File No. 1410/7010-20
Form C28-0318-3

IBM Systems Reference Library

IBM 1410/7010 Operating System (1410-PR-155)

Basic Concepts

This publication describes the IBM 1410/7010 Operating System and its various components. It includes a brief description of the supporting publications for the Operating System and a diagram of their relationship to each other.

The IBM 1410/7010 Operating System is an integrated set of programs and programming systems that provides an IBM 1410/7010 installation with a convenient, efficient means of performing its data processing. This Operating System enables an installation to write, assemble, and execute programs with a minimum of programmer time, machine time, and machine-operator time.

NOTE: The IBM 1302 Disk Storage Unit is now designated the IBM 2302 Disk Storage Unit; there has been no change in the unit itself, in the applications for which the unit may be used, or in the programming parameters used to specify those applications. The IBM 2302 Disk Storage Unit designation has been used in the text of this publication.

MAJOR REVISION (November 1965)
This publication is a major revision of *IBM 1410/7010 Operating System; Basic Concepts*, Form C28-0318-2, and makes that publication obsolete.

Copies of this and other IBM publications can be obtained through IBM Branch Offices.
Address comments concerning the contents of this publication to:
IBM Corporation, Programming Systems Publications, Department 637, Neighborhood Road, Kingston, New York 12401

Contents

Introduction 5
The Need for Monitored Processing 5
Principles and Advantages of Monitored Processing 5

Principles of the IBM 1410/7010 Operating System . 6
Minimum Machine Requirements 6
Creating an Operating System 7
Using the IBM 1410/7010 Operating System 8

IBM 1410/7010 Operating System Components 11
System Monitor 11
Input/Output Control System 12
Random-Processing Scheduler 13
System Generation Programs 13
Tele-processing Supervisor 13
Symbolic-Language Processors 13
Sorting Programs 15
Utility Programs 16
User-Written Programming 17

IBM 1410/7010 Operating System Publications 18

Glossary 21

This publication introduces the basic concepts of the IBM 1410/7010 Operating System and is intended to familiarize users with system concepts and capabilities. As such, the publication is a prerequisite for other publications that describe the Operating System, and it discusses the following:

1. Monitored processing
2. Machine requirements for the IBM 1410/7010 Operating System
3. Creation and use of the IBM 1410/7010 Operating System
4. Functions of the IBM 1410/7010 Operating System components
5. Publications describing the IBM 1410/7010 Operating System in detail
6. Glossary of general terms used with the IBM 1410/7010 Operating System

The Need for Monitored Processing

The utilization of a computer involves many jobs, and each job, in turn, involves set-up time. Although technological development has made significant advances in computer speeds, it can do very little to increase manual speed. Faster computer speeds make the time lost due to manual operations assume an even greater percentage of total processing time. If a monitor-controlled operating system is not used, the computer must stand idle while manual operations are being performed.

Manual operations fall into two general categories: steps involved in job-to-job transition (e.g., locating the next program to be used and loading that program) and steps involved in setting up a specific job (e.g., mounting tape reels and placing cards in a card reader).

Principles and Advantages of Monitored Processing

One advantage of monitored processing is that it automates manual procedures whenever possible; for example, it can conveniently handle the locating and loading of programs. Monitored processing can also overlap with other processing manual procedures that cannot be automated; for example, tape reels for one job can be mounted while another job is still in progress. A monitor keeps the system ready to accept one job after another by providing a standard procedure to handle the scheduling and processing of all computer jobs. In other words, a monitor substitutes *programmed job-to-job transition* for manual transition. Programmed transition is made possible through control-card specifications. The preparation of control cards is independent of the operation of the computer, and neither interrupts computer operation nor wastes valuable computer time.

Another advantage of monitored processing is the flexibility that can be achieved in the assignment of input/output equipment. The monitor can store, within itself, information about all input/output devices in the system. During the day's processing, the monitor can vary input/output assignments in accordance with information provided in a control-card deck. As each program is executed under monitor control, the control-card information can be used to assign symbolic input and output units to devices.

The monitor is the nucleus of the operating system, and a portion of it must remain in core storage at all times. In most operating systems, a pre-written set of input/output routines also remains in core storage at all times. These routines provide every program functioning under the operating system with an efficient method of reading and writing information.

Using these input/output routines, the monitor reads the control cards supplied by the user and executes programs under its control. Thus an operating system consists of:

1. Control programs acting on control-card specifications to direct operation of the system
2. Programs assisting the control programs in performing their functions
3. Programs that will perform a function for the user (e.g., sort and utility programs) or aid him in producing a program to perform a function (e.g., language processors)
4. User-written programs

The nature of the IBM 1410/7010 Operating System and its components is outlined in this publication.

Principles of the IBM 1410/7010 Operating System

The IBM 1410/7010 Operating System includes a set of programs supplied by IBM for use in IBM 1410 or 7010 Data Processing installations. This set of programs is provided to users in a Master file on tape. Some of the programs are in absolute format, i.e., they are ready to be executed. The absolute programs include Bootstrap routines to load the basic Resident Monitor into core storage. The basic Resident Monitor (in absolute format) controls the operations that produce an Operating System tailored to the user's specific requirements.

In addition to the absolute programs, the Master file also includes relocatable programs. These are machine-language programs that do not have fixed machine addresses and can be loaded anywhere in core storage. They can also be combined to form larger programs. Relocatable programs included on the Master file are generalized, so that certain basic program units, or modules, may be selected to form programs tailored to the installation's requirements.

The user selects the programs, or modules, he desires to use in his installation by control cards. He can select the modules directly or indirectly. If he selects them directly, he must specify the module name on a control card. For example, if the user desires magnetic tape as his medium for storing intermediate data in a program, he names tape-oriented modules in control cards. If he selects modules indirectly, he must specify application-oriented key words. These key words serve as input to an intermediate program that selects the required modules. For example, specification of one of the application-oriented key words, or parameters, for the Generalized Tape Sorting Program results in the selection of several modules. These modules, and other modules selected through other parameters, are joined to form the sorting program.

The selection and combination of modules to form programs tailored to an application is performed according to control-card specification. In addition, when programs created by the module selection/combination process are executed, they accept additional control-card directions. This method of forming programs eliminates the need for many user-written programs.

Minimum Machine Requirements

The machine requirements for the Operating System differ according to the medium on which the Operating System programs reside.

For a tape-oriented system, i.e., a system in which the System Operating File (see Glossary) resides on magnetic tape, the minimum machine requirements are:

40,000 positions of core storage
5 magnetic tape units
1 card reader (or an additional magnetic tape unit)

NOTE: It is recommended that a printer or additional tape unit be provided for a system file that is called the Standard Print Unit (SPR).

For a disk-oriented system, i.e., a system in which the System Operating File resides on IBM 1301 or IBM 2302 Disk Storage, the minimum machine requirements are:

60,000 positions of core storage
1 module of IBM 1301/2302 Disk Storage
2 magnetic tape units
1 card reader (or an additional magnetic tape unit)

NOTE: It is recommended that a printer or additional tape unit be provided for the SPR. However, because the two magnetic tape units listed above are required only for generation of the system, they may serve other system functions during the day's processing (i.e., when System Generation is not being performed). One of the tape units could be used in place of the card reader, and the other in place of the printer.

For either tape- or disk-oriented systems, the Processing Overlap and Priority features are required. These are standard features for IBM 7010 Data Processing Systems and special features for IBM 1410 Data Processing Systems.

The machine requirements are greater for certain special programming features of the Operating System. For example, a Tele-processing control system within the Operating System requires an additional 20,000 positions of core storage. All additional requirements are outlined in the publication *IBM 1410/7010 Operating System; System Generation,* Form C28-0352.

Creating an Operating System

The user obtains all of the IBM 1410/7010 Operating System programs on the IBM-supplied Master file, which fits on a single reel of tape. Two forms of the

file are available: one for tape-oriented installations, one for disk-oriented installations. One difference between the two Master files is that the disk-oriented system contains an additional program, the Tape-to-Disk Load program; this program transfers the Master file from tape to disk and can also establish the format of the disk area to be used for system residence.

The first section of a Master file contains the Bootstrap routines used to load the Resident Monitor into core storage. In a tape-oriented system, the Bootstrap routines are the first programs on the Master file; in a disk-oriented system, the Bootstrap routines are preceded only by the Tape-to-Disk Load program.

The basic Resident Monitor follows the Bootstrap routines on the Master file. After the initial execution of the Bootstrap routines, the Resident Monitor — in either basic or expanded form, depending on the user's needs — remains in core storage at all times.

The basic Resident Monitor is followed by the Transitional Monitor, which analyzes control cards, and the Linkage Loader, which transforms programs from relocatable to absolute format. Both the Transitional Monitor and Linkage Loader are in absolute format. Once loaded into core storage, the Resident Monitor loads all other absolute-format programs into core storage as required.

On the Master file, the Bootstrap routines, the basic Resident Monitor, the Transitional Monitor, and the Linkage Loader comprise the first operating section. Following this section is the Relocatable Library, which contains programs and routines of the Operating System in relocatable format.

Following the Relocatable Library is the second operating section, which consists of other absolute-format programs needed to generate the user's particular system. On a tape-oriented Master file, these programs include the following: additional copies of the Transitional Monitor; the Autocoder language processor; two programs, System Generation 1 (sg1) and System Generation 2 (sg2), that are executed to build the user's system; the Create Library, containing packets of pre-written Linkage Loader control cards; and the Sort Definition Program, used to form one or several tape sort or merge programs. The disk-oriented Master file contains the same programs with the following exceptions: the additional copies of the Transitional Monitor are not present; utility programs for establishing the format of disks and for printing the contents of core, disk, and tape storage are included in absolute format; and an additional copy of the Tape-to-Disk Load program is included. For complete lists of the programs included on the Master file for both disk- and tape-oriented systems, see the publication *System Generation.*

The process by which the user creates the specific operating system for his installation is called System Generation. If the user desires a Master file for his particular installation, he must first create the System Generator File (SGF). This tape or disk file contains components of the Operating System, selected from the Master file, that are needed to satisfy the total processing requirements of a specific installation. The SGF may also contain user-written programs.

After an SGF is created, the IBM Master file need not be used again because the SGF becomes the Master file for the installation. Thus, the user must be careful in choosing the components to be included on the SGF; he must keep in mind both the machine configuration of his installation and his over-all processing requirements.

From the SGF, the user can create any number and variety of System Operating Files (SOF's). Each SOF contains components of the SGF (e.g., the Resident Monitor) and user-written programs. Each SOF can be oriented to a specific class of processing operations.

For example, if an installation's processing requirements can be divided into two general areas, such as commercial processing and scientific processing, two SOF's are created. One SOF contains only those programs needed to perform commercial operations. The other SOF contains programs necessary for scientific processing. If the user's SGF contains the COBOL and FORTRAN compilers, he should place the COBOL compiler on the commercial SOF and the FORTRAN compiler on the scientific SOF. As with the generation of his SGF, the user must carefully analyze his processing requirements to obtain maximum operating efficiency from his SOF's.

Figure 1 illustrates the creation of the SGF. The input to this operation is the IBM Master file (or an old SGF to be updated) and the user-supplied control information. This control information specifies the Master-file components to be included on the new SGF.

The creation of an SOF from the newly created SGF is illustrated in Figure 2. The control information supplied determines which components of the SGF and which user-written programs are to be placed in the SOF. The operation illustrated in Figure 2 is repeated for each SOF the user wishes to create.

Complete SOF's can also be created directly from the IBM-supplied Master file. This procedure bypasses the SGF step.

Using the IBM 1410/7010 Operating System

Once the Operating System has been created, user-written source programs can be compiled (i.e., trans-

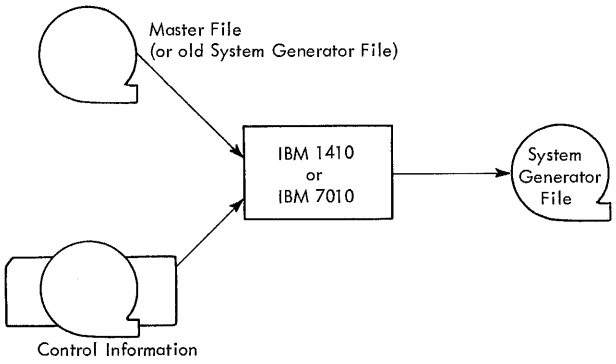

Figure 1. Creation of the SGF

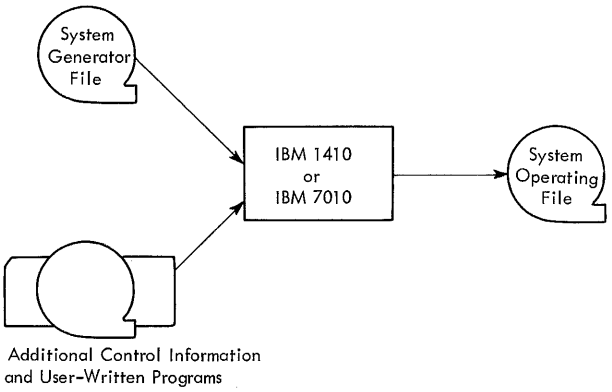

Figure 2. Creation of an SOF

lated from symbolic-language source programs to machine-language object programs). The source programs can be written in either the Autocoder, FORTRAN, or COBOL symbolic languages. The resultant object programs can be executed under control of the System Monitor.

The term System Monitor embraces the basic Resident Monitor or the expanded Resident Monitor (with the Input/Output Control System), the Transitional Monitor, and the Linkage Loader. The System Monitor, acting on control information provided by the user, exerts control over all programs used within the Operating System during the day's processing. The Resident Monitor provides the routines and information that can be used by components of the system.

After a program is compiled, it is in relocatable format; in this format, the program has addresses that have been assigned relative to some arbitrary reference point, usually the beginning of core storage. Before the program can be executed, the relocatable addresses must be converted to the actual core-storage locations occupied by the program during execution. The program must be located starting at some point beyond the area occupied by the Resident Monitor and the IOCS. The Linkage Loader determines the size of these resident components and changes the addresses of the relocatable program accordingly. This change of address makes a relocatable program an absolute one. Once created, an absolute program can be loaded into the appropriate portion of core storage and executed from it.

Relocatable programs can reside on tape or disk as modules in the Relocatable Library. The Relocatable Library can be placed on a file known as the System Library File (LIB), or on the SOF.

If a Tele-processing (TP) system is used, relocatable programs can also reside on the TP Library File (MLT); these programs process Tele-processing inquiries. The MLT, however, must contain all relocatable programs or all absolute programs, but not a mixture of the two types.

In addition to being contained in the libraries, relocatable programs (modules) may also be located on two other files used by the Operating System: the Go file (MGO) and the Standard Input Unit (SIU). Programs that have been translated into machine language by the compilers are placed on the Go file and taken directly from that file for processing by the Linkage Loader. Under this "compile-and-go" operation, a program can be compiled, be transformed from relocatable to absolute format, and be loaded and executed in one smooth operation; this operation requires no intervention by the programmer or machine operator. The user can also place previously compiled relocatable programs on the SIU.

In addition to the SOF, which includes absolute programs processed during System Generation, two other files can store and supply absolute programs during the day's processing: the Job file (MJB) and the TP Library file.

When the Linkage Loader processes a relocatable program placed on the Go file or on the SIU, it writes the program in absolute format on the Job file. The Job file, then, supplements the SOF as an absolute-program file. Job files can be created for one-time use, or can be saved for use in the repeated execution of the programs they contain.

The programs that the user has written to serve his Tele-processing requirements are placed on the TP Library file; as indicated above, this file can contain either absolute or relocatable programs.

In its processing, the Linkage Loader can combine relocatable programs (from the libraries, the Go file, and/or the SIU) into a single absolute program; this facility for combining separately written programs gives the user considerable flexibility in creating programs suited to his particular applicational needs.

Thus, during the day's processing, machine-lan-

guage programs can reside on the following system files:

1. SOF, in absolute format
2. SOF, in relocatable format
3. LIB, in relocatable format
4. MJB, in absolute format
5. SIU, in relocatable format
6. MGO, in relocatable format
7. MLT, in relocatable or absolute format (according to the user's specification at System Generation)

All absolute programs — whether found on the SOF, the Job file, or the TP Library file — are executed under control of the Resident Monitor, which is always in core storage when the Operating System is being used. By use of control cards, the user directs the Operating System to perform the tasks he desires. These control cards are analyzed by the Transitional Monitor, which is automatically loaded and executed when a control card is read.

Figure 3 illustrates the information flow for the execution of a single absolute program located on the SOF. The user specifies, in control cards placed in the SIU, the program to be executed. The Resident Monitor, already in core storage, loads the Transitional Monitor to analyze the control cards. After the Transitional Monitor determines the program required and locates it, the Resident Monitor loads and executes that program. The user may supply data on a separate file, or he may place that data following the control cards on the SIU. Output from the absolute program being executed may be placed on the Standard Print Unit (SPR), on the Standard Punch Unit (SPU), or on a separate (data) file.

NOTE: Control information and data can both be on the SIU. Similarly, the SPR and SPU can share the same magnetic tape unit.

Figure 4 illustrates the flow of information for the combination of two relocatable programs, Program A (found in the LIB) and Program B (found on the SIU), and the execution of the resultant absolute program. The user specifies in control cards the task to be performed; these control cards are placed in the SIU, preceding Program B. The Resident Monitor loads the Transitional Monitor from the SOF to analyze the control cards. The Resident Monitor then loads the Linkage Loader from the SOF; the Linkage Loader takes Program A from the LIB and Program B from the SIU and forms Program AB, in absolute format, which it places on the Job file. The Resident Monitor then loads Program AB from the Job file and executes the program. Input data for Program AB can reside on a separate file or on the SIU (following the control cards and Program B). Program AB may place output on the SPR, the SPU, or on a separate (data) file. The SPR and SPU can share the same magnetic tape unit.

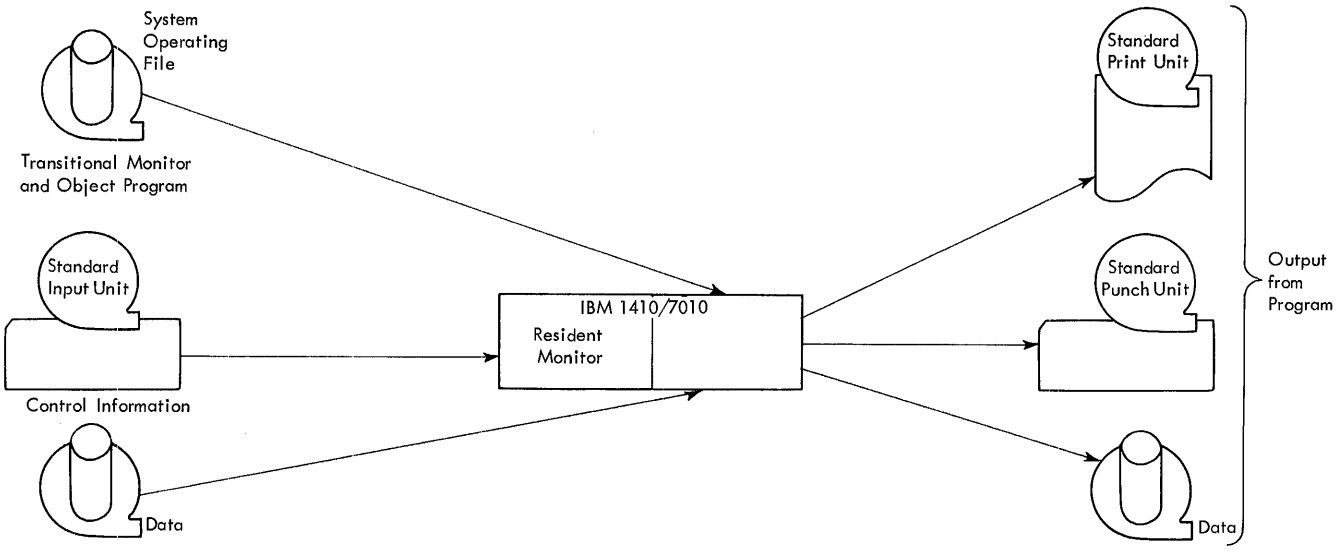

Figure 3. Data Flow for Execution of a Single Absolute Program

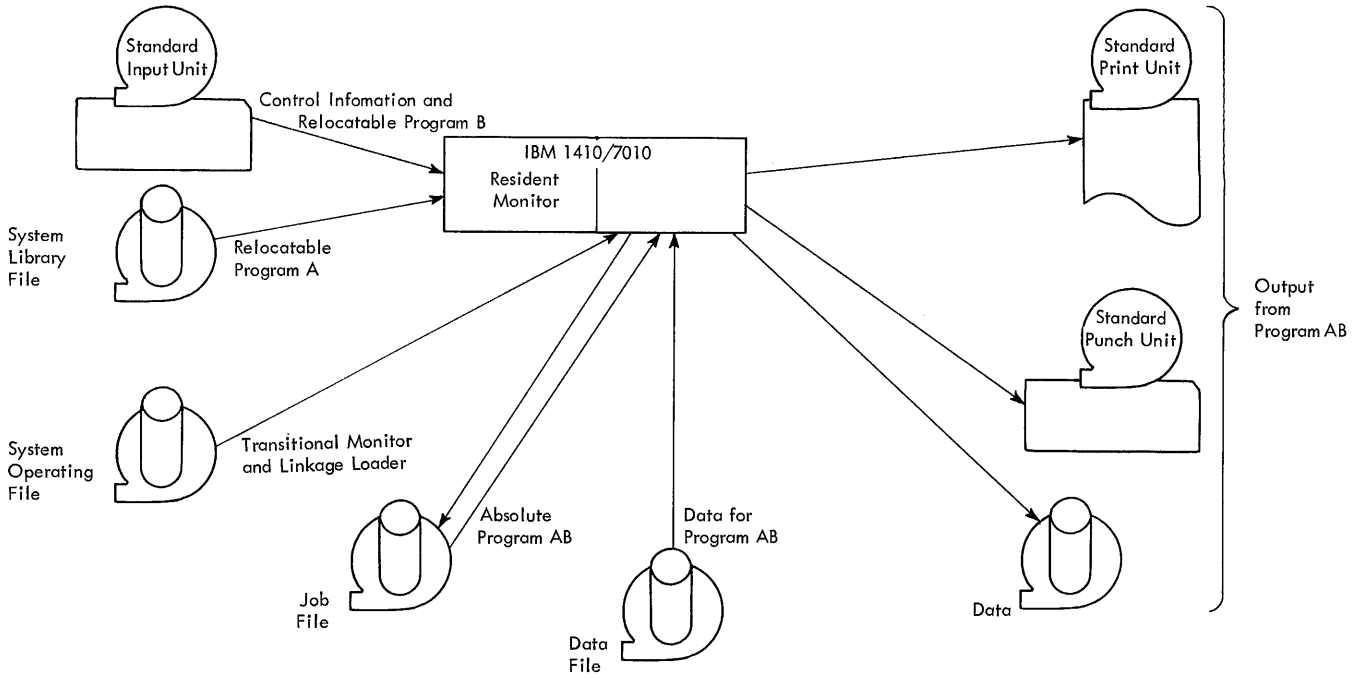

Figure 4. Data Flow for Combining Two Relocatable Programs and Executing the Resultant Absolute Program

A component of the IBM 1410/7010 Operating System is any program operating under control of the Resident Monitor. There are two main types of components:

1. IBM-supplied programs (e.g., the Linkage Loader) or programs generated as a result of the interaction of user specifications and IBM-supplied programs (e.g., a generated sorting program)
2. User-supplied programs (e.g., a program to schedule inter- and intra-plant relocation of employees)

Figure 5 illustrates the relationships among Operating System components.

The structure of the system is such that user-written processing programs can easily become an integral part of the system. User-written modification routines can also be incorporated into several IBM-supplied components (e.g., Basic Input/Output Control System, and the Generalized Tape and Disk Sorting programs). All components of the Operating System interact efficiently to handle an installation's data processing requirements.

The Resident Monitor, an IBM-supplied component, exerts control over the entire Operating System and contains routines and information that can be used by components of the system. Resident with the Monitor is the Input/Output Control System and, if specified, the Tele-processing Supervisor. These resident components of the system serve as an interface for the other Operating System components.

Each of the other components is called into core storage as required, and is executed by the Resident Monitor.

All of the Operating System components supplied by IBM are discussed briefly in the paragraphs that follow.

System Monitor

The System Monitor is the nucleus of the Operating System. The Monitor coordinates the operation of all other Operating System components, including user-written programs functioning under the system. Three programs make up the System Monitor: Resident Monitor, Transitional Monitor, and Linkage Loader.

Resident Monitor

The Resident Monitor remains in core storage at all times while the Operating System is functioning. This program reads control cards, loads other programs into core storage, assists the Transitional Monitor in processing input/output unit assignments supplied by the user, and coordinates end-of-program operations. The Resident Monitor also contains a Communication Re-

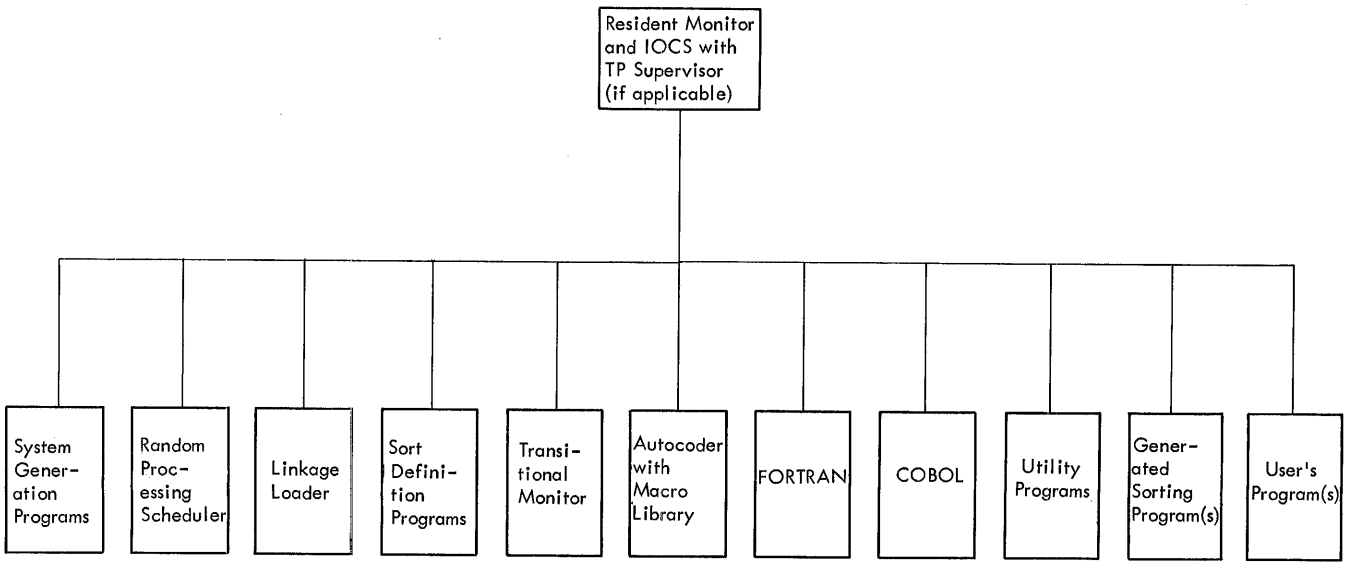

Figure 5. IBM 1410/7010 Operating System

gion through which information can be passed among Operating System programs.

As the supervisor of and interface for the Operating System, the Resident Monitor loads one program into the computer and gives control to that program. When that program is executed, control returns to the Resident Monitor, which loads the next program, etc. In this manner, a complete *batch* of jobs is processed without the stopping of the computer, and without the necessity of operator intervention.

Transitional Monitor

The Transitional Monitor, an extension of the Resident Monitor, analyzes the control cards that describe functions to be performed, and assists the Resident Monitor in processing any required input/output unit assignments specified in the control cards. Unlike the Resident Monitor, the Transitional Monitor does not remain in core storage at all times; it is loaded between jobs and between runs within a job. After the Transitional Monitor has analyzed the control cards and located the specified program, it returns control to the Resident Monitor so that the program can be loaded and executed.

Linkage Loader

The Linkage Loader reads relocatable programs from the Go file, one of the libraries, and/or the SIU, and transforms these programs into absolute format. The absolute programs produced are placed on the Job file for subsequent loading and execution by the Resident Monitor. In converting programs from relocatable to absolute format, the Linkage Loader also inserts machine addresses in place of inter-program communication symbols known as linkage and system symbols.

Input/Output Control System

The Input/Output Control System (IOCS) is a set of pre-written routines that perform the input/output functions for all programs run under the Operating System. These routines provide an efficient means of scheduling, implementing, and controlling the transfer of data between input/output devices and core storage. During System Generation, the user defines the IOCS necessary for his particular installation. The selected IOCS, a resident component, remains in core storage whenever the Operating System is used.

At System Generation time, the user specifies the device-dependent and/or function-dependent features his particular IOCS is to provide. For example, the user indicates that he wishes the IOCS to handle disk input/output operations (device-dependent) or that he wishes the IOCS to process labeled tape files (function-dependent). As a result of System Generation, an IOCS that provides routines to handle exactly those features the user desires is placed on his SOF. If the user wishes to alter the device-dependent or function-dependent features of his IOCS, he creates a new SOF incorporating routines to handle the new features.

The IOCS is capable of reading data records or blocks of data records from the IBM 1402 Card Read Punch, IBM 1442 Serial Card Reader, IBM 1011 Paper Tape Reader, IBM 729 and IBM 7330 Magnetic Tape Units, and IBM 1301/2302 and IBM 1311 Disk Storage. It can write data records or blocks of data records on the IBM 1402 Card Read Punch, IBM 1403 Printer, IBM 729 and IBM 7330 Magnetic Tape Units, and IBM 1301/2302 and IBM 1311 Disk Storage. The IOCS controls the overlap of these read/write operations with normal processing operations. Error conditions are detected and automatically corrected or are brought to the attention of the machine operator. Required and optional exits to user-written routines (e.g., end-of-file and tape-label exits) are provided.

Once the user has his IOCS on his SOF, he directs that IOCS by statements in his programs. For Autocoder programs, he issues these directions by file-description statements and macro-instructions. The file-description statements (Define the File statements) provide information necessary to the IOCS for processing the file. The macro-instructions are translated by the Autocoder processor into machine-language instructions that pass control to the IOCS to perform the desired operations.

For FORTRAN or COBOL programs, the user issues directions to the IOCS by statements that are a part of those languages (e.g., READ or WRITE). The FORTRAN and COBOL processors translate these statements into machine-language instructions that pass control to the IOCS to perform the desired operations.

As an adjunct to its normal functions, the IOCS handles the processing necessary to implement the Simultaneous Peripheral Operations On Line (SPOOL) feature. SPOOL is a feature that enables the user to perform peripheral operations (e.g., tape to printer) concurrently with batch processing. SPOOL permits not only simple peripheral operations but also peripheral operations incorporating a user-written editing routine.

In addition to the IOCS program that is always resident when the Operating System is in use, IBM provides another program designed to aid in the efficient handling of input/output operations: the Random Processing Scheduler.

Random Processing Scheduler

The Random Processing Scheduler, an optional component of the Operating System, assists the user in coordinating input/output activity for random-access input/output devices, e.g., IBM 1301 Disk Storage. By use of the Scheduler routines, the user may operate on randomly organized data in a "read a record, process it, write it" method as if the data were serially organized.

The Scheduler is of maximum value for applications involving the updating of files in disk storage, where the data used is neither batched by type nor stored beforehand.

By providing routines for the supervision and concurrent scheduling of input/output operations in a random-processing application, the Scheduler relieves the user of the task of designing an efficient random-processing input/output control system.

The Random Processing Scheduler is not a part of the resident IOCS, nor is it executed as a dependent program, as a user-written program might be. The Scheduler routines, located in the Relocatable Library, are incorporated into the user's program by the Linkage Loader.

System Generation Programs

The System Generation capabilities of the Operating System are provided by two programs, SG1 and SG2; these programs are used in conjunction with the Autocoder Processor and the System Monitor.

A third program, SG3, provides the ability to update and utilize source-language programs contained on a History file. The History file, supplied by IBM on request, contains the Autocoder-language programs for all Operating System components. The user may place his programs on the History file if he wishes.

Through System Generation, the user is able to choose from the components of the Operating System that IBM provides and create SOF's and libraries that are designed to accomplish only those functions that he desires. In addition to the IBM-supplied programs, user-written programs are also incorporated into SOF's and libraries by the System Generation process. The System Generation programs also enable the user to list directories of the contents of his SOF's or the Macro Library (a part of the Autocoder Processor).

Tele-processing Supervisor

The Tele-processing Supervisor, an optional component of the Operating System, controls all programming related to Tele-processing devices. If the user's installation includes Tele-processing devices, and the required 20,000 additional core-storage positions are available, the Tele-processing Supervisor can be incorporated into the system. The Supervisor, a resident component, becomes part of the Resident Monitor and thus remains in core storage while the Operating System is used.

The Supervisor is created via control-card specification at System Generation. Each user can create a configuration of the Supervisor that contains only those routines needed to support the particular machine configuration of the installation.

The user can create a Supervisor with facilities to support input/output for any combination of the following devices: IBM 7750 Programmed Transmission Control; IBM 1440 Data Processing System with an attached IBM 1448 Transmission Control Unit; and IBM 1414 Input/Output Synchronizer, Models 4 or 5, with the IBM 1009 Data Transmission Unit, IBM 1014 Remote Inquiry Unit, IBM 1050 Data Communications System, and/or Telegraph terminal units attached.

In executing input/output routines, the Supervisor utilizes the IOCS. Two other types of programs are used in Tele-processing operations: the TP programs and the Executive program. The TP programs are user-written routines that process inquiries from Tele-processing devices and generate a response message, if required. TP programs can remain in core storage or be loaded, when needed, by the Tele-processing Supervisor. The Executive program, also user-written, serves as a liaison between the Supervisor and the TP programs. The three elements are known collectively as the TP complex of an installation.

The coordinated functioning of these three elements provides an efficient system for processing messages from Tele-processing devices.

Symbolic-Language Processors

A symbolic-language processor is a program that translates a user's source-program statements (written according to the specifications of the symbolic language for that processor) into machine-language instructions. The symbolic-language processor must always be considered in conjunction with the symbolic coding language for which it is used; these two programs are known collectively as a programming system.

As in any language, certain rules must be followed to make communicated information meaningful. In a programming language, these rules are the specifications that the programmer must adhere to in order for the processor to do its task. The use of any programming language reduces the programmer time needed to write and debug programs mainly because of the symbolic or mnemonic nature of the language. The ability to use a single source-language instruction or statement to produce (or generate) several machine-language instructions further reduces programming time. Because these processor-generated state-

ments have been pretested and debugged, additional time is saved for the programmer.

The programming systems available with the IBM 1410/7010 Operating System are Autocoder, FORTRAN, and COBOL. The machine-language output of these processors is in relocatable format. These relocatable programs, or subprograms, can then be processed by the Linkage Loader and executed whenever necessary. The relocatable format of the language-processor output permits independently compiled programs to refer to locations in other programs; the references are resolved when the Linkage Loader is executed. A program independently compiled by one language processor can be used as a subprogram in a program compiled by another language processor. For example, a subprogram written in Autocoder can be incorporated into a FORTRAN program.

Autocoder

Autocoder is a programming system which, by use of mnemonics, enables the user to create a wide variety of programs through a convenient method of coding.

The Autocoder processor is used during System Generation as well as during the day's processing. The processor resides in absolute format on the IBM Master file; thus, at System Generation, user-written routines can be incorporated into the first System Generator File (SGF). System Generation functions also make wide use of the macro-instruction capability of the Autocoder processor; through this capability, a single symbolic-language statement can generate more than one machine-language instruction. The contents of the individual instructions, and the combinations of the instructions generated, can be varied according to the parameters specified.

The Autocoder language consists of fixed mnemonics and variable user-supplied mnemonics. Programs written in this mnemonic, or symbolic, language are translated into machine language by the Autocoder processor. Some of the symbolic statements of the language generate a single machine-language instruction; these are known as one-for-one statements. Macro-instructions can also be used.

Parameters specified by the programmer in a macro-instruction cause Autocoder to take statements from the Macro Library (on the IBM Master file) and generate machine-language instructions. Macro-instructions enable the programmer to describe the functions he wishes performed, rather than describe each operation within a function, as is necessary in one-for-one coding.

Each source-program symbolic instruction, whether it be a one-for-one instruction or a macro-instruction, may consist of as many as three elements. These elements are: (1) the name of the statement (the *label*); (2) the operation to be performed by the statement (the *operation code*); and (3) entries such as symbolic addresses, actual addresses, literals, special symbols, etc. (the *operands*). Labels and operands are variable; the user can create his own, although they must conform to the rules of the language. Operation codes are fixed; the user selects the symbolic code representing the operation to be performed. The combination of characters in the symbolic operation codes makes it easy for the programmer to recollect the function to be performed. For example, the Autocoder symbolic operation code ZA, signifying the Zero and Add operation, is more readily associated with the instruction than is the machine-language operation code (a question mark). In his choice of symbolic addresses, the programmer may also employ this mnemonic principle.

By using symbolic addresses, the programmer can assign a name to a specific location or area in core storage, and refer to that symbolic name elsewhere in his program. The processor generates the necessary machine-language addresses.

By assigning a symbolic address or name to a data field or routine that is readily associated with that name, the programmer can impart to his program structural clarity and ease of reference that is impossible in machine-language coding. For example, an area containing an employee's salary amount could be named SALARY, and a routine for computing net pay could be named COMPUTEPAY.

Assembly listings are produced for all programs compiled by the Autocoder processor. These listings show the source program (symbolic instructions) in relation to the object program (machine-language instructions). Labels are shown in relation to core-storage assignments, symbolic operation codes in relation to the machine-language operation codes, etc. The listing is arranged in a format that permits easy reference. In addition, the coding errors detected by the processor are indicated in the listing; this aids substantially in program debugging.

FORTRAN

The FORTRAN (*FOR*mula *TRAN*slation) programming system is designed for the efficient production of programs that solve scientific and engineering problems.

The user states his problem solutions in the FORTRAN language — a language much like algebra — and then uses the FORTRAN processor (or compiler) to translate the FORTRAN programs into machine-language programs. The language and the processor are known collectively as the FORTRAN programming system.

In using the FORTRAN language, the scientist or engineer (or programmer) can operate in a familiar frame of reference, that of basic algebra. As in algebra, the user of FORTRAN defines constants and variables and manipulates them through the use of equations; these equations, written in the form of FORTRAN statements, are the heart of the language. Supplementing these equation statements are several other types of statements: (1) input/output, to enable the user to get data into and out of the computer; (2) control, to give the user the ability to explicitly dictate the order in which his equations are evaluated; (3) subprogram, to allow the user to write programs that are used repeatedly by other programs (thereby eliminating the necessity to restate the solution to a common problem for each application); and (4) specification statements, to enable the user to control the allocation of core storage.

In addition to the subprograms that the user can write and place on the Relocatable Library, there are IBM-supplied subprograms for many standard mathematical functions; these subprograms, when named in the user's FORTRAN program, are incorporated into his machine-language program by the Linkage Loader. Examples of the subprograms supplied are those for calculating (1) the absolute value of a quantity, (2) the trigonometric sine of an angle measured in radians, and (3) the square root of a quantity.

The FORTRAN compiler translates each statement the user writes into one or more machine-language instructions; thus, the FORTRAN programming system is a higher-level language comparable to the macro-instruction portion of the Autocoder language.

In compiling each FORTRAN source program, the FORTRAN processor diagnoses errors in that program; it generates a listing of the source program, noting each error detected in the diagnosis. With these diagnostic error messages as a guide, the FORTRAN user can correct his program without becoming involved in debugging machine-language code.

COBOL

COBOL (*CO*mmon *B*usiness *O*riented *L*anguage) is a programming system suitable for solving business and commercial problems. The COBOL programming system consists of the COBOL language — a language patterned after business English — and the COBOL processor (or compiler); the compiler translates COBOL-language programs into machine-language programs.

Users of the COBOL language can state problem solutions in business terms without any regard to the intricacies of symbolic one-for-one, or machine-language programming. COBOL programs are divided into such familiar components as sentences and paragraphs, thereby imparting structural clarity to the source program. COBOL verbs such as READ, ADD, EXAMINE, PERFORM, and WRITE enable the COBOL user to create programs that read his data, manipulate it as per his instructions, and write out the processed data.

As in FORTRAN, each statement encoded in the COBOL language generates one or more instructions in the resultant machine-language program; thus, each COBOL statement is similar to an Autocoder macro-instruction.

As it processes each COBOL source program, the COBOL processor diagnoses that program; it generates a listing of the source program, noting each error detected. With these error messages as a guide, the COBOL user can debug his program without becoming involved in machine-language coding.

Sorting Programs

IBM provides two generalized sorting programs on the Master file: the Generalized Tape Sorting Program, and the Generalized Sorting Program Using IBM 1301/2302 Disk Storage (Generalized Disk Sorting Program). The tape sorting program uses magnetic tapes for input, output, and work files; the disk sorting program can use either tape or disk for input and/or output, but must use disk for work files. Both programs can also accept input from the Standard Input Unit, if specified.

Generalized Tape Sorting Program

The Generalized Tape Sorting Program enables the user to create, through control-card specification, different magnetic tape sorting and merging programs; each program created can accept a wide range and combination of specification values. With these values, entered at execution time via control cards, the program alters itself to fit the needs of the particular application for which it is used.

The user can create sort or merge programs by combining the following features: sort or merge fixed- or variable-length records, sort or merge on one or several control data fields, unmodified or user-written modification routines, and ability to reproduce itself on punched cards. Each program produced can sort in ascending or descending order; use a balanced or unbalanced merge; write checkpoint records; accept input from the SIU, if specified; use tape label-processing routines; etc. Linkage symbols to specific routines within the sorting or merging program are provided. The user can refer to these symbols if he includes his own modification routines.

On the Master file the Generalized Tape Sorting Program exists as part of the Relocatable Library; the program consists of a set of relocatable subroutines

and a separate routine called the Sort Definition program. The Sort Definition program also resides in absolute format on the Master file for a tape-oriented system. The Sort Definition program selects those sort subroutines required to form the type of sort programs desired by the user. Execution of the Sort Definition program and conversion of the selected subroutines to absolute format can be performed at System Generation or during the day's processing. The entire process of producing executable sort or merge programs, from source program time through object program time, occurs in two main steps:

1. The user supplies the Sort Definition program with information that enables it to select those subroutines necessary for the sort or merge program desired. An installation can create several different sort or merge programs. The Linkage Loader produces the sort and merge programs in absolute format.
2. At object program time, the sort or merge program, acting on control-card information supplied by the user, alters itself to meet the requirements of the specific application. The program is then executed.

Generalized Disk Sorting Program

The Generalized Disk Sorting Program enables the user to create, through control-card specification, different sorting programs that use IBM 1301 or 2302 Disk Storage for work files during the sort. Like the Generalized Tape Sorting Program, each disk sort program produced can accept a wide range and combination of specification values. With these values, entered at execution time via control cards, the program alters itself to fit the needs of the particular application for which it is used.

The user can create disk sorting programs by using various combinations of the following features: magnetic-tape or disk input, magnetic-tape or disk output, sort fixed-length or variable-length records, sort on one or several control data fields, unmodified or user-written modification routines, and ability to reproduce itself on punched cards. Each program produced can sort in ascending or descending order; accept input from the SIU, if specified; utilize the Write Disk Check option, if desired; write check-point records; accept Form A, Form C, or Form G disk files; use tape-label processing routines, if magnetic tapes are used for input/output; etc. If the user includes his own modification routines, he can refer to linkage symbols provided for specific routines within the sorting program.

On the Master file the Generalized Disk Sorting Program exists as part of the Relocatable Library. The program consists of a set of relocatable subroutines and a separate routine called the Sort Definition program. The disk Sort Definition program also resides in absolute format on a disk-oriented Master file. The operational principle of the disk Sort Definition program is the same as that for the tape Sort Definition program. The time of execution of the tape Sort Definition program, and the two main steps in creating tape sort programs (described above), also apply for disk. However, no merging programs can be produced with the Generalized Disk Sorting Program.

Utility Programs

The Utility programs included in the Operating System provide facilities for program testing, output-area preparation for disk, and many basic service functions needed in the daily operations of a computer installation. The user adapts these programs for his own needs by control-card specification. Program testing and medium preparation time are substantially reduced.

There are seven utility programs: Snapshot, Storage Print, Tape Print, Disk Print, IBM 1301 Disk Format/Address Generator, IBM 1311 Disk Format/Address Generator, and IBM 1311 Disk Label program. The four print programs (Snapshot, Storage Print, Tape Print, and Disk Print) use the same basic print format: 100 characters per line, in groups of 10. Output of the print programs is on the SPR. If the user's SPR is a magnetic tape unit, he can subsequently print the listings using the Peripheral Output Writer (POW) program on an IBM 1401 Data Processing System. POW is a 1401 print and punch program that may be written as the first record on a tape SPR (or a tape SPU) if the user specifies the inclusion of this program at System Generation.

Snapshot

The Snapshot utility program prints all or selected areas of core storage at intervals specified by the user during the program run. It is used primarily for testing programs, although the System Monitor can combine it with the user's program. Information is printed as it appears in core storage, with word marks indicated above the appropriate characters.

Storage Print

The Storage Print program prints all or selected areas of core storage, previously written on a file called the Core Image file (MDM), in accordance with control information provided by the user. Data is read from the MDM and edited. (If the Storage Print program is used, the MDM must be assigned as a system file.)

The Storage Print program is executed only when specifically requested by user-supplied control cards. The printed listing appears in machine language on the SPR; it displays the contents of the specified area of core storage and includes the following: identification and status information, a line containing pertinent system symbols and their absolute-address equivalents, and the contents of the first 100 positions of core storage. Error and diagnostic messages can also appear in the listing.

Tape Print

The Tape Print program prints all or a portion of the contents of a tape reel. The program can print the entire contents of one or more files, or a specified number of data records within a file. Records on the tape can be in fixed-length or variable-length format. Printed listings, which appear on the SPR, can show word separator characters as separate characters or as word marks above the appropriate characters. In addition to the contents of the tape, the listing includes identification information and record and character counts. Error and diagnostic messages also appear in the listing.

Disk Print

The Disk Print program prints all or a portion of IBM 1301 or 2302 Disk Storage, or an IBM 1311 Disk Pack. Data from the disk can be read in Load or Move mode and records on the disk can be in fixed-length or variable-length format. The printed listing appears in machine language on the SPR. In addition to the contents of the disk area specified, the listing contains identification information and character counts. Error and diagnostic messages can also appear in the listing.

IBM 1301 Disk Format/Address Generator

The IBM 1301 Disk Format/Address Generator program provides formatting and address generation separately or in the same run, according to the user's specifications. For generation of standard addresses, the user need only provide control-card information. Nonstandard addresses are not generated, only written, by the program; the user must supply his own routine to generate the addresses. He can use this program for formatting and generating addresses for an entire disk module, or any part of a disk module. The Write Disk Check option can also be used. Error and diagnostic messages appear on the SPR.

IBM 1311 Disk Format/Address Generator

The IBM 1311 Disk Format/Address Generator can be used to (1) write sector addresses for all or a portion of a 1311 Disk Pack in standard or nonstandard sequence; and (2) clear the data portions of a specified area and fill the area with a specified character (e.g., blank). Various types of addressing and various combinations of addressing and clearing can be specified. The Write Disk Check option can be used. If an error occurs, a special end-of-program condition is set in the Resident Monitor; a numeric console message is issued and the particular error message is printed out on the SPR.

IBM 1311 Disk Label Program

The IBM 1311 Disk Label program provides a means of establishing, maintaining, and altering the identifying label track portion of a disk pack. The program can be used to establish or remove labels on the entire label track, enter a new header label, delete an existing header label, change an existing header label, or print labels. If an error occurs during the execution of the program, a special end-of-program condition is set in the Resident Monitor; a numeric console message is issued and the particular error message is printed out on the SPR.

User-Written Programs

An integral part of an installation's Operating System is the set of programs and routines written by that installation for use under control of the Operating System. By employing the language processors (Autocoder, FORTRAN, and COBOL) and other IBM-supplied components of the Operating System, the installation can develop specific programming to perform its application-oriented processing.

User-written programs fit into two categories:

1. Independent programs that are at an equal rank with IBM-supplied components of the Operating System. These programs (e.g., those that perform classroom scheduling for a university or those that perform customer billing for a public utility) are named in control cards and are executed under control of the Resident Monitor just as is, say, the Autocoder language processor.

2. Routines that are used with programs supplied by IBM or generated by IBM-supplied programs; e.g., a routine incorporated into a generated sorting program for deleting certain records from the output file.

IBM 1410/7010 Operating System Publications

An integrated set of publications provides the detailed documentation for the IBM 1410/7010 Operating System. Based on the principles outlined in this publication, each publication in the set describes a specific facet of the Operating System. In addition, each publication lists prerequisite reading and recommends other applicable publications.

Figure 6 summarizes, in schematic form, the set of publications for the Operating System. To determine the prerequisite reading for a publication, follow the

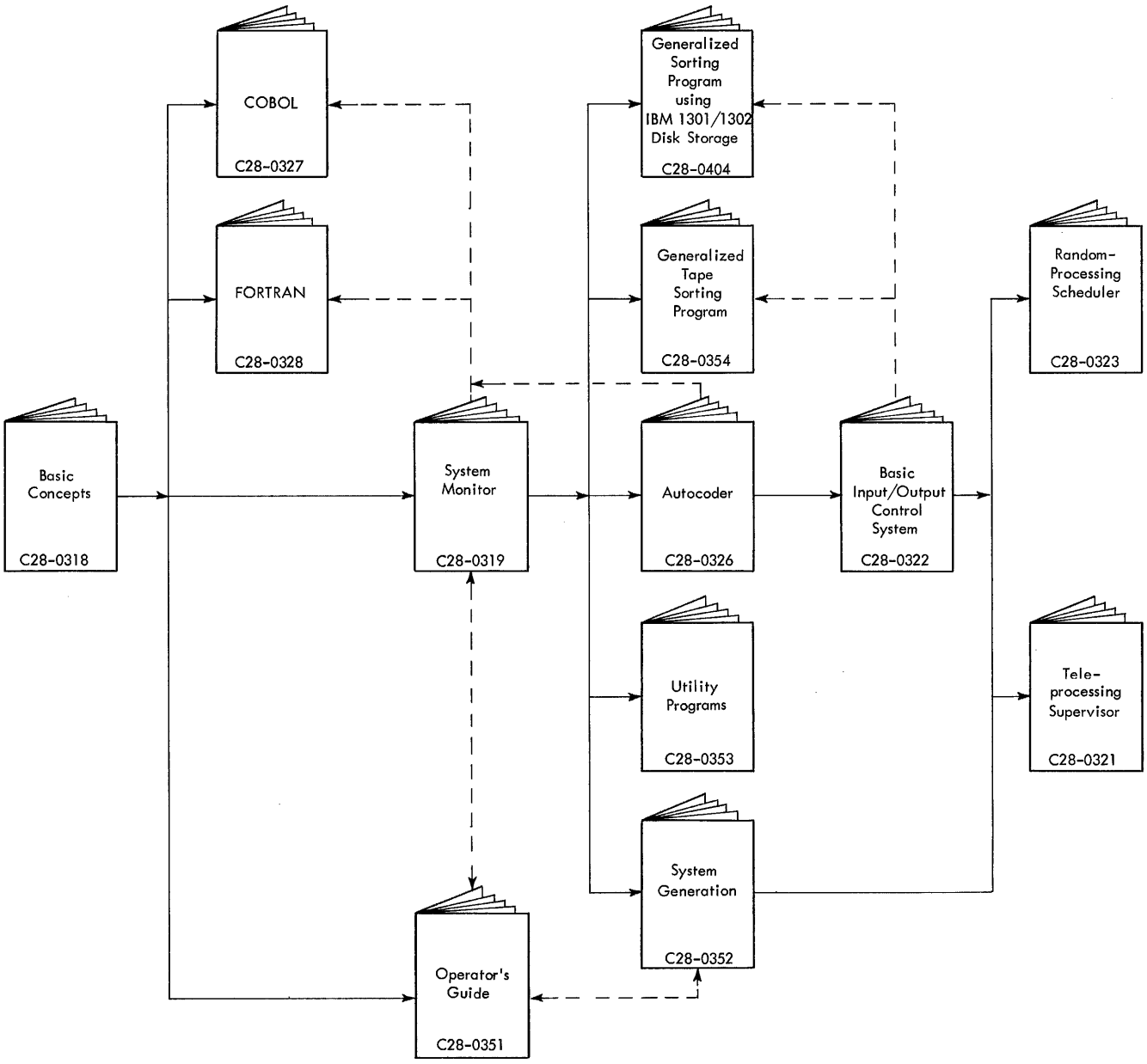

Figure 6. IBM 1410/7010 Operating System Publications

solid-line path(s) from that publication back to the *Basic Concepts* publication; all publications encountered along the solid-line path are *prerequisite.* The broken-line paths in the figure lead the reader to recommended publications. For example, *Basic Concepts* and *System Monitor* are prerequisite reading for *Generalized Tape Sorting Program; Basic Input/Output Control System* and its prerequisite, *Autocoder,* are recommended reading for *Generalized Tape Sorting Program.*

The paragraphs that follow briefly summarize the contents of the Operating System publications set; reference manuals in the set are discussed individually, but programming systems analysis guides in the set are discussed as a group. The first line of each title listed is *IBM 1410/7010 Operating System.* (This line is not repeated for the following publications.)

System Monitor (C28-0319) describes the principles and functions of the Resident Monitor, the Transitional Monitor, and the Linkage Loader, and the relationships of these elements to other Operating System components. Reference material included in the publication details the control cards required for using the Operating System, the linkage sequences that must be encoded to use Resident Monitor subroutines, the messages written by the Monitor, and the console inquiries accepted by the Monitor.

Tele-processing Supervisor (C28-0321) is a supplement to *System Monitor* for installations that use Tele-processing equipment. The Tele-processing publication describes the basic principles of the Tele-processing Supervisor and explains how to write programs to run under control of the Supervisor.

Basic Input/Output Control System (C28-0322) contains the information necessary to make efficient use of the IOCS for handling input/output and control operations in user's programs. Included are discussions of file-definition statements and macro-instructions used with the IOCS; specifications for user-written programming executed from special exits within the IOCS are also presented. The publication is divided into two parts: the first deals with basic use of the IOCS, the latter with the extended use of the IOCS.

Random-Processing Scheduler (C28-0323) provides a discussion of random-processing concepts as they pertain to the Scheduler, and presents detailed information on using the Scheduler for handling input/output operations for random-access devices.

Autocoder (C28-0326) is a reference manual for writing programs in the Autocoder language. The publication weaves together the basic concepts of a symbolic language and the specific details necessary to write Autocoder programs to be compiled and executed under the Operating System. Also included are details on the Autocoder macro-instruction capability with information needed to write additional macros.

COBOL (C28-0327), a reference manual for business personnel and programmers, is designed for use in conjunction with *IBM General Information Manual; COBOL,* Form F28-8053. The general information manual contains basic data about all COBOL programming systems; the reference publication contains additional specifications and instructions for using the Operating System COBOL.

FORTRAN (C28-0328), a reference manual for scientific/engineering personnel and programmers, explains how to write programs in the FORTRAN language and how to write Autocoder subprograms to be used with FORTRAN programs. Also presented are lists of error messages produced by the FORTRAN processor and by FORTRAN object programs.

Operator's Guide (C28-0351), intended for use at the console, explains all operational aspects of the Operating System. Operating information includes discussions of loading procedures, restart procedures, control-card decks, input/output unit assignments, and use of console inquiries. Also included are lists of messages produced by all Operating System programs.

System Generation (C28-0352) describes the building and maintenance of systems specifically tailored to an installation's data processing requirements and its machine configuration. Also included in the publication are summaries of machine and core-storage requirements for all Operating System components, and timing estimates for the IOCS.

Utility Programs (C28-0353) describes the Operating System utility programs and details the control cards and other requirements for their use. The publication also includes lists of all error and diagnostic messages that may be issued during execution of the programs.

Generalized Tape Sorting Program (C28-0354) provides the detailed information necessary to define sort and merge programs and to use these programs. The publication describes (1) the control cards directed to the Sort Definition program to create a sort or merge program to fit the particular application and (2) the control cards to alter the created program. Also included are a complete list of diagnostic and error messages, timing estimates for the execution of sort and merge programs for data records of varying lengths, and information on how to incorporate user-written editing routines into the generated programs.

Generalized Sorting Program Using IBM 1301/2302 Disk Storage (C28-0404) provides the information

needed by the user to create sort programs and to use them. Described in detail are the control cards needed to create the programs and, at execution time, to alter the programs to fit the specific application. The publication also contains a complete list of diagnostic and error messages, timing estimates for the sorting of data records of varying lengths, and information on how to incorporate user-written editing routines into the generated programs.

Programming Systems Analysis Guides, addressed to technical personnel that analyze or modify Operating System programs, are provided for some Operating System components. These analysis guides describe the internal operations of programs and programming systems at several levels of detail: the program level, the major component (or phase) level, and the routine level. In addition, details of operation at the routine/subroutine level — with accompanying flowcharts — are provided. The programs documented by Programming Systems Analysis Guides are apparent from the titles of those publications. The last line of each title is *Programming Systems Analysis Guide.* (This line is not repeated below.) The remainders of the titles and their associated form numbers are:

Resident and Transitional Monitors, and Input/Output Control System, Form C28-0396

Autocoder, Form C28-0395

COBOL, Form C28-0397

FORTRAN, Form C28-0398

Generalized Tape Sorting Program, Form C28-0393

NOTE: Information supplementary to *System Monitor, Basic Input/Output Control System,* and *Utility Programs* concerning IBM 1311 Disk Storage Drives can be found in the publication *IBM 1410/7010 Operating System; Support of IBM 1311 Disk Storage Drives Under the Operating System,* Form C28-0402. Information concerning IBM 1301/2302 Disk Storage can be found in the publication *IBM 1401/7010 Operating System; File Organization System for IBM 1301/2302 Disk Storage,* Form C28-0405.

Glossary

ABSOLUTE PROGRAM: A machine-language program that can be loaded directly into a specific area of core storage and executed from it. In the Operating System, the Linkage Loader converts programs from relocatable format (see below) to absolute format.

BATCH PROCESSING: A type of processing in which a number of input items are grouped for processing under some control system. (The term applies to both data and programs.) The Operating System is basically a batch-processing system; control resides in the System Monitor, which can supervise the execution of a wide range of jobs and runs within jobs, with a minimum of operator intervention. However, the System Monitor can, for example, give control to the Tele-processing Supervisor, an auxiliary monitor, that can operate on data and call in programs in a nonbatch, or random, manner.

BOOTSTRAP ROUTINE: In the Operating System, a routine that loads the Resident Monitor into core storage. In a tape-oriented system, the Bootstrap routine is the first program on the Master file; in a disk-oriented system, the Bootstrap routine is preceded by the Tape-to-Disk Load program.

CORE IMAGE FILE (MDM): A system file on which the Resident Monitor of the Operating System records: (1) the status of core storage at periodic intervals in the execution of a program (checkpoint records), and (2) the status of core storage at the time the Unusual-End-of-Program routine is entered (if a branch is made to that routine). Checkpoint records can be used to restart a program from the point in the program at which the core-storage status was recorded. Records written at Unusual End of Program can be used as input to the Storage Print program.

CREATE LIBRARY: Prewritten packets of Linkage Loader control cards that can be used to call groups of relocatable program units from the Relocatable Library to create absolute programs. Create packets can be used extensively at System Generation. Use of these packets substantially reduces the number of control cards that must be keypunched and verified.

DEPENDENT PROGRAM: A program executed under control of the System Monitor of the Operating System, and utilizing the Operating System Input/Output Control System.

DISK-ORIENTED SYSTEM: A form of the Operating System in which the System Operating File (SOF) resides on IBM 1301 or IBM 2302 Disk Storage (see "Tape-Oriented System"). The user receives the Master file for a disk-oriented system on a reel of magnetic tape.

JOB FILE (MJB): The output file of the Linkage Loader. The Job file contains programs, in the form of card-image records, in absolute format.

LINKAGE LOADER: A program of the System Monitor that converts other programs from relocatable to absolute format. It modifies addresses as required and converts linkage symbols to the appropriate machine addresses. The programs are thus prepared for execution. The Linkage Loader places its output on the Job file (MJB).

LINKAGE SYMBOL: A symbolic address providing communication among two or more programs, each compiled independently of the other. In the Operating System, a linkage symbol may take one of two forms (see the publication *System Monitor*). The language processors do not convert linkage symbols to machine addresses; when the Linkage Loader encounters a card defining a linkage symbol, it places the symbol, and the address of the symbol within the relocated program, in a symbol table. If, during the same run, a program subsequently processed by the Linkage Loader refers to the linkage symbol, the machine address is available from the symbol table.

MASTER FILE: The IBM-supplied tape file containing all components of the Operating System. The Master file is the source file for each user's initial System Generation run.

MODULE: A basic program or subprogram within the Operating System. It can be a complete program within itself or a program segment (subroutine) that is to be combined with other subroutines to form a complete program.

OBJECT PROGRAM: A program written in machine language that is the output from the processor, which has converted the machine language from the symbolic language.

PROCESSOR (also Language Processor): A program used to translate source statements written in a symbolic language into machine language. The processor and associated symbolic language are referred to collectively as a programming system. The three language processors included in the Operating System are Autocoder, COBOL, and FORTRAN.

RANDOM PROCESSING: The process by which data is accessed according to a sequence established by the program using the data, rather than by a sequential arrangement of the data.

RELOCATABLE FORMAT: A form of machine-language program in which addresses have been assigned starting at zero or some other arbitrary origin point. The term implies that the Linkage Loader in the Operating System will convert the addresses to those that will be used at program execution time. The principle of relocatability provides the Operating System with great flexibility. Despite the size of the Resident Monitor and IOCS (each user can tailor the Resident Monitor and IOCS to fit his individual requirements), components of the Operating System can be loaded into core storage with no wasted gap of core-storage space. In addition, the relocatable nature of the programs, or modules, in the Relocatable Library allows for various combinations and configurations of these modules, depending on the user's requirements.

RESIDENT MONITOR: A program of the System Monitor that remains in core storage at all times and controls the execution of all other programs under the Operating System.

SOURCE PROGRAM: A program written in a symbolic language (Autocoder, FORTRAN, or COBOL) that is the input to the processor, which converts the language into machine language in relocatable format.

STANDARD INPUT UNIT (SIU): A card reader or magnetic tape unit from which control information for the System Monitor and other components of the Operating System can be read. The SIU can also serve as the input device for source programs and data.

STANDARD PRINT UNIT (SPR): A printer or magnetic tape unit on which output from programs executed under the Operating System can be written.

STANDARD PUNCH UNIT (SPU): A card punch or magnetic tape unit on which output from programs executed under the Operating System can be placed.

SYSTEM GENERATION: The process by which a specific Operating System is created from the IBM Master file to satisfy a user's processing requirements. The end product of System Generation is the System Operating File(s). If desired, a System Generator File (SGF) can be produced to generate the SOF(s).

SYSTEM GENERATOR FILE (SGF): A file, created from the IBM Master file, that contains all components of the Operating System needed to satisfy the processing requirements of an installation. The SGF can thus become the Master file for an installation.

SYSTEM MONITOR: The supervisory program in the Operating System that coordinates the functions of all Operating System components, including user-written programs operating under it. The three components that comprise the System Monitor are the Resident Monitor, Transitional Monitor, and Linkage Loader.

SYSTEM OPERATING FILE (SOF): A tape or disk file created from the SGF during System Generation. The SOF contains absolute programs and relocatable programs in libraries needed to carry out the actual processing within an installation.

SYSTEM SYMBOL: A symbolic address of a routine (or data field) within the Resident Monitor. System symbols can also be used for pointing to a location in core storage; however, the symbols must be designated at System Generation. Any dependent program can refer to these symbols, which are included in the Linkage Loader symbol table. When the Linkage Loader relocates the dependent programs, it converts to actual addresses all system symbols used within the dependent programs.

TAPE-ORIENTED SYSTEM: A form of the Operating System in which the System Operating File (SOF) resides on magnetic tape.

TRANSITIONAL MONITOR: A program of the System Monitor that performs transition functions between jobs and between runs within a job. The Transitional Monitor analyzes Monitor control cards and is controlled by the Resident Monitor.

Printed in U.S.A. C28-0318-3

International Business Machines Corporation
Data Processing Division
112 East Post Road, White Plains, N.Y. 10601