches input/output operations, services interruptions, and performs various other functions.

Although part of the Supervisor remains in main storage throughout system operation, many segments of the Supervisor, called *transient routines,* are executed in a main storage "transient area" that is allocated to the Supervisor. When a routine is called, it simply overlays the existing contents of the transient area.

System elements

Translators are provided for the Assembler Language, COBOL, FORTRAN, PL/I, and RPG (Report Program Generator). In general, the languages follow the syntactical rules of the corresponding languages in OS/360. The PL/I translator is designed for 32K systems. The 16K Assembler Language differs from the full SYSTEM/360 Assembler Language in minor ways; e.g., bit-length specifications in DC and DS statements are not allowed. The system macroinstructions used by the control program are available in the Assembler Language.[4] The translators use either 2400-series tapes (DOS/360 and TOS/360) or 2311 disk storage (DOS/360 only) as intermediate work files.

program structure

Although executable programs in the system are essentially images of main storage, they may be either *absolute* or *self-relocating*. Programs are called self-relocating if they initialize (set up)

their own address constants so that they can be fetched or loaded into any main-storage area and executed there. Most problem programs, as well as the non-transient part of the Supervisor, are stored in absolute form. Transient routines are self-relocating.

A set of statements written as input to a language translator is referred to as a *source module* and the translator output as an *object module*. The object module consists of one or more control sections and a control dictionary. A control section is a segment of code intended for one contiguous area of main storage; a control dictionary contains information needed when the object modules are link-edited to produce *phases*. An executable program consists of one or more phases.

Programs can be written as source modules in any of the five programming languages. Object modules from different translations of different languages can be link-edited to produce one phase. The necessary linkage conventions, based on the traditional CALL-SAVE-RETURN concept, are essentially a subset of the OS/360 set. Direct linkage is available with Assembler Language, COBOL, FORTRAN, and PL/I for routines that always appear in main storage together as a single phase.

For circumstances in which some phases of a program are to be stored externally and then called during program execution to overlay other parts of the program in main storage, the user can define an overlay structure for the linkage editor by means of control statements. COBOL or FORTRAN overlay structures can be handled by a short Assembler Language program that invokes the Supervisor through FETCH or LOAD macroinstructions; the Supervisor reads a phase into main storage and optionally branches directly to it. PL/I contains facilities of its own for loading multiple phases of a single program.

An independent program is normally executed as a job. It is superfluous to connect two programs within an overlay structure and execute them consecutively as one job if the programs are related only by results recorded on external storage media. For such cases, *job steps* are defined. For instance, the only relation between a compilation and a subsequent link-edit is the object module resulting from the compilation, which the system always stores on an external device. Therefore, a compilation followed by a link-edit is typically viewed as two steps of one job. The system permits the remnant steps of a job to be ignored if one step fails.

the librarian

The routines comprising the Librarian facilitate insertion, deletion, and replacement of elements of any of the three libraries: core-image, relocatable, and source-statement libraries. Furthermore, tables of contents can be printed, modules from the relocatable library or the source-statement library can be punched, and listings of modules can be obtained. The librarian stores relocatable and source-statement modules in a compressed format.

It is the primary function of the Linkage Editor to combine object modules supplied in the job input stream, and object

modules from the relocatable library into executable programs. One phase can contain one or several object modules, and one object module can supply control sections to several phases. It is of interest that the Linkage Editor allows the same control section to occur in different phases of the same overlay structure. Thus, a program may use the same input/output routines in its first and last phases without sacrificing main storage for these routines during intermediate program phases.

linkage editor

Inasmuch as four higher-level programming languages are offered, many small-system users will be encouraged to code fewer applications in the Assembler Language. When using COBOL, the user can request test information in the higher-level language. When using FORTRAN or PL/I, the user may insert temporary READ or WRITE statements to monitor test data.

debugging aids

A program-testing aid called "Autotest" is provided. Autotest is particularly suited to Assembler output, although it can also help in the debugging of object modules from other translators. Since most programs are link-edited and loaded to the same absolute addresses, testing tools need not be fully symbolic. Print requests can be inserted in a program when it is link-edited under Autotest control. As each such request is honored, its storage-area limits can be displayed in symbolic form if assembly symbol tables have been furnished to the Linkage Editor. Other functions of Autotest include *patching* that reduces the number of reassemblies required during testing, *dumping* at the termination of a job step, *recording* of phase calls, and *linking* to utility programs before or after test execution.

Programs are provided for sorting and merging either tape or disk files. The tape sort/merge programs take advantage of the read-while-write feature and the tape-switching device. The sorting algorithm, a generalized polyphase merge for three to six tapes, uses the read-backward feature and allows the user to specify the final output drive without requiring a copy-only pass.

sort/merge

The utility programs can be used to move data between card readers, card punches, printers, tapes, disks, and data cells. Fields can be rearranged and records can be reblocked during the data transfers.

utilities

The system allows sequential, indexed sequential, and direct file organization. Card, printer, and tape files are always sequential; DASD (Direct-Access Storage Device) files may be sequential. To write or read a record in a direct-file organization, the DASD location of the record must be given; sequential reading can yield a completely arbitrary sequence of records. The indexed sequential file organization is used with direct-access storage devices to retain the advantages of a file that is ordered on data keys while gaining some of the advantages of random access. Direct access is accomplished with the aid of index tables, which contain the addresses of selected keys in the ordered file.

file organization

Macroinstructions designed for use with the control program are called system macroinstructions. Many of these provide for

macro-instructions

input/output control, supervisor communication, and direct linkage. Depending upon the indicated file organization, input/output control macroinstructions may take different parameters and create different coding. Macroinstructions intended for use in system generation, multiprogramming, and teleprocessing applications are also provided.

Control-program structure

supervisor

The DOS/TOS/360 Supervisor is the key component of the control program. Its functions include:

- Invoking necessary setup functions for each job
- Furnishing a description of the machine configuration for the machine operator and the problem programs
- Performing I/O services at the most elementary level
- Issuing messages to the operator whenever his intervention is required
- Responding to interruptions, taking the system default action whenever a job is unconcerned with the interruption
- Fetching phases of an overlay program
- Taking user-directed checkpoints, prints, and post-mortem dumps
- Task switching and task selection for multiprogramming (scheduling of the highest priority partition that is not in a waiting condition)

The Supervisor resides in the first 6K bytes of main storage. Slightly larger or smaller variants of the Supervisor can be generated, depending upon optional machine features and certain processing options. Rarely will a basic Supervisor either exceed 6500 bytes or be less than 5900 bytes. Supervisors furnishing multiprogramming and/or telecommunication support range up to 10K bytes.

Each installation generates one or more Supervisors tailored to its needs. Specifically, each Supervisor is defined by a set of macroinstructions whose parameters describe the configuration, options, and special supervisory services of the installation. The Supervisor is the only major component of DOS/TOS/360 requiring on-site assembly; all other processors adapt to the installation by interrogating parameters assembled into Supervisor tables. This procedure minimizes the effort expended by each installation for generating and maintaining its system.

Above the Supervisor is the problem-program area, divided into the background partition (or batch-processing partition) and, optionally, one or two foreground partitions. If neither of the latter is needed, the entire problem-program area is available to the user for batch-job processing. The Supervisor never "borrows" from it. (The Autotest monitor does borrow from upper main storage for the testing of programs.)

During the execution of each program, one or more transient supervisory functions may be requested, e.g., routines to print

main storage, open data files, etc. Other transient functions are called by the Supervisor when exceptional I/O conditions occur (such as a stacker-empty condition for a card punch). These transient routines are retrieved from systems residence into the Supervisor, where 1700 bytes are reserved for this purpose in DOS/360, and 1500 bytes in TOS/360. Certain complex routines, such as direct-access storage device (DASD) file OPEN, require several segments which overlay each other in this transient area. Such multisegment operations normally require from 0.5 to 5.0 seconds.

The nucleus of the basic Supervisor (i.e., without multiprogramming/telecommunication capability) is approximately 4500 bytes, whereas the transient functions aggregate 30,000 to 40,000 bytes. The Supervisor is a schematic "iceberg," whose "visible" main storage comprises only functions which either are used continually during problem-program execution or are necessary to access the "invisible" transients.

interruptions

The Supervisor responds to all five classes of machine interruptions on SYSTEM/360:

Machine check (hardware failure). The standard system diagnostic procedure is followed and the machine operator is notified; operation cannot continue.

Program check (problem-program failure). The current job is cancelled unless the user has requested that control be directed to his private, asynchronous routine.

Supervisor call. DOS/TOS/360 defines a fixed set of supervisory services, corresponding to approximately thirty different Supervisor-call instructions. Users can augment these services only by source-level modification of the IBM-supplied Supervisor.

External. The Supervisor optionally supports the interval timer in two distinct ways; as a real-time clock for time-stamping printed output, and for interrupting a user program when a pre-set interval elapses. In the latter event, the Supervisor posts a user-specified event block, yields control to a user-specified routine, or ignores the interruption if no linkage exists. The interrupt key is supported in a similar fashion, i.e., either by ignoring the interruption or by linking to a user-specified routine for this asynchronous interruption.

Input/Output. SYSTEM/360 offers a wide variety of different I/O devices which collectively return a multitude of status indications to the CPU whenever interruptions occur. This machine design facilitates thorough checking for exceptional conditions, efficient parallel processing, and great flexibility in configurating various systems. Nevertheless, programming the equipment directly to utilize all the features and respond to all interruptions is a very complicated and specialized task; all necessary functions are

Table 1 IOCS units

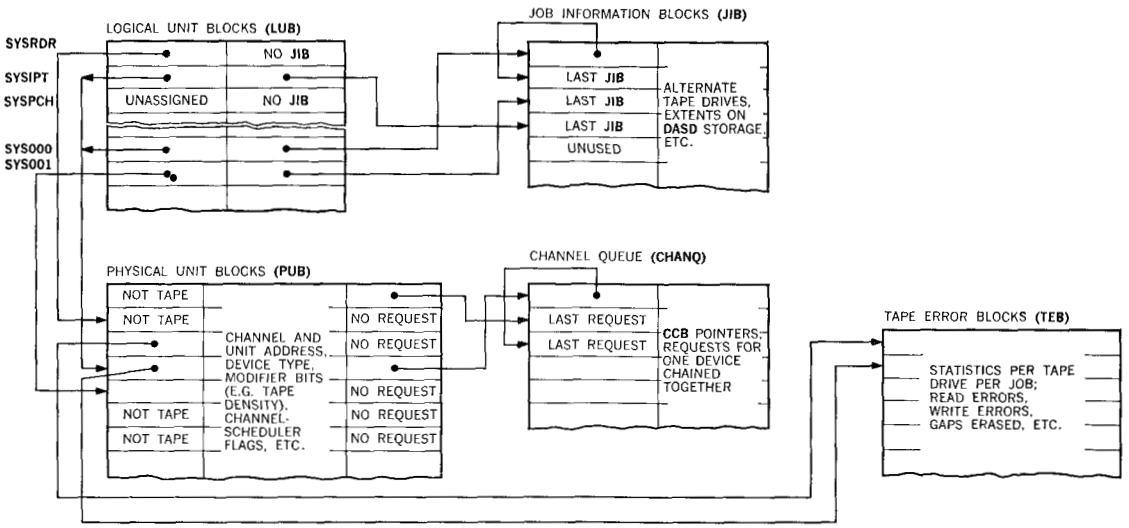
	Logical unit	*Function*	*Device type*	*Remarks*
I	SYSRES	On-line residence	Disk or mag tape**	Accessible by all programs: Supervisor, batched jobs, and foreground jobs
	SYSLOG	Two-way operator communication	Printer keyboard (or line printer for emergency output only)	Operator uses SYSLOG to initiate foreground job
	SYSRDR	Control-card reader	Card-reader, disk,* or mag tape**	Used primarily by Job Control; not available to foreground jobs****
	SYSIPT	Input stream	Card-reader, disk,* or mag tape**	Standard format: 80-byte records; monitored by Supervisor for special job-step delimiters****
	SYSPCH	Punched-output stream	Card-punch, disk,* or mag tape**	Standard format: 81-byte records (ASA first-character control for stacker selection)****
	SYSLST	Printed-output stream	Printer, disk,* or mag tape***	Standard format: 121-byte records (ASA carriage control)****
	SYSLNK	Linkage-editor input	Disk or mag tape**	Also contains tape linkage editor output****
	SYSRLB	Relocatable-module library	Mag tape	Distinct from SYSRES; must be a private tape reel (TOS/360)****
	SYSSLB	Source-statement library	Mag tape	Distinct from SYSRES; must be a private tape reel (TOS/360)****
II	SYS000	Utility file	Any supported device	Arbitrary usage by background programs
	SYS001 SYS002 SYS003	Utility file	Any supported device	Normally assigned to scratch storage on tape or disk; used by compilers, librarian, etc.
	SYS004 SYS005 .	Utility file	Any supported device	Arbitrary usage by background programs
	SYS000 SYS001 .	Utility file	Any supported device	Arbitrary usage by foreground-two programs*****
	SYS000 SYS001 .	Utility file	Any supported device	Arbitrary usage by foreground-one programs*****

* DOS/360 systems w/minimum 32K main storage
** 9-track or 7-track w/data-convert feature
*** 9-track or 7-track; data-convert feature is unnecessary
**** Accessed only by background partition
***** SYS000 is used for foreground dumps and job termination messages if assigned to a printer or tape

included in the Supervisor. Those functions which schedule overlapped I/O operations, accept I/O interruptions, and service exceptional conditions of I/O devices are collectively called *physical* IOCS.

All requests for I/O service, whether from the Supervisor, batch jobs, foreground, or telecommunication jobs, are directed to the

Figure 1 Sample network of IOCS tables



Channel Scheduler, the principal component of physical IOCS (which in turn represents over sixty percent of the basic Supervisor nucleus). DOS/TOS/360 responds to any interruption from any device, even if totally unknown to the Supervisor. For each "anticipated" interruption, i.e., resulting from a previous I/O request, DOS/TOS/360 posts the appropriate status information to the requestor's Command Control Block (CCB). Identical CCB's and I/O requests are issued by foreground and background programs; the Supervisor distinguishes these requests only by the protection keys of the requestors. Specifically, the Supervisor runs under protection key 0, background jobs under key 1, and foreground jobs under keys 2 and 3.

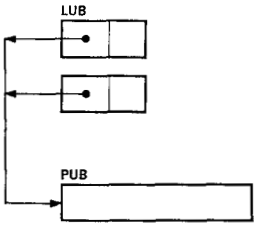
channel scheduler

Since the assignment of unit numbers to SYSTEM/360 devices may vary between and within installations, it is tactically undesirable to use such physical addresses in programming; otherwise, each application can run only on a single configuration and cannot operate if any single I/O device in that configuration becomes unavailable.

symbolic units

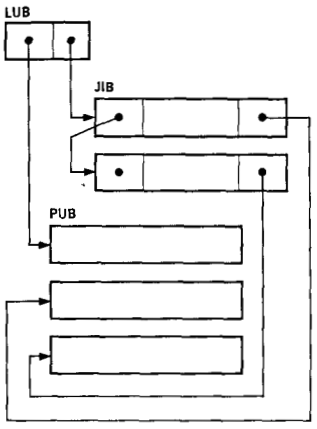
Thus, two classes of symbolic units in the form SYSxxx are defined as shown in Table 1. Each CCB references precisely one symbolic unit. Any device can be assigned to any symbolic unit except as noted in Table 1. Several symbolic units can be assigned to a single I/O device, although most programs do not ordinarily use several "aliases" simultaneously. For example, consecutive cards are usually read using a single symbolic unit. Figure 1 shows a sample set of IOCS tables in the Supervisor nucleus, and Figure 2 indicates in detail how two symbolic units (i.e., Logical Unit Blocks, or LUB's) point to one Physical Unit Block (PUB).

Figure 2 Physical/logical unit correspondence: 2 LUB's to 1 PUB



Symbolic names are primarily for the convenience of the machine operator. Knowing the symbolic names required by a

Figure 3 Physical/logical unit correspondence: 1 LUB to 3 PUB's



stream of programs, he makes assignments from a pool of I/O devices. With appropriate assignments, he can substantially reduce his effort to mount and dismount tape reels, disk packs, etc. Also, he can greatly improve system throughput by assigning faster tapes to certain functions for each program. For example, the TOS/360 systems-residence tape should be one of the faster tapes, whereas program listings should ordinarily be diverted to a slower tape (*en route* to a printer).

Another type of physical-symbolic correspondence is possible. Two or more physical devices, ordinarily magnetic tape drives, can be assigned to a single symbolic name, so that user programs need not handle end-of-volume conditions. Figure 3 gives an example of this: one LUB attached to three PUB's. The Supervisor automatically closes and rewinds each reel of a multi-reel file, then opens the reel on the next alternate drive, together with all requested label checking and label creation.

I/O queuing

The Channel Scheduler queues the requests for each I/O device and serves them in First-In First-Out (FIFO) order. Furthermore, the Channel Scheduler services each selector channel in FIFO order except when an impending request is to a busy device. In such cases, the Channel Scheduler dequeues out of order until the busy device becomes free. For example, a rewind-and-read sequence of commands to a tape drive can be performed without interlocking I/O operations on other tapes.

The Channel Scheduler fully utilizes the byte-multiplexing capabilities of low-speed devices (such as card readers and punches, printers, and terminals) when attached to the multiplexor channel by restarting these devices as soon as each becomes free (as long as there are pending requests). If burst-mode devices are attached to the multiplexor channel, only non-overrunable devices are multiplexed. For example, if magnetic tapes are attached to the multiplexor channel, the IBM 2540 card reader and punch can be byte-multiplexed but the IBM 1442 card reader and punch is started only when no tapes are transmitting data.

Often an I/O request cannot be started because the channel leading to a device is busy, even if the device itself is idle. If two channels lead to the same device, the Channel Scheduler automatically attempts to start the idle device using the second channel if the first channel is busy. This technique fully utilizes the switching capabilities of the IBM 2404, 2804, and 2816 tape control units.

file protection

Since different programs, operating either concurrently or successively, can validly use the same direct-access volume, a method is required to prevent one program from overwriting information belonging to another program. DOS/360 offers the following optional facility. Every channel program for a direct-access device must start with a SEEK Bin Cylinder Head (Long Seek) command addressing a cylinder previously assigned to the program issuing the I/O request. If this Seek command should address an unauthorized cylinder, the program is automatically

cancelled. Likewise, any Long Seek command embedded in a channel program results in job cancellation. Cylinders are assigned, i.e., "authorized," by the DASD open-file routine after verifying user-supplied control statements against the label and Volume Table of Contents (VTOC) on the volume itself.

Before a valid Long Seek command is issued, it is moved into the Supervisor. The Channel Scheduler starts the complete channel program, after setting a file mask to restrict further movement of the access mechanism.

Other important protective features are built in. For instance, records starting with /* or /& in the system input stream (SYSRDR and SYSIPT) are always considered delimiters and cannot be used as data. Thus, one job cannot erroneously overread SYSRDR/SYSIPT and destroy the succeeding job. This protection is performed by the Channel Scheduler.

levels of IOCS

Most application programs are written in higher-level languages or in the Assembler Language, using pre-defined data files. In either case, available IOCS logic is linked into application programs. The user need not concern himself with machine-language I/O requests, CCB's, and channel programs, which are discussed later. However, certain specialized I/O devices (or infrequently-used functions on standard devices) are not supported by the higher-level IOCS. Users write simple macroinstructions in their Assembler Language programs to execute channel programs (EXCP), await their completion (WAIT), and test various status bits to validate each I/O operation. Figure 4 displays the three levels of IOCS offered in DOS/360 and TOS/360: the lowest one is at the physical IOCS level (error retry, etc.), the intermediate one consists of the lowest level plus error processing and automatic generation of channel programs, and the highest one consists of a combination of the intermediate level, buffer management services, and deblocking services.

operator-to-system commands

Using the 1052 printer keyboard, the operator can, at any time, initiate a foreground program in an idle partition and cancel any currently executing job or force a pause after its completion. At cancellation, the CPU registers and main storage can be optionally dumped out on SYSLST (SYS000 for foreground programs) to provide diagnostic information for the programmer.

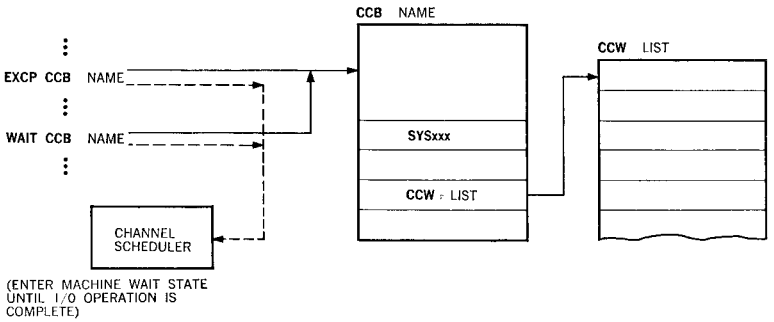
During the execution of assembler-language programs, the programmer can request additional dumps of registers and selected portions of main storage. These PDUMP's (prints of main storage) destroy general registers 0 and 1 but do not otherwise interfere with system status. The Supervisor prints the diagnostic output directly on SYSLST, thus providing a selective on-line trace of program flow.

Between job steps, the operator can use a wider selection of commands:

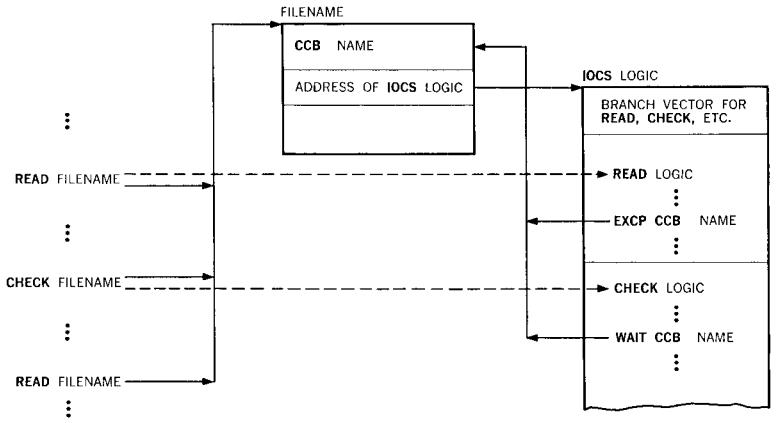
1. Assign new symbolic units temporarily or permanently—for the duration of a single job or for the entire job stream. For

Figure 4 Levels of IOCS

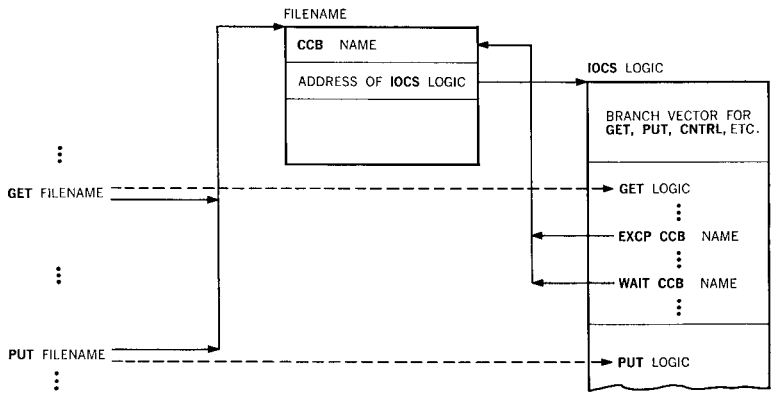
A. LOWEST LEVEL



B. INTERMEDIATE LEVEL



C. HIGHEST LEVEL



example, the operator can assign SYSIPT (a card reader, tape, or disk) for a job stream, whereas each programmer can affect only his own job by any reassignments.

2. Close any tape file manually and manipulate tape drives by simple commands, as for example,

CLOSE	SYS002	Using standard label procedure
MTC	WTM, SYS003,2	Write two tape marks on SYS003
MTC	RUN, SYS003	Then rewind and unload it

3. List I/O assignments selectively. LISTIO SYS prints on the 1052 the following display:

CHAN	UNIT	LOGICAL NAME
0	0C	SYSRDR
0	0D	SYSPCH
0	0E	SYSLST
1	81	SYSRES
.	.	.
.	.	.
.	.	.

Also, the operator can respond to various system queries during the running of each job. For example, the operator can override certain label mismatches for tape reels and disk packs, e.g., by typing *ignore*. If he judges the discrepancy to be serious, he can type *cancel*, thereby terminating the entire job. Likewise, when a print check, punch check, or other I/O error occurs that cannot be corrected by programming, the operator can *ignore*, *cancel*, or normally *retry* the operation, presumably after performing any necessary manual correction procedures. The options available to the operator are dependent upon the type of error.

system loader

The System Loader (also called Program Fetch) is part of the resident Supervisor. During execution of an overlay program, the System Loader searches the core-image library for each phase as requested. If the job is executed in a compile-and-go situation, phases are retrieved from the tape or disk containing the just-linked program. Otherwise, the permanent library is addressed as described in the following paragraphs.

With 9-track tape for TOS/360 residence, the System Loader "remembers" the SYSRES position whenever possible, i.e., if other programs do not reposition SYSRES. Each phase of a user program requests the next phase, using the FETCH macroinstruction (or EOJ, the request for End-of-Job). FETCH interruptions give control to the Supervisor, furnishing the name of the requested phase to the System Loader, which compares this name to the name of the current linear position. On high-compare, the System Loader searches forward in the permanent library; on low-compare, the System Loader searches backward, using the read-backward capability of 9-track tape.

Table 2 Result of a library maintenance run updating phases PAYROLL2 and PAYROLL3 of the PAYROLL program

	Core image directory		*Phase directory (built when requested)*	
	Phase name	*Location*	*Phase name*	*Location*
Before update	.	.		
	.	.		
	.	.	PAYROLL	CHR 1
	PAYROLL	CHR 1	PAYROLL3	CHR 3
	otherphase 1	.	PAYROLL2	CHR 2
	otherphase 2	.		
	.	.		
	.	.		
	.	.		
	PAYROLL3	CHR 3		
	.	.		
	.	.		
	.	.		
	PAYROLL2	CHR 2		
After update	.	.		
	.	.		
	.	.		
	PAYROLL	CHR 1	PAYROLL	CHR 1
	.	.	PAYROLL2	CHR M
	.	.	PAYROLL3	CHR N
	.	.		
	PAYROLL2	CHR M		
	PAYROLL3	CHR N		
	End-of-Library	CHR P		

CHR = Cylinder Head Record

With 7-track tape residence (or for compile-and-go executions), the System Loader cannot read phases backward. Instead it rewinds SYSRES (or SYSLNK) and searches forward for each phase. The same tactic is required when problem programs reposition SYSRES, e.g., rewind it to save inter-job time. For example, when the IBM-supplied Librarian searches for books outside the core-image library (i.e., in the relocatable library or in the source-statement library), this repositioning is detected by the Channel Scheduler which sets a "SYSRES moved" switch for the System Loader. However, rewind-and-search-forward is often faster than read-backward, particularly if the Librarian is searching far down the system tape.

The DOS/360 System Loader consults an abbreviated phase directory for the location of each phase on system residence. This directory is built by Job Control as each program is requested. The phase directory is merely an abstract of the core-image directory, the latter being a one-level catalog for all phases

in the core-image library. The basic cataloging unit for the core-image library is a phase rather than an entire program, thus facilitating program maintenance, reducing the size of program FETCH (in the 6K Supervisor), and achieving superior FETCH speeds. The phase directory facilitates fast phase lookup, since it is one-level, compact, and completely current, as shown in Table 2. The DOS/360 compilers and large application programs would suffer significantly with additional FETCH overhead.

Programs can retrieve instructions and/or data from the core-image library without yielding control, using the LOAD macro-instruction. Such a text can be loaded into any part of the problem-program area, using either a program-specified load address or its linkage-edited address by default. User programs can thus rapidly access tables or subroutines outside of main storage—the core-image library consists of 1700-byte blocks on the disk system and at least 4000-byte blocks on the tape system.

checkpoint

Since DOS/360 and TOS/360 are designed for small-to-intermediate SYSTEM/360 configurations, a checkpoint facility is required for "rollback" of prolonged jobs if such jobs are deliberately or inadvertently interrupted.

Checkpoints can be taken either on disks, private tapes, or—in the case of sort/merge and other programs using all available drives—on output or work tapes. The user generates checkpoint requests into his program, which can be honored either unconditionally or under operator control: the operator sets one of the user-program switch indicators, which can be interrogated by checkpoint requests in the object program.

Programs are restarted by operator action. The desired checkpoint *generation*, i.e., serial number, is keyed in to select precisely the desired degree of "rollback." Tape files are automatically repositioned during program restart. DASD files are self-repositioning.

job control

Job Control is a program alternating with each job step in the background job stream and furnishing a diversity of services to the job stream:

- Setting of switches to control execution-time options
- Assignment of I/O devices
- Partial verification and temporary storage of label information (tape, disk, etc.)

Job Control is the key element for batch processing of arbitrary jobs (in contrast to batch processing of identical jobs, e.g., multiple FORTRAN compilations). In other words, each programmer uses Job Control statements describing his options and data to either his own programs or to standard processor programs. Job Control furnishes special services to certain processor programs:

For the Linkage Editor. Copying object modules from SYSIPT to a special intermediate file (SYSLNK).

For the compilers. Setting switches in the Supervisor nucleus to control object-module output (DECK/NODECK), symbol-table

output (SYM/NOSYM), listings (LIST/NOLIST, LISTX/NOLISTX), admissible character set (48C/60C), compile-and-go facility (LINK/NOLINK).

For the Program FETCH *subroutines.* A description of where to find executable programs: the tape drive (for compile-and-go jobs) or a special directory giving the disk address of every phase in the requested program. A program being defined as any collection of phases whose names have a common four-character prefix, one program can exit to another program merely by issuing a FETCH macroinstruction for its first phase.

I/O assignments

A second major function of Job Control is to assign physical I/O devices to symbolic names. This function, illustrated in Figure 5, is directed by: (1) the programmer who anticipates standard assignments of a single type, e.g., tapes; (2) the machine operator who assigns devices out of pools; and (3) the standard-configuration description (which is generated into the system). The machine operator has ultimate control of assignments. If Job Control detects a discrepancy, it asks him to correct the assignment or cancel the job.

The operator exercises this control only by exception. The standard configuration usually assigns specific devices to all required symbolic names. Of course, this standard configuration is appropriate to a single machine. If the system volume is moved to another machine, the operator must furnish a set of new assignments. These assignments can be entered by control card as well as through the printer keyboard, avoiding prolonged keyboard tedium for the operator whenever the system must be reloaded.

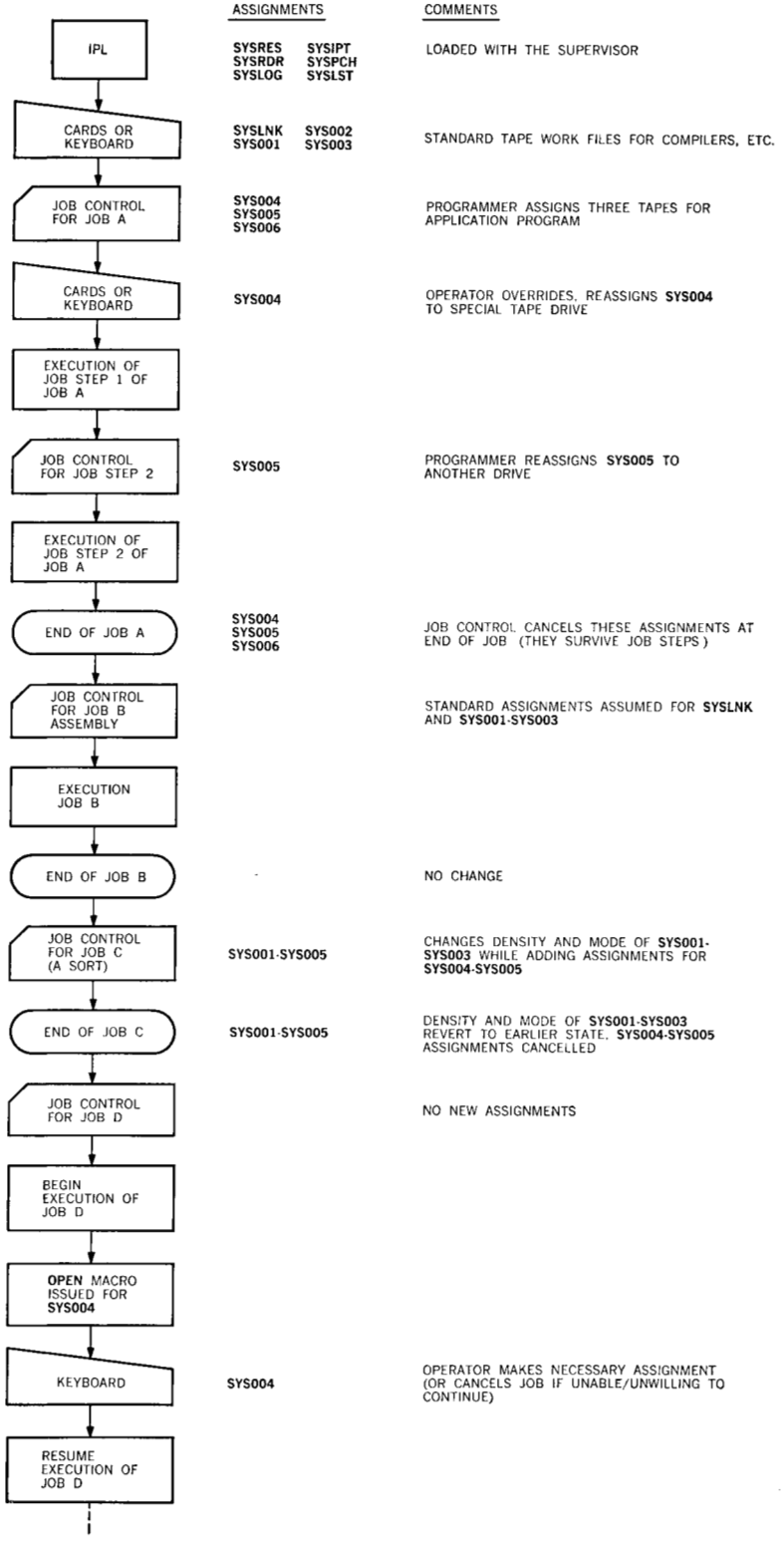
label processing

The third major function of Job Control is to accumulate label information and move it (in compressed format) to a fixed area, thus making it available for OPEN processing. On the tape-resident system, this area is a small block of main storage at the beginning of each program partition (since no external read/write storage is necessarily available, all tapes may be committed to the current application program). On the disk-resident system, the label area is part of the system volume from which label information is read into the transient area of the Supervisor by OPEN.

There are two types of label statements. Volume (VOL) statements establish the correspondence between symbolic units and named files. Tape label (TPLAB) and disk label (DLAB) statements furnish the actual volume and file-label data to be checked against input file labels or written on output files. Disk-label statements are supplemented by extent (XTENT) statements which confirm or assign tracks to each input/output file on direct-access devices.

A distinctive characteristic of disk and tape operating systems is programmer or operator control of label and extent information. (In OS/360, much of the burden is assumed by the control program.) Since each job furnishes this information,[5] the DOS/TOS/360 control program need not retain it internally. On the tape-resident

Figure 5 Sample sequence of I/O assignments



system, the 6K Supervisor is far too small to catalog the tape files of an installation. Many disk installations are unwilling to commit much external storage to catalog their tape and disk files. DOS/TOS/360 serves these requirements by matching the labels on tape reels, disk packs, etc. against control statements, subject to operator-override actions. This places significant responsibility on machine operators in installations dependent upon information security. Such installations will doubtless require sterner DOS/TOS/360 audit trail procedures than would be necessary with OS/360's data management.

Other functions of Job Control include: (1) accumulate input for the Linkage Editor, i.e., copy it from SYSIPT to SYSLNK; (2) manipulate magnetic tapes, e.g., write tape mark(s), rewind unload, space forward files and/or records, etc. (this convenience is also available to the machine operator, who needs it for various emergencies); (3) list I/O assignments to aid the machine operator in making routine assignments out of a pool and resolving assignment discrepancies; (4) initiate each processor, after building its Phase Directory (on the disk-resident system).

Auxiliary functions

system libraries

DOS/360 and TOS/360 offer three on-line libraries, into which elements can be inserted, retrieved by name, or used explicitly/implicitly in application programs.

The core-image library contains executable programs which are blocked into records several thousand bytes long. This format reduces the overhead for program retrieval to a minimum. This is extremely significant for such processor programs as the COBOL and PL/I compilers, which require numerous retrievals of logic per compilation. Thanks to the large blocking factor, most retrievals require no more than five records from disk systems residence.

Furthermore, the core-image library requires relatively little external storage. The tape must be searched, with other processing suspended, whenever any library element is retrieved. With disk residence, pack capacity is the important consideration rather than speed of access. Many users may wish to retain all executable programs on a single pack, allocating remaining cylinders to work files or data storage. The DOS/360 libraries offer a fairly tight utilization of external storage.

The relocatable library contains object modules in a compressed format, i.e., increasing the density of text bytes per block above the 56/80 ratio of ordinary object-module punched cards. The relocatable library furnishes a high-speed source of input to the Linkage Editor, whose AUTOLINK feature automatically retrieves subroutines to resolve external-symbol linkages. Modules can be inserted, retrieved, and replaced by name. They can be explicitly included as well as implicitly included (by way of the AUTOLINK function) into program phases.

The source-statement library contains arbitrary collections of 80-character records (called books), principally for compile-time inclusion in Assembler Language and COBOL programs. System macro definitions are normally retained in this library, e.g., OPEN, GET, DTFxx, FETCH, etc. Many users augment these macro definitions with those standard to their installation. Definitions used infrequently and/or by a single programmer can be furnished at assembly time rather than from the source-statement library.

In addition to Assembler Language and COBOL books, users may retain test data, object decks, elements of other programs, etc. in the source-statement library for security and ease of access. The Assembler Language, COBOL, and Librarian programs retrieve named elements readily. Other uses of the source-statement library require detailed assembler-language programming and knowledge of the library format. The Librarian extracts blanks as it blocks the original 80-character card images into books which are inserted into the library. Blank extraction normally reduces the internal representation of each card to less than 80 characters. When books are retrieved, the original card images are exactly reconstructed. Blocking records reduces the inter-record gaps (conserving external storage) and improves the speed of book retrieval.

compilers

In contrast to previous practice, none of the compilers use the assembler as an intermediate language. DOS/TOS/360 compilers (Macro Assembler, FORTRAN, COBOL, PL/I, and RPG) convert source programs directly to machine language. This method is used to avert machine-language patches by furnishing satisfactory compilation speeds and bypassing an intermediate representation suitable for patching.

Code generated by the DOS/TOS/360 compilers uses a common set of data management (IOCS) subroutines, which are defined in the Macro Assembler language. Thus, improvements in IOCS can be implemented as quickly as the I/O device support for higher languages is available (where this support is appropriate to the language).

multi-programming

For configurations with at least 32K bytes of main storage, a fixed-partition form of multiprogramming permits concurrent execution of up to three programs. Unless a program is self-relocating, each program is assigned to a fixed location when cataloged. The total amount of main storage available to a program may be specified at the time of system generation, or by the operator each time a program is loaded for execution. Storage protection is required.

An application program is classified either as a background or a foreground program. Background programs are initiated by Job Control from the ordinary input stream, whereas each foreground program is individually initiated by the operator via the printer keyboard and special service routines. In the operating sense, background and foreground programs are completely independent of each other.

One background and one or two foreground programs can be executed concurrently. In problem-program requests for the CPU, the foreground-one program has highest priority, the foreground-two program is next, and the background program is last in priority. Control passes to a program when the program with next-higher priority encounters an event that temporarily suspends CPU processing. If all three programs are suspended at the same time, the CPU is placed into the wait state—enabled, of course, for interruptions.

In their conventions, the FORTRAN and PL/I compilers assume that object programs have the properties of background programs. With certain limitations,[6] the object programs produced by the Assembler, COBOL, and RPG compilers may be executed as foreground programs.

In a multiprogramming environment, telecommunication programs are normally run in the foreground-one area because of its highest priority.

tele-communications

The Basic Telecommunication Access Method (BTAM) controls transmission and reception of messages over telecommunication lines in response to READ and WRITE macroinstructions issued in the user's problem program. To accomplish this function, BTAM dynamically generates and executes channel programs and, at the user's option, provides buffer allocation.

The facilities of BTAM are made available through the macrogeneration capabilities of the DOS/360 assembler. From a macroinstruction describing the types of terminals, lines, and other facilities to be used, the user generates his BTAM logic module. It may be assembled with the user program or separately assembled and combined with it at link-edit time. During assembly of a problem program, macroinstructions coded by the user are expanded into: (1) tabular information defining the lines, terminals, and options to be used; and (2) linkage to the BTAM routines. Other information is supplied by the user in the Job Control statements at the time the program is loaded for execution.

When an OPEN macroinstruction is executed in the problem program, telecommunication lines are prepared for data transmission, and initialization for buffer management is performed. When a message is to be received or sent, a READ or WRITE macroinstruction causes a branch to the BTAM READ/WRITE routine. This routine builds a channel program to perform the requested operation and passes the request to the I/O Supervisor, which starts the channel program. Control passes back to the user at this point.

An important distinction between BTAM and other access methods of DOS is in the way in which the channel program is executed. For certain types of terminals and line configurations, the BTAM channel programs may be repeatedly restarted in response to conditions on the line. This allows a single READ to successively poll a number of terminals on a line. A single WRITE can signal a number of terminals to prepare for receiving a message.

CITED REFERENCES AND FOOTNOTES

1. A. S. Noble, Jr., "Design of an integrated programming and operating system, Part I, System considerations and the monitor," *IBM Systems Journal* **2**, 153–161 (June 1963).
2. G. A. Blaauw and F. P. Brooks, Jr., "The structure of SYSTEM/360, Part I, Outline of the logical structure," *IBM Systems Journal* **3**, No. 2, 119–135 (1964).
3. G. H. Mealy, B. I. Witt, and W. A. Clark, "The functional structure of OS/360, *IBM Systems Journal* **5**, No. 1, 2–51 (1966).
4. D. N. Freeman, "Macro language design for SYSTEM/360," *IBM Systems Journal* **5**, No. 2, 62–77 (1966).
5. This is true except for certain procedures available for the disk-resident compilers, etc. A standard label set can survive over jobs if it identifies a "scratch file," i.e., a file containing no information needed for succeeding job steps.
6. Object programs produced by the Assembler and the RPG compiler may not reference any system logical unit except SYSLOG. Object programs produced by the COBOL compiler may not reference any system logical unit except SYSLOG, and may not contain EXHIBIT and TRACE (the output of the EXHIBIT and TRACE statements is on the system logical unit SYSLST). The DISPLAY and ACCEPT statements may be used only for the system logical unit SYSLOG, normally assigned to the 1052 printer keyboard.