SPERRY UNIVAC 90/30 Data Processing System

System Description

H

SPERRY UNIVAC COMPUTER SYSTEMS

UP-8060 Rev. 2

Contents

CONTENTS

TABLES

1. Introduction

1.1. SPERRY UNIVAC 90/30 DATA PROCESSING SYSTEM

The SPERRY UNIVAC 90/30 Data Processing System (Figures 1—1 and 1—2) features high-quality compact integrated hardware, the 90/30 system, complemented by an advanced, yet easy-to-use modular software package, the SPERRY UNIVAC Operating System/3 (OS/3). Together they extend the computer resource into a maximum usage environment.

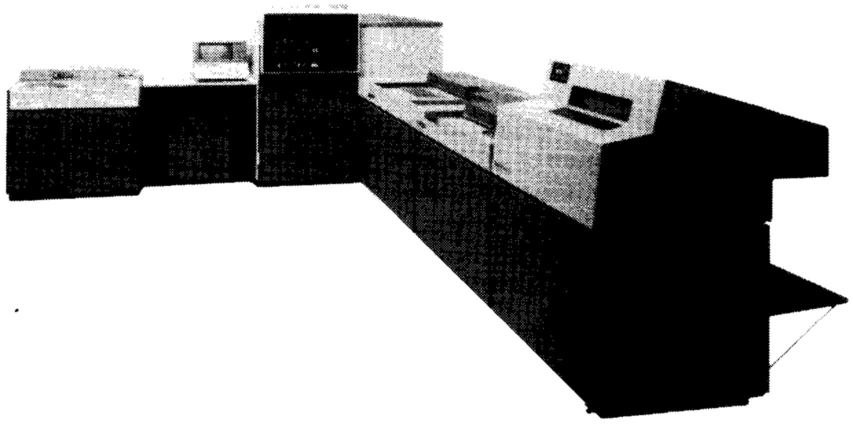


Figure 1—1. 90/30 System

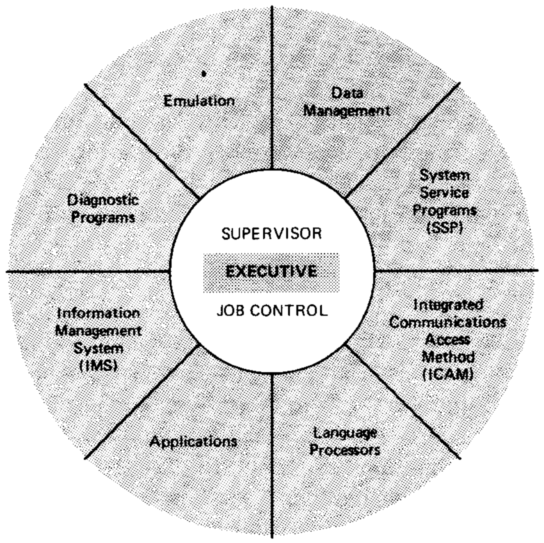

Emulation
Data Management
System Service Programs (SSP)
Integrated Communications Access Method (ICAM)
Language Processors
Applications
Information Management System (IMS)
Diagnostic Programs
SUPERVISOR
EXECUTIVE
JOB CONTROL

Figure 1—2. Operating System/3

Today, computer performance is being challenged by a variety of applications which have increased in scope and complexity. As a result, a significant number of established data processing systems cannot meet the requirements of these new applications because the systems lack the inherent processing capability. Establishments that are considering the possibility of computerizing functions which have grown too unwieldly have a difficult time choosing an initial data processing system. If a small company with a high growth rate chooses a small system, it may rapidly outstrip the capabilities of the system. On the other hand, if the company selects a medium-to-large scale system, it may be paying for unused power. Sperry Univac has developed the 90/30 Data Processing System for both the uninitiated prospective user and the established user who want to upgrade their data processing capabilities. It is designed for businesses that require a computer resource offering high performance at low cost and extended capabilities that encompass an unmatched range of computational requirements.

Some of the advanced capabilities of the 90/30 Data Processing System that increase its productivity include:

- Job control Language (JCL)

  JCL is a set of descriptive control statements that automate computer operations. It allows the programmer to define the characteristics and requirements of each job, thereby directing OS/3 to automatically perform resource assignment and job execution. By reducing or eliminating the amount of operator intervention required to process each job, JCL improves the productivity of both the 90/30 Data Processing System and the computer operators.

- Multijobbing

  Multijobbing increases throughput. Throughput refers to the actual productivity of a data processing system and is measured by the amount of useful computing work done per minute or hour. Multijobbing increases system productivity by interleaving the execution of job steps from more than one job. When any job step is waiting for an external event (I/O request) to occur before processing can continue, another job is given control. Priorities and available resources determine which job step is given control by the data processing system at any point in time.

- Large Capacity DASD Files

  One of the highlights of the 90/30 system is the SPERRY UNIVAC 8416/8418 Integrated Disc Subsystem. Each removable pack provides 28.9 or 57.9 million bytes of usable storage for direct access storage device (DASD) files, depending on the model used. This offers the following advantages:

  — Supports disc-oriented processing

  — Provides for economical conversion of large tape or card files to direct access files

  — Offers additional storage which permits such OS/3 enhancements as spooling, cataloging of job streams, inquiry/response operation, and message processing

  — Permits expansion to a data base which puts more information online

- Communications

  Data communications utilizes commercially available communications facilities to link a central processing site with remote sites to accomplish a variety of data processing applications. In meeting today's rising demand for this type of service, the 90/30 Data Processing System includes a sophisticated but easy to implement communications package — the integrated communications access method (ICAM). All users, whether they are investigating communications for the first time or are upgrading their present capabilities, will find ICAM an indispensable tool. It is an integrated system capable of supporting several levels of communications processing. ICAM can be tailored to fit the user's needs, the type of service desired, and the installation configuration.

- SPERRY UNIVAC Series 90 Information Management System (IMS 90)

  IMS 90 adds a new dimension to computer applications through an inquiry/response capability in a real-time environment. IMS 90 has been designed to provide an easy-to-use communications/data management package.

  User support for IMS 90 is in the form of Sperry Univac-supplied applications programs called UNIQUE, which require no programming effort on the part of the user. Also, coding required for line and device handlers is provided; hence, the user does not require in-house communication expertise to go online.

- SPERRY UNIVAC Series 90 Data Base Management System (DMS 90)

  DMS 90 is a collection of system programs that support the development of integrated data bases. These programs provide for the description, initialization, creation, accessing, maintenance, backup, and recovery of data bases. The languages used in the description and manipulation of DMS 90 data bases are derived from the CODASYL data base specifications. A data base may be accessed by batch application programs and communications application programs.

- Conversion

  In the purchase of any new system, the conversion of existing programs and data files is always uppermost in the customer's mind. Sperry Univac provides utility routines to convert data files generated for the SPERRY UNIVAC 9200/9300 System and the IBM 360/20 System into data files that are suitable for use on the 90/30 Data Processing System. Also, information is provided on how to convert your SPERRY UNIVAC 9200/9300 and IBM 360/20 programs.

- Emulation

  Two emulators are provided: a SPERRY UNIVAC 9200/9300 Emulator and an IBM 360/20 Emulator. These emulators enable the user to run his SPERRY UNIVAC 9200/9300 and IBM 360/20 programs while he is converting his programs and data files.

For more details about the extended capabilities of the 90/30 Data Processing System, see Section 2. Section 3 provides a complete technical description of the entire software package.

2. Extended Capabilities of the 90/30 System

2.1. JOB CONTROL LANGUAGE (JCL)

Job control manages the system resources and prepares jobs which are submitted for execution. A job represents a unit of work to be performed by OS/3. Each job consists of one or more job steps, each requesting the execution of a system or user program. Job control services are performed prior to the execution of the initial job step, during the transition between job steps, and at the conclusion of the job.

The services of job control are directed by the user through statements provided by the job control language (JCL). These control statements define the system resources required for proper execution of a job and facilitate the efficient management of these resources. OS/3 JCL is a flexible language that enables the user to specify the requirements for a variety of essential resources and affords a high degree of independence from limitations imposed by system configurations. Through the use of cataloged procedures, OS/3 effectively reduces the usual effort required when running frequently executed jobs.

Services performed by the OS/3 job control include:

- Automatic job scheduling and initiation
- Automatic main storage allocation
- Device assignment
- Volume and file label processing
- Retrieval of cataloged control streams for subsequent modification and execution
- File catalog maintenance
- Program restart from a checkpoint
- Object code debugging
- Scheduling of additional jobs from within a job
- Control streams and data storage

A control stream is a group of sequenced statements, written in the OS/3 job control language, which defines a job and directs its execution. These statements are subdivided into the following functional groups:

- Job Coordination
  - Job identification and delimitation
  - Priority scheduling

- Main storage requirements
- Scheduling a job from within a job
- Restart of an interrupted job
- Modification of cataloged control streams
- Data management common code modules

■ Device Assignment

- Device assignment sets
- Change device code
- Release peripheral devices
- Delete files

■ Job Step Operation

- Program execution
- Altering prestored programs
- Data and parameter specification

■ Regulating Job Environment

- Job step options
- Skip control statements
- Trace mode
- Job-to-operator communications
- Date change
- Magnetic tape positioning

2.1.1. Job Coordination

This group of control statements provides an interface that coordinates overall job execution. These statements can specify information regarding:

■ Job Identification and Delimitation

Uniquely identifies the job and indicates its starting and ending points.

■ Scheduling Priority

Indicates one of the following priorities to be used when scheduling a job:

- Normal priority

  Used for regular scheduling considerations within the system

- High priority

  Used for rush scheduling

- Preemptive priority

  Used for urgent jobs that require immediate scheduling and execution

Within priorities, all jobs are scheduled for execution on a first-in, first-fit basis with main storage allocated for preemptive jobs via the rollout/rollin capability.

- Main Storage Requirements

  The main storage requirement for a job can be calculated automatically by OS/3 if all programs to be executed by the job either currently reside in a load library or are specified by the user through the job control language. The user has the option of specifying a minimum and maximum value.

  The minimum value is the basic storage requirement to properly execute all programs within the job. The maximum value specifies an additional storage requirement that, if available, could be dynamically utilized by the programs within the job to improve and speed up job execution. To exploit this additional main storage, the programs within the job must be specifically designed to take advantage of the additional main storage allocation if it becomes available to the job.

- Scheduling a Job From Within a Job

  OS/3 JCL enables the user to serially execute jobs by allowing a currently active job to request another job, which resides in the permanent library, to be executed. The requested job is processed and entered on the job queue as soon as the control stream processor is loaded. Also, a new priority level may be specified for the called job and it does not have to be the priority specified in the original job.

- Restart of an Interrupted Job

  Job restart from a specified program checkpoint is specified through JCL by giving the required checkpoint information and resource requirements for the recurring portions of the job. Checkpoint data must be established by the user prior to requesting job control to restart a job.

- Modification of Cataloged Control Streams

  Permanently cataloged control streams may be dynamically modified at execution time. Job control incorporates the submitted changes, as required, into the control stream retrieved from the permanent control stream library and subsequently processes the modified control stream for execution, leaving the content of the original control stream intact.

- Data Management Common Code Modules

  This feature defines, for the system, the specific data management common code modules (also referred to as shared modules) required for job execution. This facility allows job control to ensure the availability of required common code modules when the user has specified a load library other than system residence.

2.1.2. Device Assignment

This group of control statements is used to declare the devices required for the proper execution of a job. These statements can specify the following:

- Device Assignment Sets

  OS/3 JCL offers a set of control statements which provide the information required to assign devices and establish the relationship between files or volumes and devices such as the number of files per volume and the number of volumes that contain a file. The peripheral devices that may be assigned to satisfy the requirements of a job include card readers, printers, punches, discs, and tapes.

- Change Device Code

  This function provides the ability to temporarily change the logical unit number associated with a given device type from that specified at system generation time to one that will be utilized for a specific job, thus permitting the user to execute existing control streams between different 90/30 Data Processing System installations.

- Release Peripheral Devices

  This function is used to release peripheral devices that are presently assigned to a job and that are not required in subsequent job steps. Upon release, they are no longer available to the original job, and they are made available for subsequent assignment to other jobs being considered for initiation.

- Delete Files

  This function deletes specified files from the system prior to the execution of a job step.

2.1.3. Job Step Operation

This group of control statements is used to specify job step operation. A job always consists of one or more job steps. One job step could assemble a source program; another job step would link the object program, followed by a job step to execute it. These statements can specify the following:

- Program Execution

  OS/3 JCL provides the facility to specify that a user or system program be loaded for execution from a system library or an alternate user library. A job step priority can also be indicated, with the lowest value indicating highest priority.

- Altering Prestored Programs

  This feature of job control allows programs to be altered at execution time. The control statement gives the name of the module to be altered. After being loaded into main storage, the module is altered according to the specified changes. However, the master copy of the named module, contained in the appropriate load library, remains intact.

- Data and Parameter Specification

  OS/3 JCL allows the user to include information in the control stream as embedded data or program parameters. This information is stored in the temporary control stream library for subsequent retrieval by a user or system program.

2.1.4. Regulating Job Environment

This group of control statements is used to regulate the environment of the job by making designated job steps conditional on the outcome of a previous job step or on any errors that might previously have occurred during execution, or by modifying the environment of the job. These statements can specify the following:

- Job Step Options

  OS/3 JCL allows the user to specify certain optional software functions to be performed in a job step. The specified functions are effective only in the job step in which they are included. The user can request any of the following options:

  — Alter

    Indicates that a loaded program is to be altered prior to being given control for execution

  — Binary Overflow

    Indicates that the user program to be loaded is given control for execution with binary overflow enabled

  — Decimal Overflow

    Indicates that the user program to be loaded is to be given control for execution with decimal overflow enabled

  — Load Module Construction

    Indicates that the source program being compiled is to be linked and executed using default linker parameters

  — No Abnormal Dump

    Indicates that a dump is not desired in the event of abnormal job step termination

  — System Dump

    Indicates that a system dump is desired in place of a job dump in the event of abnormal job step termination

  — No Volume Label Check

    Indicates that VOL 1 header labels are not to be read and verified on disc or tape volumes

- Skip Control Statements

  This feature allows the user to bypass any number of control statements, including all statements in an executing control stream. The user can indicate a forward skip from any control statement, to any control statement in the stream. If no destination is specified, job control advances to the first statement in the next job step.

- Trace Mode

  OS/3 JCL provides a trace option for use in program debugging. When the trace mode is invoked, every instruction in the job step is examined before execution without interfering with the normal operation of the program and the pertinent operational information for every traced instruction is printed by the system.

- Job-to-Operator Communications

  This feature enables the user to communicate with the operator. The user can place a message of up to 60 characters anywhere in the control stream and the message will be displayed on the screen of the system console during control stream processing.

- Date Change

  This allows the user to alter or modify the calendar date for a specific job by submitting a six-character date consisting of the month, day, and year in any order, or a five-character date consisting of two characters for the year and three characters for the day.

- Magnetic Tape Positioning

  This feature allows the positioning of tape volumes prior to the execution of a job step. It can be used to position a data file or pre-position a multifile tape volume. It will space the tape volume forward or backward a specified number of tape marks or blocks, rewind the tape volume, or write a tape mark.

2.2. MULTIJOBBING

The 90/30 Data Processing System can concurrently process from one to seven jobs, with each job consisting of one or more job steps (programs) which are executed serially. A job step may also have one or more tasks which may be executed concurrently. This capability allows the user greater flexibility in attaining maximum use of the system's resources and in scheduling tasks.

OS/3 multijobbing consists of scheduling multiple jobs (up to 7) for concurrent execution. The allocation of processor time is based on a system switch list which contains information regarding program priorities, task synchronization, and input/output utilization. While one task is awaiting the completion of an external event (such as completion of an input/output request), OS/3 activates another task that is ready to run to ensure optimum utilization of the processor's capabilities. Since the majority of programs require support other than processing instructions, OS/3 multijobbing provides an effective method for the user to reduce processor idle time and increase system productivity (throughput).

OS/3 was designed around the multijobbing concept. All the software that Sperry Univac provides in the OS/3 package is designed to take full advantage of this multijobbing environment. The controlling software automatically concurrently processes any user jobs submitted to the system.

In addition to multijobbing, Sperry Univac also offers a programming capability to the user whereby he can interface directly with the control software (OS/3 executive). The additional capabilities provided by the executive and available to the user are:

- Multitasking
- Physical Input/Output Control (PIOCS)
- Main Storage Allocation

2.2.1. Multitasking

OS/3 has a multitasking capability which distributes processing time to tasks within job steps based on the switch list priorities and input/output utilization.

The user/system programs submitted to OS/3 are contained in the job steps of a job. Every job step is considered to be one program and a job is automatically processed by OS/3 from job step to job step.

Every job step submitted to OS/3 is established as a primary task. A task is the **lowest viable entity that** can compete for processor time. OS/3 permits up to 256 tasks per job step. The switch list **has the capacity to allow** the user to specify up to 60 levels of processing priority for tasks. A more practical limit of **3 to 15 is sufficient** to achieve a high degree of processor utilization. When a task is interrupted to perform **external processing** (external to the instruction processor), it frees the processor and OS/3 searches the switch list **for the highest** priority task that is not waiting for an external event to be completed. This task could be in the same job or it could be from any other job currently being processed.

OS/3 has another level of multitasking which may occur within a job step. The primary task is capable of initiating other tasks, called subtasks, within the job step. Primary tasks and subtasks are simply two categories of tasks; each is processed in the same manner. However, the primary task is automatically initiated into the multitasking environment by OS/3 at job step initiation — while subtasks must be created by the program in the job step. Subtasks can be given the same priority as the primary task or they can have a lower priority. Thus, a program in a job step may consist of a primary task and several possible subtasks, all of which compete independently for processor time.

OS/3 multitasking is a substructure of multijobbing and is a valuable asset to user program processing. OS/3 multitasking not only involves distributing processing time to tasks, which is also what multiprogramming does, but extends the capabilities of multiprogramming by allowing a single copy of a program module to be used by more than one task. (This is, in fact, the basic difference between multitasking and multiprogramming.) There are two types of user program modules that are capable of being shared: reentrant program modules and serially reusable program modules. A reentrant program module can be used by several tasks in an interleaved fashion — while a serially reusable program can be used only by one task at a time.

2.2.2. Physical Input/Output Control System

OS/3 performs all input/output (I/O) operations with peripheral devices through the physical input/output control system (PIOCS). PIOCS handles the queueing and initiation of all I/O commands and the processing of I/O interrupts. OS/3 PIOCS is composed of general purpose software routines designed to provide maximum throughput on all peripheral devices and to allow for ease of expansion to support new devices.

PIOCS receives control whenever data management requests that an I/O operation be performed. Control is not returned to data management until the I/O request has been completed. However, other tasks in the system may be activated if their status indicates a ready-to-run condition. Requests for I/O operations are initially queued, by priority, in device and channel queues. Dispatching follows the queueing of an I/O order if the channel and device are free or, if not, upon completion of a previous order on the same channel. The I/O dispatch routines perform service functions, as required, such as disc address verification and parameter checking. Interrupts from I/O channels are serviced as a high priority function of the supervisor in order to free the channel for dispatching other I/O orders which may have been queued. Upon completion of an I/O order, a general I/O status analysis is performed to determine whether any abnormal conditions have occurred. For normal I/O terminations, control is returned to the I/O dispatcher. If no additional requests remain in the queue, control is passed to the task switcher for return of control to another task.

If an error condition occurs, the appropriate device is flagged as unavailable for all tasks. In the case of an error on the system resident device, the resident error recovery routine receives control.

Other error conditions require more detailed analysis. A device error recovery overlay routine is called in to complete processing of the error condition.

OS/3 also provides the user with the option to perform his own error processing. If this option is employed, system error analysis and recovery is bypassed and the hardware status information is returned to the task that issued the I/O request. The responsibility for processing the error remains with this task. OS/3 also supports the capability for the user to interface directly to PIOCS, if desired, to provide special handler interfaces for nonsupported devices, or to use supported devices at the physical level.

2.2.3. Main Storage Allocation

A function of OS/3 multijobbing is to determine if a job's resource requirements can be satisfied. One of these resources is main storage. OS/3 assigns a contiguous region of main storage, large enough to satisfy the requesting job, and loads the job to be executed into that region. OS/3 tabulates all assigned and unassigned main storage regions and dynamically reassigns unoccupied regions to other requesting jobs.

2.3. SPOOLING SERVICES

When card readers, printers, or card punches (paper peripherals) are required by a job, they generally must be dedicated to the job at the point where they are requested. The serial nature of these devices does not lend itself to multijobbing. Since the speed of these devices is far less than the speed of the other resources in the system, they drastically impede thoroughput if allowed to tie up the system. To eliminate this problem, OS/3 provides automatic spooling (or, *s*imultaneous *p*eripheral *o*perations *o*n *l*ine) which buffers the input/output information to and from disc. This allows the card and print services to operate at their rated speeds while the user jobs are processed without regard to device contention. OS/3 spooling is made up of the following types of routines:

- Input Readers

  Read cards from card readers using PIOCS and store these card images, via the spooler, in the spool file for subsequent use by the intended program.

- Spooler Cooperative

  Handles all spool file input and output operations and accesses the disc when necessary, using the system access technique (SAT) for accesses to the spool file. It provides record level input and output to and from the spool file for each element in the system needing access to that file. It can handle any number of read, punch, and print files simultaneously, including multiple files per job.

- Output Writers

  Read data from the system spool file and print or punch this data on the physical devices. Output writers can also output this data to a tape or disc file for processing at a later time by the user.

In OS/3, spooling requires no changes to any user programs.

2.4. LARGE CAPACITY DASD FILES

Direct access storage devices (DASD) are auxiliary storage devices that can store large quantities of data at a relatively economical cost. Although DASD does not directly interface with the processor, as does main storage, it is still promptly accessible to the system. DASD may be used for sequential files, indexed sequential files, and direct access files.

OS/3 is a disc-oriented operating system. Most of the system software resides on disc (referred to as the system resident device or SYSRES) while the software needed for system operation is resident in low-order main storage. The minimum configuration for the 90/30 Data Processing System includes a SPERRY UNIVAC 8416 or 8418 Disc Subsystem with two disc drives. One of these drives is used primarily as the system resident device and to store user programs. The other drive is used to store user data. This disc subsystem is integrated, which means it has its own input/output (I/O) interface (integrated disc adapter) and does not need a separate selector (I/O) channel.

A user can increase the auxiliary storage capacity of the minimum system configuration by adding up to six more 8416 or 8418 disc drive units (for a total of eight) or by adding or substituting any of the following SPERRY UNIVAC disc subsystems on an optional selector channel:

- 8430 Disc Subsystem
- 8414 Disc Subsystem*
- 8411 Disc Subsystem*

The disc capacity of the 90/30 Data Processing System provides the user with a repository for large capacity DASD files. The potential for practical application is boundless.

2.4.1. Corporate Data Base

The most important application of DASD is as the primary media for establishing a *corporate* data base. This is the data base that contains all relevant information that exists in a company at any time. The corporate data base usually consists of one or more *application* data bases which contain all the information about one process within the company's operation.

- Application Data Base

  This is a type of master file which is permanent and can only be used for one type of application. If this file is on disc, transactions may be processed in any order with no need for sorting or classifying the data. Also, there is no need for batching as transactions can be processed as they occur.

- Real-Time Data Base

  A data base can be designed so that it can be accessed by users through entry and inquiry terminals with practically no delay during transactions. A user who is interested in this type of file processing system should consider IMS 90 which is available with OS/3.

- Statistics Tabulation

  DASD is the most practical medium that can be used in the tabulation of business statistics, which is a common data processing application. DASD provides the large amount of storage needed to simultaneously accumulate the statistics for the many different statistical tables; DASD also eliminates the necessity to sort input records. It is possible to prepare large statistics tables all in one pass and make them available on a timely basis.

- Program Segmentation

  A program that is so large that it cannot be completely stored in main storage can be stored on disc and executed in parts. The user program is divided into independent segments. Each segment is loaded into main storage when required, overlaying previously executed segments. Program segmentation reduces the use of main storage and contributes to the efficient use of this important resource.

* *Subject to availability*

2.5. COMMUNICATIONS

The users who will be considering a communications system are small-to-medium size companies having one or more plants, warehouse locations, or sales offices that are considerably distant from the home office. Data communications can increase the efficiency in information flow by providing timely party-to-party data transfer. The 90/30 Data Processing System offers a multilevel communications system which consists of a modular software package, the OS/3 integrated communications access method (ICAM), and a complement of SPERRY UNIVAC Data Communications Terminals which interface through the SPERRY UNIVAC 90/30 Communications Adapter (CA). These terminals include:

- UNISCOPE 100/200 Display Terminals
- DCT 500/524 Data Communications Terminals
- DCT 1000 Data Communications Terminals
- DCT 2000 Data Communications Terminals
- 1004/1005 Card Processor Systems
- 9200/9300 Systems

OS/3 communications supports two basic types of communication: narrow band and wide band transmissions. Narrow band transmission permits voice grade communications which is referred to as dial-up, switched DDD (direct distance dialing), or privately leased lines. Wide band transmission permits data transmission over high speed, privately leased lines. Sperry Univac provides full-duplex and half-duplex interfaces for its own terminals as well as commercially available data sets.

OS/3 ICAM provides three levels of supports:

- Message Control Program (MCP)

  MCP is the highest level of support offered by OS/3. It provides a complete communications support package controlled by macro instructions issued by the user program (GET, PUT, etc.).

- Remote Device Handler (RDH) Interface

  This level of support is intended to provide a communications capability in the smallest systems while maintaining a degree of device independence for user programs. At this interface level, the user program communicates directly with the remote device handlers.

- Channel Control Routine (CCR) Interface

  This level provides an interface between the user and ICAM at the physical input/output level. This ICAM configuration requires less space in main storage because the greater part of the programming effort to provide communications support is shifted to the user. The primary purpose of this level is to permit users to write their own specialized MCP's without having to modify the OS/3 interfaces.

- Remote Batch Processor (RBP)

  The RBP capability permits jobs to be entered into the 90/30 Data Processing System from a variety of remote terminal devices.

2.5.1. Message Control Program (MCP)

The MCP is a modular software package that is capable of supporting either simple or complex communications environments. A single MCP can provide concurrent support for multiple user message processing programs that use a variety of terminals and line types. MCP is responsible for preventing conflicting resource assignments and for releasing facilities when jobs terminate. User programs are provided with a group of macro instructions which perform the following:

- control table generation;
- data transfers to and from user-specified buffer areas;
- communication facilities initialization and control; and
- dynamic terminal and poll table entry alterations in the communication control areas.

In addition to the macro instruction interface, MCP provides the following interfaces:

- RPG II Telecommunications

  OS/3 RPG II users are provided with a simplified remote I/O capability via an RPG II telecommunications interface.

- IMS 90 Communications

  IMS 90 is a file processing system operating in a communications environment that offers a simplified inquiry/update language for manipulating information in user data files. This support is in the form of action programs which require no programming effort by the user, or it may be in the form of a communications network and file processing system for user applications written in COBOL or BAL.

2.5.2. MCP Options

The following enhancements available with the OS/3 MCP support the following complex message processing functions:

- Message Queueing

  Message queueing is the stacking of complete messages in main storage or disc storage while they are waiting to be serviced by a communications line handler or a message processing program. A network may contain one or more message queues associated with lines, terminals, or processing queues.

- Multiple Destination Routing

  The disc queueing facility provides multiple routing of messages. Provision is made for allowing up to 255 destinations for a single message. Only a single byte is required in main storage to determine that a message has been delivered to all of its destinations.

- Activity Scheduling and Priority Control

  An activity scheduling routine is provided with an optional priority suspension and scheduling capability. In small system configurations, a single level of scheduling is provided; therefore, a suspension capability is not required. Multiple priority levels and a suspension capability are provided for larger systems with critical timing requirements.

- Timer Service

  A centralized timing service provides a timer for active data buffers with a gradient value of 1 second. In addition, this service provides a feature that permits the scheduling of an activity after a specified period of time has elapsed. Timer increments for this service are 1/10 second.

- Checkpoint/Restart

  Restart procedures can reconstruct message queues in their original status at the point of system hardware or software failure. When a checkpoint is initiated, the complete communications control area is written onto a tape or disc file. When checkpointing has been specified, it is performed automatically on an adjustable periodic basis.

- Journal Control

  Journal files are required when a user requires a recovery/restart capability, or when a statistical accounting of the network operation is required. Journal control provides for the recording of message processing events and message data on a disc or tape file.

- Statistics Accumulation

  Statistics are maintained for each communications line and terminal and they are available to a user program upon request. The accumulation contains totals for the following types of information:

  — Number of messages received

  — Number of input retransmission requests

  — Number of messages transmitted

  — Number of output transmission requests

  — Number of poll messages

  — Number of no-traffic responses

2.6. DATA BASE MANAGEMENT SYSTEMS

2.6.1. Series 90 Information Management System (IMS 90)

IMS 90 is a file processing system, operating in a communications environment, that offers a simplified inquiry/update language for manipulating information in user data files. The users who should consider IMS 90 are those who have an extensive library of applications programs or large data bases which will become an integral part of the daily application of the 90/30 Data Processing System. OS/3 also has a file processing system, functioning at the system level, which processes files for individual programs operating in the multijobbing environment. This is fine for tasks that are performed maybe once a day, and the user is not concerned with immediate response, but applications that need an immediate response cannot be processed in this manner. IMS 90 allows the user to consolidate application programs, written in COBOL or BAL, and the data files into one common data base. IMS 90 has a simplified inquiry/update language which interfaces the users with their data base from a local or remote visual display, or from hard copy TELETYPE* terminals. By using IMS 90, the user can configure the 90/30 Data Processing System so that nonprogramming personnel can access the installation's data base. Information retrieval and updating can be performed by personnel with no formal data processing training. All these processes can take place in a multijobbing environment since IMS 90 is executed as just another user program under control of OS/3. When designing a data base for IMS 90, the user can optionally specify one or all of the following features:

**Trademark of Teletype Corporation*

- Password Capacity

  A password capability is provided to facilitate security measures in the online environment of IMS 90. IMS 90 is capable of not only limiting system access to authorized personnel, but also limiting those persons to certain elements within those files.

- Reorganize Data Base Files

  Data can be selected from records in several files and combined to construct a new file. IMS 90 requires that the basic files be defined, and how the selected data is to be combined for the new file.

- Master and Remote Terminal Commands

  IMS 90 provides commands for a master terminal to assist in monitoring the system and commands for remote terminals which can be used for educational purposes or to resolve various administrative or operational problems.

- Automatic Rollback

  This procedure can be initiated when a transaction is abnormally terminated or cancelled by the terminal operator. Any file modified prior to termination is returned to its logical state as it existed before the transaction was initiated.

- System Statistics Report

  IMS 90 maintains a wide range of system statistics that are available to the user in report form. These statistics are designed primarily to assist users in evaluating their system and possible changing system configuration parameters to achieve greater efficiency.

In summary, the features and capabilities of IMS 90 provide a vehicle for users to design their own data bases and file processing systems for concurrent use with other applications in the multijobbing environment of the 90/30 Data Processing System.

2.6.2. Series 90 Data Base Management System (DMS 90)

DMS 90 is a collection of system programs that support the development of integrated data bases. These programs provide for the description, initialization, creation, accessing, maintenance, backup, and recovery of data bases. The languages used in the description and manipulation of DMS 90 data bases are derived from the CODASYL data base specifications. A data base may be accessed by batch application programs and communications application programs.

- Description of Data

  The description of data in a DMS 90 data base is entirely separate from the manipulation of that data in application programs. This results in a higher level of data independence for application programs. All description is done in high-level languages comparable to the data declaration language of COBOL.

- Storage Structures and Access Methods

  Records can be related in sets in which a record of one type is designated as an owner and records of one or more other types are designated as members. Sets are implemented as record occurences linked into chains by data base keys.

Records can be retrieved from the data base by one or more of the following access methods: physically sequential retrieval within an area, direct retrieval by data base key, calculated retrieval by key value, chained retrieval by set relationships.

- Manipulation of Data

  The data base is accessed by means of data manipulation language (DML) statements that are written in the procedure division of a COBOL application program intermixed with conventional COBOL statements.

- Recovery

  Offline recovery utilities make use of a journal file generated by the data base management system for forward and backward data base recovery.

2.7. COMPATIBILITY

Most computer users, when considering the change to a new and more powerful system, give careful consideration to the problem of compatibility. Although the existing system can no longer carry the requisite work load, the user has made a substantial investment in its creation. A user may recognize the necessity for conversion but also realizes that some measure of the existing operation must be maintained during the conversion process. Compatibility, therefore, represents two quite distinct areas of concern: conversion and emulation.

2.7.1. Conversion

In order to convert the existing system, data files must be transferred to the new system's peripherals, and those program elements which are not accepted by the new system must be changed. If the user is satisfied with the structure of the existing data base, file conversion becomes a relatively simple question of dumping existing files to a transportable medium and restoring them on the new system. If the instruction repertoire of the new system is substantially identical to the existing system, the conversion task is reduced to modification of the programs' macros, file definitions, and incompatible instructions.

2.7.1.1. OS/3 Conversion Assistance

OS/3 provides a conversion utility package which assists present SPERRY UNIVAC 9200/9300 System and IBM 360/20 System users in upgrading to the 90/30 Data Processing System. The conversion utility package consists of the following elements:

- UNLOAD

  Dumps a user's current data base to either tape or card.

- RELOAD

  Rebuilds data files on 90/30 discs (either in their present structure or in a revised structure which is designed to take the fullest advantage of the 90/30 disc design).

- SCAN

  Points out the disparities (through a source code analysis routine) between the BAL instructions written for the existing system and the instructions accepted by the 90/30 Data Processing System. Other OS/3 language processors are highly compatible, yet they may require the user to perform some minor modification to existing programs.

2.7.2. Emulation

During the conversion process the user must maintain production without incurring the cost of maintaining two systems. In some cases, the user may wish to immediately convert all of the existing programs to be executed on the 90/30 Data Processing System. In other cases, because of extensive requirements, only some of the current programs are destined for conversion. In either case, however, provision must be made for executing existing programs, and, where the time consumed by conversion demands, updating of these existing programs.

2.7.2.1. OS/3 Emulation Assistance

OS/3 emulation supports virtually all user programs and system software written for the SPERRY UNIVAC 9200/9300 Systems and the IBM 360/20 System. Only communication programs and programs which operate with nonstandard software are excluded.

A very important feature of the OS/3 emulators is their capability to execute concurrently with other OS/3 user programs. In other words, OS/3 jobs can be intermixed with emulated programs. This means that the user can concurrently execute SPERRY UNIVAC 9200/9300 or IBM 360/20 programs under emulation while also executing OS/3 programs in a multijobbing environment.

The emulated SPERRY UNIVAC 9200/9300 and IBM 360/20 programs operate under control of their respective supervisors, which, in turn, interface with the OS/3 emulators. The OS/3 emulators, of course, operate under control of the OS/3 executive. The emulators process the job control statements from the SPERRY UNIVAC 9200/9300 and IBM 360/20 programs and permit them to operate until they attempt to execute an I/O transfer or other privileged instruction. At this juncture, 90/30 microcode traps the exception and returns it to the emulator for interpretive execution. Once that operation has been completed, control is returned to the emulated program. OS/3 also provides emulation aids which reduce the problems of executing under emulation to an absolute minimum. These aids will:

- Generate an emulator program

  Using simple parameters as input, the emulation process assembles and links the correct emulator, creates OS/3 job control cards to execute the emulator, and builds software images of the printer carriage control loops.

- Transfer a data base

  These aids transfer the user's data base on a bit-for-bit basis to the specified 90/30 disc.

When using the OS/3 emulation package, the user spends minimal time learning the operation of the 90/30 Data Processing System. Even though the users can modify the emulator they create, the existing programs may be emulated, by default, with only an understanding of the SPERRY UNIVAC 9200/9300 System or the IBM 360/20 System characteristics.

3. OS/3 System Software

3.1. SOFTWARE DESIGN CONCEPTS AND CAPABILITIES

Through central control of all activities of the 90/30 Data Processing System, the combined hardware and software capabilities are fully established and maintained to satisfy the requirements of all applications. The responsibility for efficient and flexible centralized control is borne by OS/3, which allows the programmer to use the system with relative ease, while relieving him of concern for the internal interaction between his program and other coexistent programs.

Design capabilities of OS/3 span a broad spectrum of data processing activities. Each activity can function independently or interrelate to support other activities without imposing inefficiency. Specific capabilities that are not desired by a particular installation may be eliminated at system generation time.

OS/3 is a comprehensive library of programs consisting of a flexible executive, a collection of language processors, utility routines, and application programs. The programs that make up the executive are the OS/3 Supervisor (supervisor) and Job Control (job control). Through its versatile job control language, OS/3 organizes and directs operations and system activities to achieve maximum use of computer facilities.

3.1.1. Software Design Concepts

OS/3 has been designed and implemented to offer an efficient multijobbing/multitasking environment that is needed to utilize the full capabilities of the 90/30 system.

The speed and hardware capabilities of the 90/30 system are used to maximum advantage, and a given hardware configuration is used most effectively in the complex internal operating environment created by OS/3. This environment allows for the concurrent operation of programs; for immediate reaction to the inquiries, requests, and needs of many different users at remote and local stations under the demands of message processing application; for storage, filing, retrieval, and protection of large blocks of data; and for optimum use of all available hardware facilities while minimizing job turnaround time.

Ease of use is emphasized in the system. Work to be performed by the system is described through the OS/3 job control language, which was designed to minimize job turnaround time and operator intervention. At his desk, the user may construct any logical combination of programs for a particular job by inserting the proper control cards in his job deck.

Job decks can be collected and entered into the system from many sources, central or remote. The executive controls the loading, allocation, and execution of the programs once they have been entered. Job steps that cannot be completed because of program error are automatically deleted from the system with appropriate diagnostic information. The console operator is, in effect, responsible only for participation in the data processing activities as directed by the executive.

3.1.2. System Capabilities

The ensuing paragraphs describe the functional capabilities of OS/3. Additional information regarding each capability is presented in those paragraphs that discuss the related software or hardware components of the 90/30 Data Processing System. The minimum configuration of the 90/30 system is: card reader, printer, two disc drives, and 32K of main storage.

- Communications Processing

  The executive efficiently responds to the demands of communications processing and gives preference to the operational needs of a message control program (MCP), these being the most critical requirements of the system. Programs of the executive dedicated to this type of processing receive the highest priority for scheduling within the system. The contingencies of message control are supported by procedures for rollout of conflicting user programs, system restart, and other necessary functions.

- Batch Processing

  Facility of job preparation and submission, with minimization of job turnaround time, is a design feature of the executive. User-assigned priority by job can provide preferential service when batch jobs are submitted from remote terminals, or where turnaround time is critical.

  The executive provides an operational environment for a high volume of jobs. The user may specify preferred priorities for certain jobs within a group of jobs; this fulfills the user's responsibility related to multijobbing scheduling to achieve machine optimization. Automatic job-to-job transition, communication within jobs, and associated services, such as logging and accounting, are automatically provided by the system.

- User Program Development

  The executive interfaces with a specific set of source code language processors that enable user programmers to employ COBOL, FORTRAN, basic assembly language (BAL), and report program generator (RPG II). The output from any one of these language processors (object module), when combined with the linkage editor, will produce a program (load module) that can be executed on the 90/30 Data Processing System.

- Disc-Oriented Processing

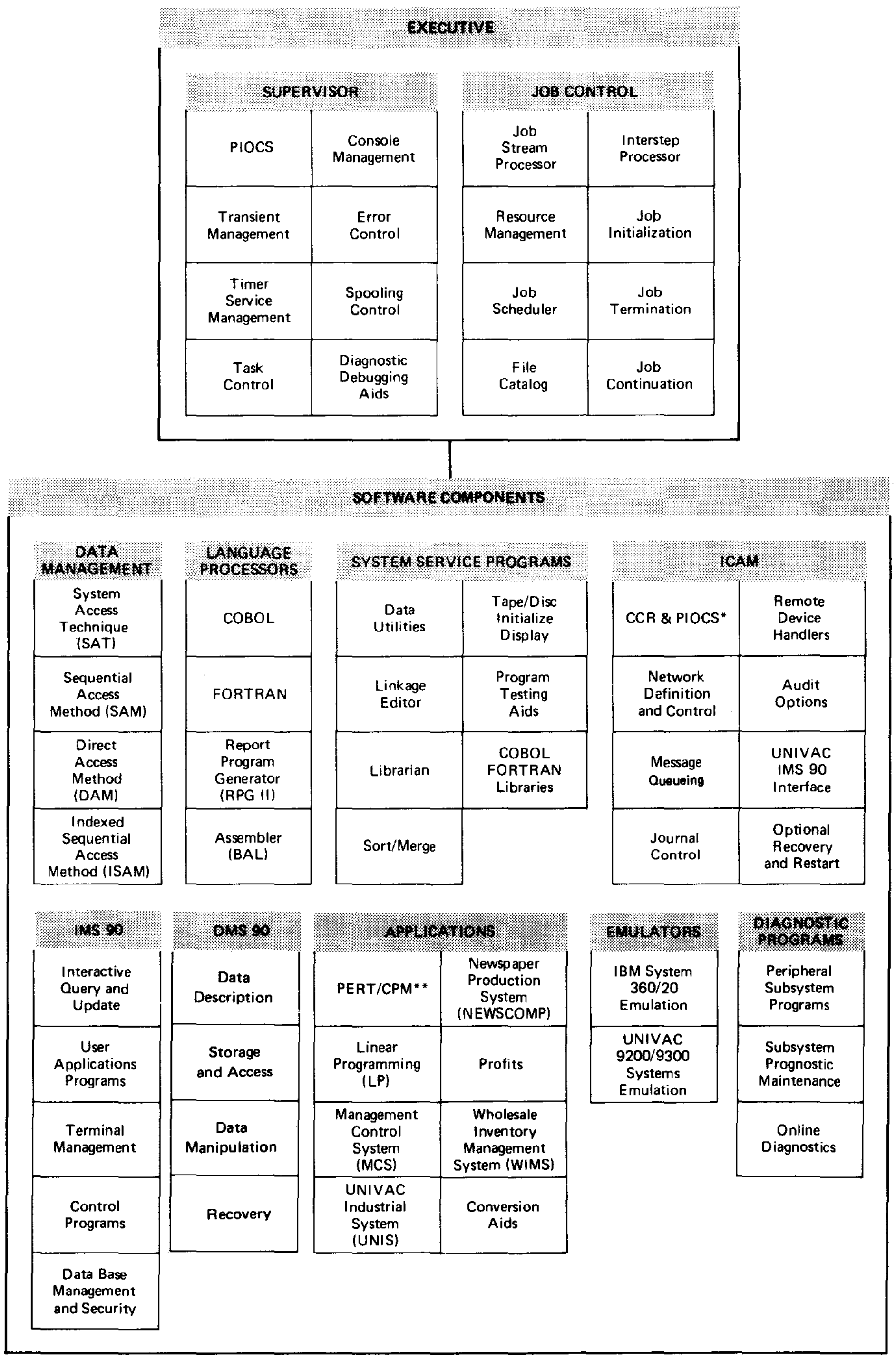
  The executive uses disc storage as buffers for accommodating any job backlog and for storing output data resulting from executed jobs. The buffering allows the system to operate independently of the essentially low-speed peripheral devices. All executable programs are obtained from disc storage through one of the system libraries maintained by job control. Temporary files required during program execution are generally assigned to disc storage rather than magnetic tape to facilitate disc-oriented processing. Operator participation is explicitly defined and is minimized as much as possible.

- Modularity

  The OS/3 is modular to facilitate future expansion. When generating a system, the programmer can tailor the operating system to the specific needs of the installation through the selection of the various modules contained in the software release package. This package is designed to be flexible and uncomplicated in detailing the procedures for generating an efficient operating system.

3.2. SYSTEM SOFTWARE DESCRIPTION

The programs of the OS/3 Executive — the supervisor and job control language — provide control for coordinating and executing system-supplied and user programs and for furnishing a flexible processing environment. In concert with a collection of programming languages, utility routines, and application programs, Sperry Univac provides the user with a viable program library to take full advantage of the extended capabilities of the 90/30 Data Processing System.



*Channel Control Routines and Physical I/O Control System
**Program Evaluation and Review Techniques/Critical Path Method

Figure 3—1. Operating System/3 Software Components

Subsequent paragraphs within this section describe the software components of OS/3. The presentation shown in Figure 3—1 is intended to give the system specialists background information about the design objective, organization, and functions of OS/3.

3.3. SUPERVISOR

The OS/3 supervisor consists of main storage resident routines that provide the central control, interface, coordination, and allocation of the hardware to achieve optimum system utilization. The supervisor also controls the initiation, loading, execution, and termination of user jobs.

Supervisor design is oriented toward reducing main storage requirements. To meet this goal, only those functions that are continually necessary to execute jobs or to control the system are resident at all times in main storage. Those functions involved with the initiation and termination of job steps, operator communications, and certain error contingency conditions are transients and are brought into main storage only when required. The functions and services provided by the supervisor are:

- Supervisor Interrupt Requests

  Services the various interrupts generated by the system.

- Task Switcher

  Determines the order in which the various tasks are given CPU time.

- Physical Input/Output Control

  Controls the dispatching, queueing, and interrupt processing for all I/O devices directly connected to the system.

- Transient Management

  Schedules, locates, and loads the transients that perform the nonresident supervisor functions.

- Timer and Day Clock Services

  Provides a system clock for timed, batch, or real-time activities.

- Error Control

  Handles any program or machine error that causes an interrupt.

3.3.1. Supervisor Interrupt Requests

There are six logical classes of requests made of the supervisor by means of system-generated interrupts:

1. Supervisor call (SVC)
2. Interval timer
3. Input/output
4. Program errors

5. Hardware errors

6. Operator request

3.3.2. Task Controls

A job may consist of several job steps. Each step may contain several interrelated and concurrent yet separate activities. These activities are known as tasks. A step may have from 1 to 256 tasks; however, it is unlikely that any step will approach this technical limit. It is the function of the supervisor to schedule these tasks for execution.

3.3.3. Physical Input/Output Control System

The physical input/output control system (PIOCS) handles the queueing and initiation of I/O commands and the processing of I/O completion interrupts. This part of the supervisor is composed of a set of specific routines to provide support for all planned I/O devices of the 90/30 system. Design emphasis is on maximum throughput for all devices, flexible maintenance, and expansion to new device types. Most of these routines are constructed as independent routines.

3.3.4. Transient Management

The supervisor takes full advantage of auxiliary storage to provide maximum services while keeping main storage requirements as low as possible. This is accomplished by having two types of supervisor routines:

1. Resident Routines

   Those software modules that are frequently used are made a part of the supervisor program that resides in main storage at all times.

2. Transient Routines

   Routines that are used only occasionally by the supervisor or by the user programs and are kept in the form of overlays on disc storage are referred to as transient routines. When needed, the transient routines are transferred from disc storage to main storage and activated. Once resident in main storage, the code in the transient routines can be used again when needed.

   The number of areas in main storage set aside to contain transient routines is specified by each user. An important function of the supervisor is managing these transient areas by monitoring their use, controlling the loading of a requested transient routine into a selected area, and transferring control to the transient. Only one disc access is required to load a transient. Open files, close files, and terminate jobs are examples of transient routines. If a transient routine's usage is heavy, it can be made resident at the option of the user.

3.3.5. Timer Service Management

The central hardware contains a high-resolution timer that can provide an interrupt after any time period greater than 1 millisecond. The calling job may specify a wait interval in milliseconds or seconds, or may specify a time of day at which an interrupt is to occur.

A simulated day clock provides the time of day to jobs upon request and is used by the supervisor for time stamping messages and job accounting entries.

3.3.6. Console Management

Console management provides for displaying messages on the system console, with responses and commands coming from the operator. The screen images are rolled upward, with new display lines or operator input appearing on the bottom of the screen. Console management routines selectively delete messages not requiring responses from the top of the screen. Console management is nonresident and is loaded as a series of transients when needed.

3.3.7. Error Control

Any error that causes a program interrupt is examined to determine the type of user or system interrupt, such as program check or protection violation, and the type of job.

The processing of error information by means of user-supplied subroutines is optional. Their entry point can be defined to the supervisor by various macro instructions. Standard error control actions are initiated in the absence of user code. If an error is detected, the system is brought to an orderly halt, and a restart from this point is attempted. If recovery fails, information is collected for an orderly abnormal termination.

Invalid or inconsistent requests for supervisor functions are reported to the requesting program.

Machine check interrupts are examined by the system to determine whether the error is recoverable. If not recoverable or recovery fails, the system is brought to an orderly abnormal termination. The pertinent information is collected and logged.

3.3.8. Spooling Control

Spooling is a technique that increases the throughput of the 90/30 system. Data from low-speed peripheral devices is transferred to disc storage independently of the program that will use the data. When a user program is logically retrieving data from a low-speed peripheral, it is physically retrieving the data from a higher speed device. On output, the user program logically specifies a low-speed device, but the images are physically recorded on disc storage and later, under system control, transferred to a low-speed device. Spooling of output allows concurrent use of a specified device by multiple programs.

There are several types of routines (Figure 3—2) used for spooling operations in the OS/3 system:

- input readers
- spooler cooperative
- output writers

Input readers accept data files from local or remote subsystems. When these data files are used, no distinction is made as to the method of submission.

The spooler routine stores this card data in the spool file until it is required by a user program. Data output from a user program is stored by the spooler in the spool file until a printer or punch is available.

Output writers provide local or remote users with output from user programs which receive identical service regardless of final destination. In addition, the operator of the printer can control forms recovery at page increments, control the number of copies, and control starting and stopping.

Figure 3—2 shows several active jobs utilizing the concurrent spooling capability of OS/3.

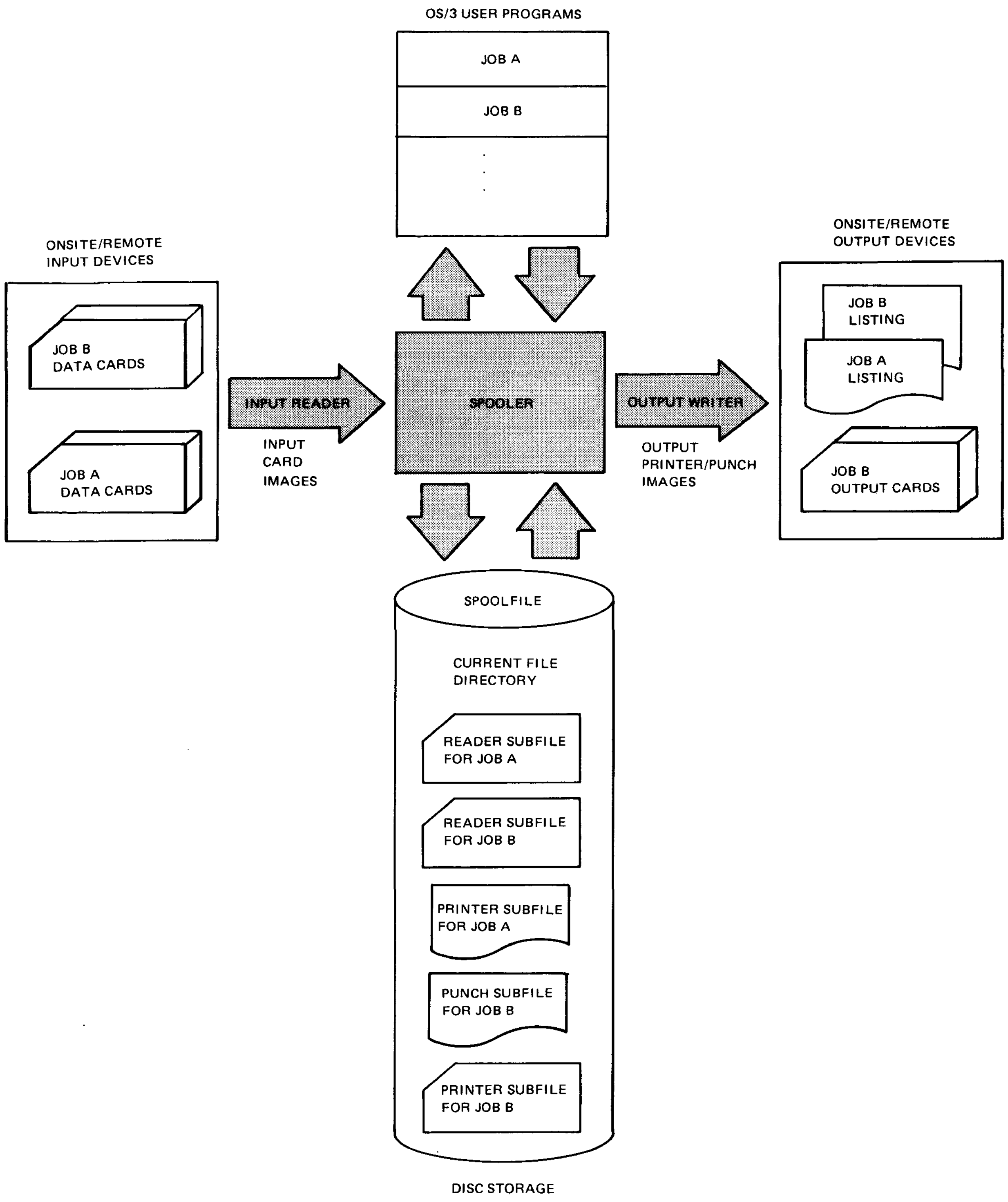


Figure 3—2. Spooling Flow

3.3.9. Diagnostic Debugging Aids

Diagnostic debugging aids in the supervisor are:

- Monitor mode
- Snapshot display of main storage
- Main storage dumps
- Standard system error message interface
- Error response to user jobs
- Program checkpoint restart

Descriptions of these aids are provided in the following paragraphs.

3.3.9.1. Monitor Mode and Trace

A hardware monitor interrupt enables the monitor routine to trace the execution of a program so that errors can be located and corrected. The routine interrupts each instruction before it is executed and tests for the conditions specified in the monitor input. When a specified condition is satisfied (a specified storage location, instruction, or instruction sequence is reached), the monitor can print out current program information (PSW and register contents, next instruction to execute, etc.) and can suspend program execution or continue with or without monitor intervention.

Trace conditions and information to be printed can be specified via the job control stream or entered at the system console. An entire program or a portion of a program can be monitored.

3.3.9.2. Snapshot Display of Main Storage

A partial storage printout at given points in a program can be obtained by means of a SNAP macro instruction when coded in the user program. It is also possible to specify, by parameters at run time, a portion of main storage to be printed and at which points the printouts are desired. This enables a program to be tested without recompilation to include and alter SNAP requests.

3.3.9.3. Main Storage Dump

A main storage dump may be provided for programs under the following conditions:

- Abnormal Termination Dump for User Program

  This provides a main storage dump of a region in hexadecimal, alphanumeric, or both, plus a formatted display of error codes, job-oriented tables, and supervisor information to assist the user in debugging.

- Program or Operator Request Dump

  This enables the operator or any program to request a main storage dump in the same format as the abnormal termination dump.

- System Failure Dump

  This is a routine intended for use when an abnormal condition occurs and other dump programs cannot be used.

3.3.9.4. Standard System Error Message Interface

An error message service routine provides complete and specific error messages. This routine locates the message in a disc file and transfers control to the system console handler for message display.

3.3.9.5. Error Response to User Jobs

Error codes that are returned by the supervisor to the calling program are standardized to provide a uniform interface for all system services.

If a user requires return of control after the detection of hardware failure or software exception, a user-supplied subroutine must be provided to handle this situation; otherwise, an orderly abnormal termination, which optionally may include a main storage dump, is invoked for the user job.

3.3.9.6. Program Checkpoint Restart

A checkpoint request facility provides for restarting a job with synchronization of all disc and tape sequential files. If unit record files are buffered to disc or tape, the position of these files may also be restored on restart.

3.3.10. System Generation (SYSGEN)

The complete OS/3 is delivered on disc packs or magnetic tape. Since hardware configurations vary and use of the OS/3 differs from site to site, it may be desirable to tailor OS/3 to the particular configuration needed. This tailoring is referred to as SYStem GENeration (SYSGEN).

The SYSGEN process is simplified through the use of the parameter presentation method, which is supplemented by a base of default definitions. With this as a basis, the user need only submit to the SYSGEN procedure the parameters required to satisfy his hardware and software requirements. Upon completion of the SYSGEN run, a disc pack containing the newly configured user operating system is produced. Automatic validation of the configuration is also available to the user to ensure that the proper hardware and software components were configured and generated.

3.4. JOB CONTROL

OS/3 job control consists of a group of nonresident routines that can be called into main storage rapidly and efficiently to manage system resources and prepare programs for processing. The job control language (JCL) enables the user to direct the activity of job control. A collection of JCL statements referring to a job is called a job control stream.

Job control streams specify the order in which the programs are loaded and executed, as well as the availability and assignment of main storage and peripheral equipment. Jobs may be entered into the system from onsite or remote devices.

The basic functions performed by job control are:

- The statements in the control stream are analyzed by the job stream processor. The information obtained from these statements is used to build the necessary control blocks that describe the requirements of each job to the system.
- Main storage areas and input/output devices are allocated. Resource management ensures that all necessary subsystems are assigned, that space is obtained on disc, that the operator has mounted any necessary disc packs or magnetic tapes, and that the amount of main storage needed to start the job is available.
- The job scheduler provides the means for selecting and initiating jobs according to the scheduling priority designated by the user.

The services of job control are directed by the programmer through the control stream or by the operator through system console commands.

The control statements furnish information regarding:

- job identification and priority
- file label information
- extent request
- storage allocation
- device assignment
- magnetic tape positioning
- job termination and deletion
- job parameters and data

Control streams are read from the system input device and queued by priority on disc for subsequent retrieval and execution. A control stream may contain data required during execution of the job, such as source code for an assembler or compiler task.

3.4.1. Control Stream Files

Control streams submitted to job control may be entered temporarily or permanently filed. Those entered temporarily in the control stream library are deleted after job execution. Those filed on a permanent basis by using a system console command are stored in a job stream library and are available for subsequent retrieval.

3.4.2. Job Scheduling, Initiation, and Continuation

Jobs submitted to the system are queued for initiation by a scheduling priority designated by the user. Jobs are initiated by this priority within the limits of available resources.

If adequate resources are available, they are allocated, and the job is established and identified in the system. If sufficient resources are not available, the job is bypassed until the next scan, and the next job in the queue is processed.

The initial switch-list priority level is established during job initiation. This priority, like the scheduling priority, is specified by a control statement and determines the sequence in which the supervisor gives control in the multijobbing/multitasking environment.

The continuation function of job control is activated at the end of each job step to provide interjob processing. Normal job step termination results from the execution of a supervisor macro instruction. This activates job control, which is loaded into the low area of main storage assigned to that job. Job control continues to interpret the control stream at the next control statement. If subsequent job steps are in order, they are initiated and established. This process continues until the end-of-job delimiter is sensed, at which time control is passed to the job termination routine.

3.4.3. Job Termination

The job termination function of job control is activated either for normal or abnormal termination. Job termination causes the release of all assigned storage and peripheral devices. Unprocessed control statements and unprocessed data in the control stream are bypassed.

Normal terminations are initiated by the user program, control stream, or system operator and are generally caused by an end-of-job statement. Jobs can be terminated by an operator using the STOP command; jobs can be suspended between job steps by using the PAUSE command.

Abnormal terminations are those caused by the detection of an error. The error may be a program error, control statement error, or the expiration of estimated processing time for the job. Abnormal terminations caused by other than a control statement error cause a main storage printout for programmer assistance in determining the cause.

3.5. DATA MANAGEMENT

The OS/3 data management consists of logical input/output processing modules, transient routines, declarative macro instructions, and imperative macro instructions. Data management acts as a convenient intermediary between user programs and the input/output facilities of the supervisor and assists in the task of accessing data files on various peripheral devices. An OS/3 supervisor input/output control system called system access technique (SAT) provides a standard interface between disc subsystem and the physical input/output control system (PIOCS) by efficient management of disc files. Data management relieves the user of the necessity of coding the routines for blocking/deblocking, buffering, and communicating with PIOCS.

Data management offers the following access methods:

- Sequential access method (SAM) — Magnetic tape, disc, card reader, card punch, and printer equipment
- Direct access method (DAM) — Disc storage devices
- Indexed sequential access method (ISAM) — Disc storage devices

The OS/3 access methods have a high degree of compatibility with the SPERRY UNIVAC 9200/9300 Series, IBM 360/20, and IBM/DOS.

3.5.1. Logical Input/Output Control System

The logical input/output control system (IOCS) modules that control each access method consist of reentrant common code subroutines and are capable of supporting a single or multiple program environment. When linked to a specific job step, these subroutines provide the access requirements of a single program. When referenced within a user program, these subroutines are dynamically loaded into main storage and can then be shared by many user programs.

3.5.2. Transient Routines

Some of the functions performed by the transient routines are:

- Data management — open and close files.
- Job control — cancel, end-of-job, and subroutines required when establishing jobs in the system.
- Supervisor — checkpoint, certain operator commands, and extensions of supervisory functions.

These routines are main storage resident within assigned supervisor transient areas only when required.

3.5.3. Declarative Macro Instructions

Declarative macro instructions permit the user to supply file description parameters, buffer and workspace location, option selections, and intentions for use of the file.

3.5.4. Imperative Macro Instructions

Imperative macro instructions permit the user to request that data management perform a specific function on a file; open a file, get a record, and put a record are examples.

3.5.5. Access Methods for Disc Storage

The disc subsystems capable of being attached to the 90/30 system are the 8411*, 8414*, 8416*, 8418, and 8430 disc subsystems. These subsystems vary in storage capacity and in data storage characteristics. The 8416/8418 disc subsystems are fixed-sector discs having 40 records per track, each record holding 256 bytes of data.

Data management accommodates variability in disc subsystems by using methods and storage formats that are within the capabilities of all the devices; however, means by which the user requests data management services are not changed for each device, and the overall data throughput is improved. The user can be independent, to a large degree, of the actual device being used. However, he should be aware of the following aspects that affect his file design:

1. The undefined-length specification for SAM and DAM records is not accepted.
2. The variable-length, unblocked specification is accepted for both SAM and DAM records. In both cases, the result is the placement of one record per block, in blocks of uniform size, equal to the specified buffer size.
3. All ISAM records are blocked in uniform blocks, each holding as many as its capacity allows.
4. Variable-length records are accepted by ISAM.
5. Each ISAM record placed on disc storage has a pointer field appended to provide for chaining to records inserted later.
6. There is no independent overflow in ISAM. When overflow space on a cylinder is exhausted, space is taken from remaining overflow space on other cylinders. Overflow records are blocked.
7. ISAM uses blocked indexes where each entry points to a prime data block. Tracks of these indexes are in the index extent; hence, there is no track sharing on prime data cylinders.

Subject to availability

8. ISAM can be specified to be without directory. In this case, all records stored in the prime data area are reached by sequential or direct access. Added overflow records are chained by pointers in the prime data area and are treated as appendages accessible by following the chain to the overflow data area.

9. All ISAM records are placed in unoccupied space and are not moved thereafter.

3.5.6. Access Method for Magnetic Tape Subsystems and Unit Record Equipment

Magnetic tape subsystems, card readers, card punches, and printers are devices on which the operations are performed in sequential order; that is, records are handled from the first to last according to physical placement.

For these four device types, OS/3 data management provides sequential access method (SAM) programs to handle the access requests of user programs.

3.5.7. Data Management Compatibility

Ease of conversion is provided for customers going to OS/3 data management. Names and functions of macro instructions are the same as those of other byte-oriented computer systems. The spellings of keyword parameters used in these systems are accepted to provide compatibility.

The selection of return points to user programs in other systems is not uniform. OS/3 data management returns inline to a user program whenever the requested function is completed. A special return can be established at which the user can insert procedure (PROC) call statements to handle exceptions and errors according to his requirements.

3.5.8. Disc Space Management and Volume Table of Contents (VTOC) Services

The creation and processing of disc files require procedures for allocating space, releasing unused space, creating scratch files, obtaining label and extent information, and renaming files. These procedures are required by job control, data management, and various service and utility programs.

Disc space management routines provide an efficient and completely automatic space accounting and maintenance feature. This relieves the user of the responsibility of knowing the precise contents of disc volumes. The routines also permit resolution of competing demands for allocation, along with establishment of standard interfaces.

3.6. SYSTEM SERVICE PROGRAMS

3.6.1. Data Utilities

The file processing utility allows the user to copy (with or without correction) an input card, magnetic tape, or disc file to any other card, magnetic tape, disc, or printer device. The user can also compare two files for equality. The various options allowed are specified by a series of control statements.

A utility control statement specifies the various options which the user desires. The available options are:

- Input and output device type specification

- Multiple output devices (one of which is either the printer or the punch) specification

- Copy or compare
- Print in LIST or DISPLAY format
- Fixed, variable, or undefined record format
- Input and output record size and block size
- Column binary or Extended Binary Coded Decimal Interchange Code (EBCDIC) card data format
- Tape rewind specification
- Optional writing of leading tape mark
- Write disc check
- Printed output data format (hexadecimal or character or both)
- Output card stacker select
- Output printer spacing
- Page numbering on printed output
- Position of first logical input record
- Sequence checking of input data
- Printer mismatch specification
- American Standard Code for Information Interchange (ASCII) tape specification
- Cancel compare function after specified number of nonmatching records
- Halt after specified number of input records (or blocks)
- Label checking and writing specification
- Block number checking and writing specification
- Edit by field selection

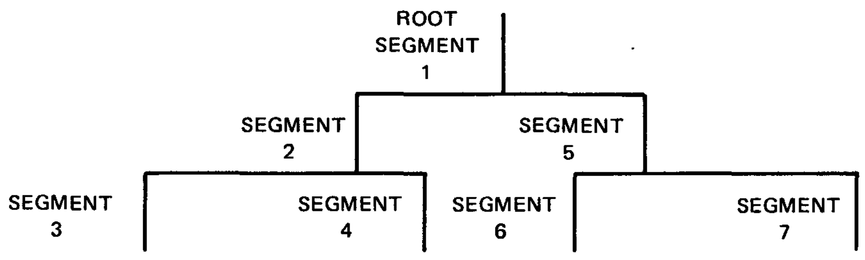
3.6.2. Linkage Editor

All OS/3 language processors (COBOL, FORTRAN, RPG II, and BAL) produce object elements as the output of their compilation processes. These elements may subsequently be structured into a user-tailored executable program by the OS/3 linkage editor. Object elements from various language processor compilations can be structured by the linkage editor into a single loadable program embodying the segment and overlay characteristics described by linkage editor control statements. Additionally, the object elements thus collected may contain cross-references to each other for specific purposes of program execution and communication. Such cross-references between separately compiled elements are resolved by the linkage editor when these elements are collected and a loadable program is constructed.

3.6.2.1. Loadable Program Structure

The loadable program that the linkage editor builds can consist of program segments fashioned into multiple regions. Each of these regions is structured as a hierarchical tree with specific boundaries inherently defined as branches of a tree trunk, or root segment. This structure permits a user to produce a program that is larger than the main storage area assigned for its execution. The various segments of a multiphase load module are loaded and executed as required by the logic of the program. The responsibility for loading the various phases of a program can be assumed by the user programmer, which requires that the appropriate FETCH and LOAD macro instructions be incorporated in his object elements, or loading may be left for the linkage editor to resolve based on the cross-references contained in the various object module elements. If the latter option is chosen, the cross-references contained in the object elements included in the load module must be the V-CON type.

The following diagram illustrates the division of a loadable program into segments.



3.6.2.2. Cross-Reference

Cross-references made by the user (within his compiled object elements) and resolved by the linkage editor in forming the loadable program may exist within or across segments. Segments are composed of one or more object elements, and the loadable program is composed of one or more segments, the initial segment always being designated as the root phase segment. The user's tools for establishing cross-references between processor object elements are:

1. Language processor external reference declarations (EXTRN) used to create indirect references in requesting object elements.

2. External definition declarations (ENTRY) used to create indirect definitions in satisfying object elements.

These declarations are used by the linkage editor to resolve cross-references between two or more object modules that are linked together at link-edit time. Executable program generation is, therefore, essentially a two-fold process:

1. The source program code must be compiled by the various language processors to produce one or more object modules.

2. The various object module elements that are to comprise an executable program must be combined into a single executable load module by the linkage editor.

At link-edit time, the linkage editor assigns, to the object module elements included in the load module, a new relative address based upon their new relative position within the load module. Any cross-references that exist between object module elements are satisfied by replacing them with the relative address of their respective definitions or references, as the case may be.

The output of the linkage editor is, therefore, a loadable executable program acceptable to the OS/3 loader.

The role of the linkage editor in program preparation is shown in Figure 3—3.

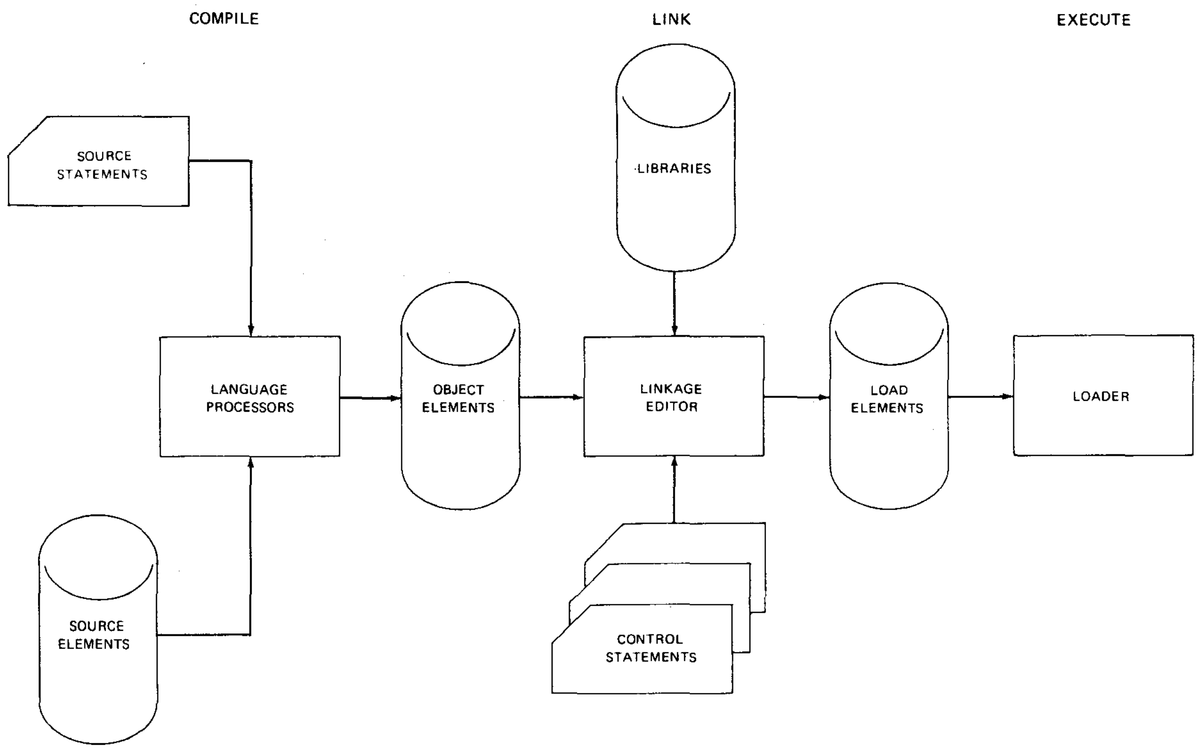


Figure 3—3. Program Preparation

The ability of the linkage editor to construct a single executable load program from several object elements has the following major advantages:

- If a change is required in one of the object elements included, only the specific object element involved must be compiled or assembled again.
- The various source elements may be written in the appropriate language, such as COBOL, FORTRAN, RPG II, or BAL, and combined into a single executable program.
- Routines common to two or more object elements need be assembled or compiled only once and the resulting object code linked as needed; the result is reduction in the total time required to generate an executable program.

In addition to the basic linking function, the linkage editor performs the following:

- Searches an appropriate library and incorporates object elements other than those in its primary input, either on request or automatically.
- Performs program modification by deleting and rearranging control sections of an object element as directed.
- Produces an optional overlay structure to be used by the supervisor during loading.
- Reserves space automatically for common storage requests generated by the language processor.

3.6.3. System Librarian

The OS/3 system librarian is a set of integrated subroutines existing as a separate system entity and functioning as a housekeeping and maintenance tool for the system and user libraries. Programs and elements such as language processor source code, language processor output (relocatable object code), or system-executable load programs reside within a library (which is either a system or private file in the OS/3 environment). The storage, collection, creation, modification, correction, deletion, addition, duplication, and transposition of library elements are the primary tasks of the librarian. These housekeeping functions may be performed on entire program libraries, groups of program elements within a specified library, or individual program elements.

The librarian also transfers program library elements from one medium to another. While library elements reside primarily within disc files, the librarian is capable of transposing program libraries to or from a card or tape medium.

Control statements supplied to the librarian via the control stream are the means by which the user directs the performance of the operations desired. Through the use of control statements, the librarian initiates such functions as copy, delete, add, compare, merge, compress, correct, group, rename, sequence, reproduce, list, and punch.

The librarian may modify existing libraries, create new libraries via copy or merge operations, or duplicate library files in their entirety. The librarian also ensures that a given library file does not contain any elements of identical name and type.

The listing and punching capabilities of the librarian allow a user to examine the contents of individual or groups of program elements. An element may be listed by the printer as well as punched into cards. The user also may obtain a table-of-contents-type listing of all the program elements in a file. A correction facility allows modification and updating within program elements existing in a library file. Source elements may be corrected or updated, as well as sequenced. Whenever any program element is processed by the librarian, its format is verified.

At the option of the user, the librarian provides a map of the functions requested and the results of the action taken, along with diagnostic error indications and warnings, as appropriate. The extent and amount of information provided on the librarian control statements coded are dependent upon the options specified by the user. At the descretion of the user, the control statements themselves may be a part of the library map.

3.6.4. Sort/Merge

The OS/3 sort/merge can operate in one of two modes:

1. A stand-alone program defined and initiated by job control

2. A modular subroutine integrated into a user program

Sort/merge has the same capabilities in either mode:

- An interface that permits disc or magnetic tape for use as work areas

- Input and output that may be associated independently with disc or magnetic tape

- Sorting of either fixed-length or variable-length records

- Handling of six types of key field formats:

  — Character

  - Unsigned binary
  - Fixed-point integer (signed binary)
  - Packed decimal
  - Zoned decimal
  - Floating point
- Specification of up to 255 key fields
- Sorting of noncontiguous key fields in ascending or descending sequence in any combination
- User specification of collation sequence
- Execution of output own code
- Execution of input own code
- Elimination of records of equal fields by specifying a keyword parameter
- Shared input and reserved output devices
- A separate merge
- Data checksum
- Convenient restart procedure

3.6.4.1. Sort/Merge Program

The sort/merge program is constructed as a processor that consists of interrelated modules operating within the framework of a system-driver program. Each module is designed to perform a specific function. As sort/merge progresses through the execution phases, the modules required are called by the driver program, loaded into main storage, and executed.

3.6.4.2. Sort/Merge Subroutine

The subroutine concept of constructing sort/merge allows the user flexibility and freedom with respect to the external format and source of input records and to the external format and disposition of the output records. The subroutine can be called from a language processor program that includes the verb to specify this operation. The modular structure gives the user efficient operation despite variations in hardware configurations and data requirements. The modular structure allows a module to be changed, replaced, deleted, or added.

3.6.4.3. Configuration Restrictions

For a disc sort, the available disc storage must be large enough to contain the entire file of records plus 10 percent.

For a tape sort, a minimum of three tape units is required. A maximum of 14 tape units can be assigned; a maximum of six tape units can be used by the sort as work units. The sort allows a user to execute a tape sort by means of the shared input and reserved output devices.

3.7. LANGUAGE PROCESSORS

Four language processors (COBOL, FORTRAN, RPG II, and BAL) allow flexibility in preparing programs for use with OS/3.

COBOL is a language for data processing problem solutions involving the maintenance and processing of large volumes of files; FORTRAN is a language oriented to computational problems. The source program written in COBOL or FORTRAN is input to a COBOL or FORTRAN compiler, respectively. The compiler translates the COBOL or FORTRAN source program into object code.

RPG II is a programming language that is not machine oriented, thus reducing the need for coding extensive instructions to direct the compiler to achieve the desired result. The RPG II compiler translates a source language description of report requirements into an object language program that accepts data and, from it, produces a report conforming in content and format to requirements specified by the user.

The symbolic language of the assembler is a versatile method of writing programs through the use of mnemonic instruction codes, assembler directives, data generation macro instructions, and procedural calls.

3.7.1. COBOL

Common Business Oriented Language (COBOL) is a programming language oriented toward problems in business applications. The language is similar to the English language, rather than a notation which considers the technical aspects of a particular data processing system. The source programs are easily transferable among systems that accept *American National Standard COBOL X3.23—1968*. Each of these systems provides a COBOL compiler to translate the COBOL source program into a machine-oriented object program. The ability to advance from one generation of equipment to another in a logical, orderly, and rapid manner is assured through this limited machine dependence. Source programs written in COBOL consist of four major divisions:

1. Identification

   This division contains information identifying the source program and the output of a compilation; the author, installation, and so forth also may be identified.

2. Environment

   This division specifies a standard method of expressing those aspects of a data processing problem that are dependent on the physical characteristics of a specific system; also, it permits specification of the compiling system hardware characteristics, input/output control techniques, and so forth.

3. Data

   This division describes the data that the object program is to:

   a. accept as input;

   b. manipulate;

   c. create; or

   d. produce as output.

   The division is further divided into sections to facilitate the description of data contained in input or output files or developed during the program, or which is present as constant information to be used in the object program.

4. Procedure

   This division describes the logical steps that must be taken in the solution of the data processing problem.

Two COBOL compilers are provided for OS/3. Both conform to *American National Standard COBOL, X3.23—1968*. The two compilers are:

1. Basic COBOL Compiler

   The particular language characteristics are in accordance with the requirements of *American National Standard COBOL, X3.23—1968*. The following levels of American National Standard COBOL modules are included:

   - Level 2
     - Table handling
     - Sequential access
     - Random access
   - Level 1
     - Nucleus
     - Sequential access
     - Random access
     - Segmentation
     - Library

2. Extended COBOL Compiler

   The particular language characteristics are in accordance with the requirements of *American National Standard COBOL, X3.23—1968*, with the exception of the report writer. The levels of American National Standard COBOL modules are:

   - Nucleus — level 2
   - Table handling — level 3
   - Sequential access — level 2
   - Random access — level 2
   - Sort — level 2
   - Segmentation — level 2
   - Library — level 2

3.7.2. FORTRAN

FORTRAN is a programming language designed primarily for performing mathmetical computations required for solving engineering and scientific problems. FORTRAN is also useful for many nonscientific data processing applications.

FORTRAN is designed so that the user can express an algorithm in a way natural to the problem. The user requires minimal considerations of the particular characteristics of the system in which the program is executed. Procedures defined outside the FORTRAN program, and possibly written in a language other than FORTRAN, can be referenced by name and thereby be made an implicit part of the program.

Two FORTRAN compilers are provided:

1. Extended FORTRAN is a proper superset of the *American National Standard FORTRAN (X3.9—1966)*. It is also a compatible superset of the IBM/DOS 360 FORTRAN IV. This system features code optimization, high-performance I/O, and extended functional capability.

2. Basic FORTRAN is designed for greater compatibility with the FORTRAN offered by the IBM 1130 and System/3 and for increasing the level of multijobbing in smaller configurations. Basic FORTRAN does not contain all of the more sophisticated FORTRAN features, such as COMPLEX arithmetic and LOGICAL data types, but the system includes the LOGICAL IF statement and a large mathematical library. This system is a subset of Extended FORTRAN.

For both compilation and execution of FORTRAN programs, the micrologic expansion feature is required in the 90/30 processor.

3.7.3. RPG II

The OS/3 report program generator (RPG II) is a processing language that generates an object program from a series of interrelated specifications. RPG II, unlike most other programming languages, does not require the user to establish the logical flow of the program. Through the various RPG II specifications forms, the user can define records and indicate the operations to be performed on these records. Each record is then processed by the same RPG II program cycle, and the user-specified operations are performed, provided previously and presently specified operation conditions are satisfied.

To define a report program for generation, the requirements are listed by the programmer on seven specifications forms. The forms are:

- File description specifications
- File extension specifications
- Line counter specifications
- Telecommunications specifications
- Input specifications
- Calculation specifications
- Output specifications

The information coded on these forms is punched into cards that become the actual input into the system. The compiler generates an object module for input into the linkage editor; the subsequent output is a loadable program module.

3.7.3.1. Enhancements to RPG II Language

Numerous enhancements have been included in the RPG II that enable the user to develop more sophisticated report programs within the existing framework of the language. These enhancements together with statements which summarize the extent of their modifications, are:

- Telecommunications interfaces are supplied via a new RPG II specifications form offering extensive communications capabilities to the RPG II user.
- The system console is now accessible through programmed operations.
- Input file chaining is now possible at calculation time by means of a CHAIN operation code.
- A DEBUG operation code is included to aid the user in source-level debugging.
- AND/OR relationships are allowed at calculation time for conditioning of operations by more than one line of indicators.
- A square root operation code has been added to aid in arithmetic calculations.
- The ability to define an internal RPG II subroutine that may be executed during detail or total calculations. Execution may be initiated anywhere within the calculation specifications, with control being returned to the calculation line which immediately follows the CALL line.
- The ability to edit fields by specifying a single-character edit code rather than the traditional edit mask. Edit codes are included for all common requirements.
- Output of duplicate portions of the same record is facilitiated by the addition of a new command. This allows side-by-side listings to be created without respecifying the individual output fields.
- The addition of an operation code that causes output to be written during calculation time (either detail or total). Upon completion of the operation, calculations are resumed at the line following the output command.
- The facility to look ahead to the next record awaiting processing within any sequential file. The contents within specified look-ahead fields may be used for selecting the next record to be processed or for calculations or output.
- The addition of eight control-stream time user switches. These switches (indicators) may be used to condition calculations, input files, output files, or specific output records.
- The addition of READ and FORCE operation codes to offer the user additional controls over the selection of input records.
- Translation to allow character-by-character replacement for specific files. Translation tables are created by the programmer to indicate desired character changes.
- The ability to define an array within an RPG II program as intermediate internal working storage. The array may be created in storage or read as data. Addressing is accomplished by indexing of the like fields in the array.

3.7.3.2. RPG II Compatibility

OS/3 RPG II provides an expanded range of features in comparison with the 9200/9300 RPG systems; however, 9200/9300 RPG programs written for card, tape, or disc systems can be compiled and executed on the 90/30 system by means of the OS/3 RPG II compiler. This source code compatibility is extended to encompass RPG programs written for the IBM System 360/20. Feature similarity also exists between the SPERRY UNIVAC OS/3 RPG II compiler and the RPG compilers offered with the IBM System/3 and the IBM 360/DOS System.

3.7.4. Assembler

The OS/3 assembler is a versatile and detailed symbolic language. The combination of a macro facility and the ability to handle procedural directives within the assembler gives it unique capabilities. Each instruction within the language is assigned a mnemonic to denote the particular hardware function performed.

The symbolic format for writing an assembler language instruction, macro instruction, or procedural call consists of three basic symbolic fields; use of the fields requires conformity to simple rules so that efficient translation of symbolic to object code can be performed. The three fields are:

1. the label field, which may contain a symbolic name used to provide an entry point or a label for a block of data or a block of instructions;

2. the operation field, which must contain a mnemonic instruction code or the name of a macro instruction or procedural call; and

3. the operand field, which provides for a variety of uses ranging from simple to complex specifications.

Combining names, parentheses, arithmetic operators, and self-defining terms into operand expressions makes possible the use of highly sophisticated coding sequences. Operand expressions gain power by being able to include location counter references and literals.

A wide range of data types is provided for constant generation and storage definition. Binary, hexadecimal, decimal, fixed-point, floating-point, and character formats can be used to specify absolute values in the source code.

Output from the assembler run consists of a complete listing of symbolic coding, generated object coding, diagnostic messages, and a cross-reference listing. A relocatable object module is produced that is suitable for linking to other modules prior to loading for subsequent execution.

The assembler includes a set of directives that can be used to specify instructions to the assembler itself. These directives allow the user to control program sectioning, base register assignment, the format of the output listing, sequence checking, and other auxiliary functions.

The assembler includes a macro facility that can reduce the effort required in writing patterns of coding either repeated in one program or common among several programs. The flexibility of the macro facility allows a macro to be written so that the pattern of coding generated can vary widely, depending upon the parameters supplied with the call. Macro definitions may be specified in two formats: MACRO and PROC. The elements of each format type may not be mixed within a definition; however, definitions of both types are permitted within a program. The OS/3 provides the user with a comprehensive selection of system macros that interface with data management, the supervisor, and other elements of the 90/30 software.

3.8. INTEGRATED COMMUNICATIONS ACCESS METHOD (ICAM)

As with all the OS/3 software, the design emphasis for the ICAM package is on ease of use for the casual as well as the major real-time communications user. A wide range of system options provides support for users committed to a real-time multijobbing environment without penalizing those whose needs are more modest.

At system generation time, the OS/3 communications interface is tailored to each user's requirements. Levels of support may be arranged by combining the appropriate ICAM library elements. The levels are named for the interface between the user's message processing program (MPP) and the ICAM components combined into a loadable software module (message control program). Each interface contains its own unique set of macro instructions.

- Standard Interface (STDMCP)

  The STDMCP is a conventional GET/PUT level interface for communications that automatically queues input and output messages via network buffers. In this interface, data is requested from a process file (input queue) and not directly from the line. Conversely, output is placed in a destination queue (output queue) for transmission by the message control program (MCP). An extended set of communications control area (CCA) macro instructions known as message processing procedure specifications are optionally available to this interface for additional formatting and processing of messages.

- Transaction Control Interface

  The transaction control interface is an interface designed for efficient processing of transaction programs, including the information management system (IMS 90). Messages may optionally be stored on disc or in main storage. The user may select disc queueing or disc buffering, depending on whether he chooses to operate in transient or resident mode. Text fields may be previewed and selectively retrieved for immediate processing in main storage or stored on disc for deferred processing.

- Device Handler Direct Data Interface

  User message processing programs may interface directly with the remote device handlers via a special configuration of the ICAM library elements. This level of support is intended to provide a communications capability in the smallest systems while maintaining a degree of device independence for the user's program.

- Communications Physical Interface

  Here the interface between the ICAM components and the user's program occurs at the physical input/output level with the channel control routine. This level utilizes less main storage with the greater part of the programming effort to provide communications support being shifted to the user's message processing program. The users are permitted to write their own message control programs without modifying the operating system interfaces.

Each level represents a different trade-off between the amount of main storage required when the ICAM components are attached to the system and the variety of services provided. While choosing the MCP level seems expensive in terms of main storage usage, writing a message processing program at the channel control routine interface level can be both difficult and costly if many services are required.

3.8.1. Message Control Program (MCP)

The MCP is a modular software package that is capable of supporting either simple or complex communications environments. A single MCP can provide concurrent support for multiple user message processing programs that use a variety of terminals and line types. It is composed of ICAM library elements that interface via the supervisor with the user's message processing program.

The MCP prevents conflicting facility assignments and releases facilities when jobs terminate. User programs are provided with macro instructions that perform the following services:

- control table generation;
- data transfers to and from user-specified buffer areas;
- communication facility initialization and control; and
- dynamic terminal and poll table entry alterations in the communications control areas.

3.8.1.1. MCP Components

The OS/3 ICAM components that make up the MCP are:

- Channel Control Routine (CCR)

  The CCR provides the physical input/output interface to the communications adapter (CA) and the specific type of communications subsystems.

- Remote Device Handlers (RDH)

  The remote device handlers provide the software logic and control required to interface the unique characteristics of specific remote devices to the other ICAM components.

  There is an RDH provided to accommodate each of a wide range of terminal equipment, including the UNISCOPE 100 and 200 Display Terminals; the DCT 500, 524, 1000, and 2000 Data Communications Terminals; the 1004 and 1005 card processor systems; the 9200/9300 system; and TELETYPE teletypewriter models, 28, 33, 35, 37 and 38. Binary synchronous communications procedures are also supported.

  Full-duplex communication to a SPERRY UNIVAC 1100 Series system (via the NTR system utility) is also available.

  Device handlers provide all device-dependent functions that are required to permit other ICAM components to function independently of the terminal device. These device-dependent functions include:

  — station and device polling where applicable;

  — code translation;

  — compression and decompression of data where necessary;

  — error detection and correction; and

  — accumulation of operation statistics.

- Communications Network Controller (CNC)

  The CNC coordinates message flow between the remote device handlers and either main storage or a disc-based message queue. The CNC is the MCP component that places incoming messages on their appropriate processing queues or submits them to special system functions (the user program interfaces described later) for disposal. It provides the interface to user-specified message processing procedures specification (MPPS) routines and user-generated routines. The CNC also detects the presence of a message on an outgoing (destination) queue and provides the orderly transmission of that message.

Additional functions performed, either directly or indirectly, by the CNC include:

- providing the control to dynamically modify a network in accordance with a changing operating environment;
- monitoring the orderly shutdown of the MCP during end-of-job processing (this service is extended to both the MCP and user programs); and
- scheduling the message user service transcriber (MUST) routines when new activity is detected and there is an outstanding request waiting.

■ Communications Control Area (CCA)

A CCA contains all the tables required to define and control a specific communications network configuration. User message processing programs may assemble CCAs tailored to their specific needs or they may identify and link standard CCA elements from the system library. A user message processing program may also specify a network CCA by using a control card with a network name in the run stream. Job control then automatically locates the CCA in the system library and loads it at job execution time.

Network control is exercised by the setting of indicators and flags in the line, terminal, and queue tables within the CCA. These flags control polling, indicate the operational status of communication hardware, and reflect the current disposition of message queues.

Each CCA associated with a user program contains a pool of network data buffers that are under the control of the CNC. Incoming and outgoing messages are temporarily staged in these buffers during their active transition through the system. Buffers from this pool provide the base when main storage message queueing is specified. When disc storage queueing is specified, these buffers provide intermediate storage during the active input or transmit phase of a message.

■ Message User Service Transcriber (MUST)

A MUST routine provides a message staging service that isolates a user program from the device dependence that is usual in data communications programs. MUST is responsible for copying data into user-designated buffer areas from the network buffer pool or copying data into the network buffers from user-designated buffer areas.

Variations of a MUST routine provide support for specific user program interfaces, such as remote batch processor (RBP), RPG II, and IMS 90. The MUST routine isolates these functions from the CNC and device handlers, obtaining the maximum commonality of MCP components.

User programs are capable of receiving or sending message data by exercising one or more options when calling the MUST routine. Both batch data and nonbatch modes of operation are provided.

■ Deferred User Service Transients (DUST)

The deferred user service transients are a series of MCP overlays which perform those functions that are not time-dependent, or that are used infrequently. An overlay control routine within the CNC provides queueing for multiple requests for DUST functions because only one overlay may be active at one time.

The overlays provided by DUST perform functions such as:

- MCP initialization
- CCA initialization

— subsystem parameter loading

— line connect and auto dialing

— program termination

— system console message capability

3.8.1.2. Special System Interfaces

The MCP always includes the assembly language interface as a matter of course, but other system interfaces may be included, depending on the user's requirements. These special system interfaces are: an RBP capability, an RPG II telecommunications capability, the interactive, transaction-oriented IMS 90, the ICAM device emulation system (IDES), and the NTR system utility.

- RPG II Telecommunications

  RPG II users are provided with a simplified remote I/O capability via an RPG II telecommunications interface. This capability is provided by standard ICAM components and the MUST optional routine. Support for RPG II telecommunications is extended to all the devices already listed under remote device handlers, as well as the IBM System/3, the IBM System 360/20 (BSCA equipped), and the IBM System 360 (with OS or DOS BTAM binary synchronous communications support).

- Remote Batch Processing (RBP)

  An RPB capability is provided for the OS/3 ICAM; this permits jobs to be entered into the computer system from a variety of remote terminal decvices. The software components that perform the RBP function operate as system tasks and are referred to as symbionts.

  These symbionts may be configured as input only, output only, or input and output. In small computer installations, the execution of the input symbiont, the user's job, and the output symbiont occurs serially. In larger systems, the entry and execution of a specific job are serial, but more than one job may be processed concurrently.

  Remote symbionts may be loaded and initialized manually via the system console, or these remote jobs may be processed automatically in response to a call from a remote station. An instruction set is available to send job requests, activate work stations, defer processing, obtain status, and deactivate work stations.

- IMS 90 Communications

  IMS 90 is an interactive, transaction-oriented data management system. It is designed to provide communications data management that is easy to implement. This support is in the form of action programs supplied by Sperry Univac that require no programming effort by the user, or it may be in the form of a communications network and file processing system for user applications written in COBOL or basic assembly language.

  The transaction control interface that may be included in the MCP provides special logic to control the input and output of inquiry/response messages. The logic prevents the building of message queues in main storage that cannot be serviced immediately, so that only enough messages are maintained in the system to drive the IMS at its maximum capacity. This procedure provides maximum economy of main storage without impairing the throughput or response time.

- ICAM Device Emulation System (IDES)

  The IDES permits a SPERRY UNIVAC 90/30 System to emulate a SPERRY UNIVAC 1004 Card Processor, a SPERRY UNIVAC DCT 2000 Data Communications Terminal, or an IBM 2780 terminal. With IDES, a remote computer system (referred to as the host system) can perform remote batch processing via the 90/30 system as though one of these terminals were physically connected. The host system may be:

  — another SPERRY UNIVAC 90/30 System;

  — a SPERRY UNIVAC 1100 Series system;

  — an IBM 360 or 370 System; or,

  — any computer system capable of supporting the emulated remote batch terminals for reading and punching of card images and handling of print images.

  IDES is defined to the system when network definition is specified. To initiate remote batch processing via the 90/30 system, a network descriptor card must be included in the job control stream. This card contains parameters that define the communications network (including the device being emulated) and cause the start of the operation. Subsequent control of the remote operation is accomplished by use of commands entered through the 90/30 system console. These console commands allow the operator to control input, output, termination, and line functions.

  Input and output data to the system (i.e., reading or punching card images and printing functions) are handled by standard OS/3 data management macro instructions.

- NTR System Utility

  The NTR system utility gives a user the ability to have his 90/30 system act as a remote terminal to an 1100 system. The NTR is a combination system utility and job task that may run concurrently with other job tasks. It makes use of the direct data interface configuration, with some modification, as the connection between the two systems. The user is afforded data transmission control, operator-to-operator communications, and control features for status and command functions between both systems. The system utility controls local input and output tasks through use of a combination of data management macro instructions and a set of special NTR macro instructions for user own code control of device buffers.

The MCP interface to IMS 90 provides many services:

— It controls the input polling of terminals and lines by setting indicators in the associated terminal tables.

— It can receive and relay the input completion status to the IMS 90 program.

— It operates in a privileged mode with access to all MCP facilities and the IMS 90 resident buffer areas.

— Optionally, this IMS 90 interface can transfer messages to the IMS without prior queueing in the MCP.

— A unique supervisor call is provided to interface the IMS 90 program to the MCP.

— Provision is also made for holding an output message on a special holding queue until it can qualify for transmission.

3.8.1.3. MCP Enhancements

The MCP enhancements to the system are as follows:

- Message Queueing

  Message queueing is the stacking of complete messages in main or disc storage while they are waiting to be serviced by a communications line handler or a message processing program. A single message queue consists of one or more messages with their header segments linked together. The message text that overflows the header segments is contained in additional segments, which have secondary links out of the header segment. A network may contain one or more message queues associated with lines, terminals, or process queues.

  A user, when defining the network configuration, may select line or terminal level message queueing. Terminal level queueing results in a single queue being defined for each terminal on a communication line. Line level queueing results in a single queue being defined for all terminals associated with the line. When line level queueing is specified, the message header prefix contains the terminal name to which a message is directed.

- Multiple Destination Routing

  The disc queueing facility provides multiple routing of messages. Provision is made for allowing up to 255 destinations for a single message. Only a single byte is required in main storage for determining when a message has been delivered to all of its destinations. The message is released from its queue only when all deliveries have been made. Any mix of nonbatch devices may be designated in a single multirouted message.

- Activity Scheduling and Priority Control

  An activity scheduling routine is provided with an optional priority suspension and scheduling capability. In small system configurations, only a single level of scheduling is provided, and a suspension capability is not required. Multiple priority levels and a suspension capability are provided for large systems with critical timing requirements.

- Timer Service

  A centralized timing service for control of active data buffers and scheduling an activity is provided for use by all MCP software elements. The user can specify a value, in 1-second increments, to the timer for determining the maximum time period an active buffer can wait for data transmission. In addition, the user can specify a value to the timer routine, in 0.1 second intervals, for determining elapsed time prior to the scheduling of an activity.

- Checkpoint/Restart

  Restart procedures can reconstruct message queues to their status at the point of system hardware or software failure. The reconstruction involves only complete messages; incomplete messages must be reprocessed after the recovery operation. Status messages are transmitted to remote terminals to effect an orderly restart and provide assurance that messages are not lost in the system. Restart is supported only in conjunction with a checkpoint and journaling feature. Checkpoint results in the complete CCA being written on a tape or disc file. When checkpointing has been specified, it is performed automatically by the MCP on an adjustable periodic basis.

- Journal Control

  Journal files are required when a user of the MCP requires a recovery/restart capability, or when a statistical accounting of the network operation is required. The journal control routine provides for the recording of message processing events and message data on a disc or magnetic tape file. In order to reduce the number of I/O accesses required, each journal entry is transferred into a larger buffer area for staging prior to writing on the output device. A method of queueing journal requests by priority permits preferential treatment for critical entry processing.

- Statistics Accumulation

  The MCP may maintain an accumulation of statistics that reflect the operating status of the communication environment. These statistics are maintained by line and by terminal and are available to a user program upon request. The accumulation contains totals for types of information such as:

  — number of messages received

  — number of input retransmission requests

  — number of messages transmitted

  — number of output transmission requests

  — number of poll messages

  — number of no-traffic responses

  These statistics are maintained in the CCA and provide the user with information not generally available from journal files.

  A diagnostic log used to accumulate error statistics associated with lines and terminals is also included. These statistics are available to online diagnostic routines that are loaded when an operational failure is detected by the MCP.

3.8.2. Standard Interface (STDMCP)

In the standard interface, the user program requests data from a process file and places data on a destination queue. It is thus relieved of the burden of soliciting input or transmitting output directly on a line.

A communications packet is constructed and formatted by means of a DTFCP declarative macro instruction. The communications packet thus defined references a process file or destination queue previously defined in the communications control area (CCA). (The process file is a queue for input messages, while the destination queue is for output messages and is associated with a particular line.)

The GETCP/PUTCP imperative macro instructions identify the specific packet with the process file or destination queue name, submit the DTFCP address to the line handler and transfer the message accordingly.

Messages sent or received in the network are processed according to a set of message processing procedure specifications (MPPS) optionally set by the user at network generation. If none are specified, a basic default MPPS is supplied.

3.8.3. Transaction Control Interface (TCI)

The transaction control interface is similar in component structure to the standard interface but its procedures have been modified to allow more efficient processing of transaction programs. Message staging may be optionally selected to be in main storage or disc by means of the MREAD/MWRITE functions. The principal means of control is the transaction terminal table which is contained in the user's program area. This table is generated by the MTABLE declarative macro instruction. Deferred processing may be selected by an MDEFER macro instruction that causes messages to be written onto disc or into a linked buffer pool. An MSWITCH macro instruction permits the user to route messages without having to perform an MREAD. A simple preview of the message will suffice.

3.8.4. Device Handler Direct Data Interface

User programs may interface directly with the remote device handlers via a special configuration of the MCP. At this level, the MCP does not contain the standard CNC and MUST routines. No message queueing, network buffering, or MPPS functions are supported. If any of these services are necessary, they must be included in the user's message processing program.

The primary purpose of this interface is to provide a communication capability with reasonable device independence in small configurations of the 90/30 system. A simple method of line and device identification is provided in order to reduce the complexity of network definition.

The ability of users to implement special applications that could not be efficiently supported by the standard MCP interface is another benefit of this MCP configuration.

When the device handler direct data interface is used, data transfers to and from remote devices are made directly in and out of user-specified buffer areas. A buffer chaining capability is provided, along with a checking ability to detect buffer late conditions. An end-of-message condition code is also provided; this enables the user to exercise control over communications lines and devices without special calls to the MCP.

The user's message processing program interfaces with the remote device handlers by submitting the address of a message control table (MCT) to the MCP. The same MCT may be used for both input and output in a half-duplex environment. A single imperative macro instruction, MCPCALL, is used to submit the address of this table to the MCP.

By using the MCT, the user may perform the following functions:

- Opening, connecting, or assigning a line
- Disconnecting or releasing a line
- Sending data
- Receiving data
- Setting either the batch or interactive modes

This level of ICAM support has many advantages for the user who is willing to invest more programming effort in order to gain a more flexible data communications capability.

3.8.5. Communications Physical Interface

User program message processing may interface with a communications facility at the physical input/output level. This interface between the user program and the supervisor-resident channel control routine (CCR) is effected via a standard software packet and calling sequence to the supervisor. The packet is the communications physical input/output control packet (CPIOCP); the CPIOCP is submitted with software command codes and is returned with software status codes. These codes are standardized so that the message processing program need not recognize or generate specific hardware codes rquired by the communication system hardware. At this level of support, the user must provide almost all the communications services related to unsupported devices. This interface is primarily intended for those users who wish to completely design their communications system while retaining the advantages of OS/3.

3.9. APPLICATION PROGRAMS

Sperry Univac offers a wide variety of application programs with the 90/30 Data Processing System. It should be noted that the list of applications that will be made available to the users of this system will become larger than indicated in this section. Indeed, our first concern is to offer realistic continuity to our current users. The applications now available to the user are:

- Project Control
  - Management Control System 90 (MCS 90)
  - Program Evaluation and Review Technique/Critical Path Method 30 (CPM 30)
  - Linear Programming (LP)
- Industry-Oriented Application Programs
  - UNIVAC Industrial System 90 (UNIS 90)
- Printing and Publishing
  - Newspaper Composition Program (NEWSCOMP)
- Banking
  - PROFITS
- Wholesale and Retail
  - Wholesale Inventory Management System (WIMS)
- Conversion Aids
  - COBOL to COBOL (Honeywell H-200 D, NCR Century)
  - Easycoder to COBOL (H-200)
  - Easytran to COBOL (H-200)
  - NEAT/3 to COBOL (NCR Century)
  - NCR Century data files to OS/3 data files

3.9.1. Project Control Application Programs

3.9.1.1. Management Control System 90 (MCS 90)

MCS 90 approaches, in a unique way, the problem of scheduling tasks and costing an allocation of resources within today's complex multiproject contracts. The MCS is broader in content than a PERT/TIME or a CPM application package. It does, however, include these functions. MCS 90 performs resource allocation and uses the results to modify activity schedules, permits alternate description and processing of activity-on-node or activity-on-arrow networks, offers network interfacing and multilevel summarization, is consistent with government directives, and complies with current American National Standards Institute specifications.

MCS 90 offers the following major features:

3.9.1.1.1. Planning and Scheduling

- Extensive input editing
- Data base file organization to aid in the easy retrieval and updating of data
- Processing traditional PERT/CPM networks in activity-on-arrow or activity-on-node notation
- Multiple start and end events or work items
- Alphanumeric, randomly named events and work items
- Assignment of schedule and actual dates to both the start and finish of each work item and to events
- Input of percentage completion to date for in-progress activities
- Incorporation of arbitrary nonworkdays in the calendar
- Assignment of a workweek length, start day, and continuity code to each activity
- Network interaction via interfaces

3.9.1.1.2. Cost Control

- Use and maintenance of resource rate tables
- Cost summarization
- Attachment of resources and cost to activities
- Parallel cost control and reporting based on work breakdown structure (WBS) and organizational accounting structure (OAS)
- Projection of financial plan and resource requirements

3.9.1.1.3. Reports

Extensive sort and inclusion/exclusion parameters allow the user to tailor the reports to meet management requirements.

Upon the user's request the following reports will be furnished:

- Activity and predecessor-successor reports and bar charts
- Event and milestone reports
- Network summarization
- Management-oriented reports
  - Cost summarization
  - Project status
  - Financial plan and status
  - Organization status
  - Manpower loading
- Resource Scheduling
  - Resource status
  - Resource requirements plan

3.9.1.2. Program Evaluation and Review Technique/Critical Path Method 30 (CPM 30)

CPM 30 is an application program that monitors the operation of assigned work within a planned schedule.

In addition to showing the order in which assigned units of work (activities) must occur and how they interrelate, the CPM 30 network processor estimates duration of individual tasks, as well as the entire project. Therefore, schedule slippage, if it is to occur, can be predicted.

CPM 30 is compatible with current needs of users and presently established standards. The program includes:

- input editing
- activity-on-node (precedence relationship) and activity-on-arrow (I-J) processing;
- maximum of 2000 activities in each network;
- variable workweek definition and continuity codes for each activity; and
- activity event, and bar chart reports.

A number of sort codes and inclusion/exclusion parameters are available.

An updateable library of networks is maintained. CPM 30 input is upward compatible to MCS 90.

3.9.1.3. Linear Programming

The OS/3 linear programming (LP) package is a mathematical applications program that optimizes allocation of limited resources (capital, manpower, or raw materials) to obtain a specific objective, such as maximum profit or minimum cost. The object is routinely obtained when the business or industrial system being optimized can be modeled in linear (first degree) equations.

LP analyzes alternate courses of action that are available to the manager. The program accurately determines the effect on profit or cost of variations in available resources, expansions of capacity, or combinations of such changes.

The program operates under a very simple agenda control language and includes:

- Bounded variables
- A composite primal
- Multiple and partial pricing
- Crashing
- Resettable tolerances
- Logging
- Right-hand side and objective function parametrics and ranging
- Revise
- BCD output
- Dump and restart
- Advanced inversion

3.9.2. Industry-Oriented Application Programs

3.9.2.1. UNIVAC Industrial System 90 (UNIS 90)

UNIS 90 is a comprehensive manufacturing system consisting of four major subsystems: master-data processor, inventory management, production planning and scheduling, and work order management.

- Master-Data Processor

  The master-data processor subsystem of UNIS 90 provides maintenance of the standard manufacturing data base, and processing capabilities in the areas of bill of material retrievals and standard routings. Capabilities include:

  — adding, deleting, and changing product-defining records

  — replacing product structure, operation, and tool reference records

  — copying bills of material and standard routings

- deleting bills of material, standard routings, and tool lists
- maintaining single-level, multilevel, indented, and summarized bills of material and where-used lists
- retrieving standard routings and work center where-used and tool where-used lists

■ Inventory Management

The inventory management subsystem of UNIS 90 adds capabilities of: inventory control, statistical forecasting, material requirements planning, order recommendation, order allocation, and ABC analysis and statistics. The master-data processor is required for use of the inventory management subsystem. Capabilities include:

- linear, trend, seasonal, and trend seasonal forecasting, forecasting model analysis, forecast deviation, and tracking signal
- inventory transaction processing, on-hand, on-order reservations, physical count, and transaction costing
- safety stock calculation and order point control
- lot sizing, lead time offsetting, control effectivity, net-change, pegging, and reservations
- order recommendation and release notification
- order monitoring and rescheduling, expediting, and cancellations

■ Production Planning and Scheduling

The production planning and scheduling subsystem of UNIS 90 provides infinite and finite capacity planning and loading, scheduling of networks and single orders, work center load charts, and shop order schedules. Production planning and scheduling can be used as a standalone system. Capabilities include:

- backward and forward scheduling
- splitting and overlapping
- reduction of queue times
- earliest and latest start dates
- earliest and latest end dates
- priority calculations
- alternate routings and work centers

3.9.3. Printing and Publishing

3.9.3.1. Newspaper Composition Program (NEWSCOMP)

NEWSCOMP is an online, real-time newspaper composition system consisting of four subsystems: basic NEWSCOMP, edit control, classified ads, and wire service.

- Basic NEWSCOMP

  The basic NEWSCOMP subsystem is used for production of composed material (justification, hyphenation, organization, and sizes). Following input the text is processed, and output is justified on paper tape or magnetic tape for input to a full range of photocomposing equipment. Basic NEWSCOMP also provides automatic typesetting of textual material. Commands are provided for:

  — delimiters

  — space control

  — face selection

  — format control

  — language control

  — character string manipulation

  — justification modifiers

  — auxiliary control

- Edit Control

  The edit control subsystem adds the following capabilities to basic NEWSCOMP UNISCOPE 100/200 editing:

  — proofread/correction

  — merge takes

  — copy takes

  — insert takes

- Classified Ads

  The classified ads subsystem augments basic NEWSCOMP with the following capabilities:

  — ad extension

  — sort by classification

  — skip dates

  — automatic ad deletion

  — output by classification

- Wire Service

  The wire service subsystem adds the following to the basic NEWSCOMP wire service capabilities:

  — read, strip, and rejustify wire service tape

  — wire service acquisition

3.9.4. Banking

3.9.4.1. PROFITS

PROFITS is a real-time application package designed for financial institutions in the thrift industry. It consists of two major subprograms: loan accounting and time deposit.

Processing modules are written in COBOL as reentrant action programs running under IMS 90. Data base access is controlled by IMS 90. Most teller units and other communication devices can be accommodated by ICAM with specially developed device handlers.

PROFITS includes the following processing capabilities:

- Loan Accounting System
  - mortgages
  - discount loans
  - simple loans
  - student loans
  - construction loans
- Time Deposit System
  - regular savings
  - payment orders
  - certificates of deposit
  - lease security accounts

3.9.5. Wholesale and Retail

3.9.5.1. Wholesale Inventory Management System (WIMS)

WIMS provides a distributor with the information necessary to determine when and how much to order from vendors. The WIMS processes minimize the overall cost of inventory maintenance, purchasing and receiving, freight costs, and loss of discounts. Factors considered are: lead time and variability, forecast demand, service level required for each item, inventory carrying costs, purchasing and receiving costs, discount structures, carload and pallet sizing, item minimum order quantities, and shelf life.

The WIMS system operates in two groups of programs:

1. Initializing and Estimating System
2. Maintenance System

The first group of programs is for the initializing of demand history, determining forecast models and ordering strategy, and is executed when the WIMS system is originally implemented, and thereafter only as required by changing conditions or company objectives.

The second group of programs, which make up the maintenance system, is run on a scheduled basis, daily — weekly — monthly, utilizing the parameters generated by the initializing programs to decide "when and how much" to order for each item in inventory.

The main inventory management functions supported are:

- Forecasting

  A forecast of future demand patterns is the basis upon which all ordering decisions are made. The WIMS system forecasts projections of past demand patterns into the future. For example, when past demand information indicates a strong uptrend and a seasonal pattern, the WIMS system will consider both of these pattern characteristics while forecasting future demand. Forecast modules are developed which handle vendor items exhibiting up or down trends, seasonality demand, or a constant pattern movement.

- Determination of Safety Stock Levels

  To guard against errors in forecasting future demand, it is necessary to provide safety stock. The amount of safety stock required is basically in direct proportion to the forecast error. The WIMS system measures the forecast error so that the correct amount of safety stock can be determined to compensate for error and to provide the desired level of customer service.

- Determination of Order Quantity

  The selection of the order strategy for a vendor line is determined by the WIMS system on the basis of when, and how much to order for each item of inventory, utilizing the most desirable order strategy. Ordering of vendor line items may be combined to effect a quantity discount or freight rate, or the entire vendor line may be ordered independently of other vendor lines. In ordering decisions, consideration is given to lead time, predetermined minimums, shelf life expectations, and pack descriptions.

- Economic Order Allocation

  The WIMS system performs a series of computations on the total cost curve at each discount breakpoint to assure that quantity discounts or freight "breaks" are realized.

3.9.6. Conversion Aids

3.9.6.1. COBOL-to-COBOL Convertors

These convertors will process source language programs (card input format) and translate them into an equivalent OS/3 COBOL source statement. The translated OS/3 COBOL statements must be subsequently compiled to produce an object program.

Data base files are not converted by language translators and this aspect, as well as other systems considerations, such as job control, must be separately considered.

The COBOL-to-COBOL convertor will accept source libraries of either Honeywell H-200 D Level COBOL or NCR Century COBOL. A card parameter will indicate to the convertor which one of the two respective libraries is to be processed. It should be understood that inputted convertor source statement libraries are assumed to be production programs and not newly developed programs that require native mode testing.

3.9.6.2. Easycoder to COBOL and Easytran to COBOL (H-200)

Easycoder and Easytran programs that exploit physical I/O idiosyncrasies of the particular processor, address modification, register-to-register operations, and bit manipulation provide a poor source for COBOL translation because COBOL has no language capability to properly handle these programming techniques.

3.9.6.3. NCR NEAT/3 TO COBOL (NECON)

NECON converts NCR NEAT/3 source programs to the comparable 90/30 system OS/3 Extended American National Standard COBOL source programs. In addition, NECON provides:

- a listing of the input source program and the corresponding output statements;
- a diagnostic listing of input statements that are not translatable or where the provided translation may be doubtful in certain cases; and
- a conventient method for the user to supply the specific translation required in special conditions.

3.9.6.4. Data File Convertor (FICON)

FICON is a file conversion utility program that enables the user to convert cards and data tapes to OS/3 standard format files. The output file from FICON may be punched cards, magnetic tape, or OS/3 library files for program storage. The ability to selectively modify both records and fields within records is provided. Translation and modification of data formats is provided as well as the ability to change the data type of fields.

A variety of tape formats may be processed with emphasis on converting Honeywell 100, Honeywell 200, and NCR Century files.

3.10. DATA BASE MANAGEMENT SYSTEMS

3.10.1. Information Management System 90 (IMS 90)

IMS 90 is an interactive transaction-oriented data management system. The functional capabilities of IMS 90 include:

- acceptance of input from remote terminals identified in a network definition
- transaction-oriented processing
- resource allocation for, and scheduling of, applications programs to process each transaction
- UNIQUE (uniform inquiry/update) — an easily learned and used command language
- transmission of generated output
- maintenance and report generation of analytical statistics
- acceptance of defined files in the data base
- terminal command language to maintain and control the communications network

- recovery and restart procedures

3.10.2. Data Base Management System 90 (DMS 90)

DMS 90 is a collection of system programs that support the development of integrated data bases. These programs provide for the description, initialization, creation, accessing, maintenance, backup, and recovery of data bases. The languages used in the description and manipulation of DMS 90 data bases are derived from the CODASYL data base specifications. A data base may be accessed by batch application programs and communications application programs.

- Description of Data

  The description of data in a DMS 90 data base is entirely separate from the manipulation of that data in application programs. This results in a higher level of data independence for application programs. All description is done in high-level languages comparable to the data declaration language of COBOL. The logical structure and access methods for an entire data base are described in the schema data description language (DDL). The subset of the data base to be referenced by any given application program or group of programs is described in the subschema DDL. Many different subschemas may exist for the same data base, each representing a distinct view of the data. The run-time data base management system (DBMS) restricts the accesses of any application program to only that subset of the data base described in the subschema invoked by the program thereby providing a degree of data base security.

  DMS 90 schema and subschema compilers store descriptions into the data dictionary portion of a data base. The subschema compiler also generates data dictionary object modules that are referenced by the DBMS in order to interpret data base requests from application programs. The data dictionary is referenced by the compilers to generate reports describing the data. Since it is, in all respects, a standard DMS 90 data base, it can also be accessed by programs written by the data base administrator (DBA) to generate specialized data dictionary reports.

- Storage Structures and Access Methods

  A data base is subdivided into areas. All occurrences of a record of a given type must be stored in the same area. Records of different types can be stored in the same area. Each area is assigned a specified number of pages where a page is a physical unit of storage that is some integral multiple of 512 bytes.

  A data base is stored in a direct access method (DAM) file.

  Records can be related in sets in which a record of one type is designated as an owner and records of one or more other types are designated as members. Sets are implemented as record occurrences linked into chains by data base keys (logical direct access pointers). A given record type may be an owner of multiple set types and a number of multiple set types. The set mechanism can be used to relate records in sequential, hierarchical, or network structures.

  The physical placement (location mode) of records can be controlled to optimize performance. The location mode of a given type of record can be specified as direct to allow application programs to dynamically specify where each occurrence is to be stored, calculated to cause records to be stored as a function of a hashing algorithm, or via set to cause each member record occurrence to be stored in or near the page containing the corresponding owner record occurrence in a given set relationship. When a record is stored into the data base a unique logical direct access address called a data base key is assigned to it. This key remains unchanged for the lifetime of the record.

  Records can be retrieved from the data base by one or more of the following access methods: physically sequential retrieval within an area, direct retrieval by data base key, calculated retrieval by key value, chained retrieval by set relationships.

Space within a data base is managed dynamically on a page level. Space within an existing page is allocated to a record when it is stored. When a record is deleted, the allocated space within the page containing the record is compacted with other free space and the free space is immediately available for reallocation. Pages within an area are statically allocated at the time the data base is initialized.

- Manipulation of Data

  The data base can be accessed by means of data manipulation language (DML) statements that are written in the procedure division of a COBOL application program intermixed with conventional COBOL statements. DML statements are comparable in level to the procedural statements of COBOL. COBOL programs containing DML are processed by a DMS 90 preprocessor to produce pure American National Standard COBOL source programs that can be subsequently compiled by the COBOL compiler.

  The DML contains statements that invoke a specified subschema, establish contact with the DBMS, open and close areas of the data base, find data base records and get them into working storage, store new records, modify and delete existing records, insert and remove records from set occurrences, save current information and test for set membership.

- Recovery

  A journal file is generated by the DBMS to support the operation of offline recovery utilities. The offline recovery utilities can perform forward recovery of a data base to its final state or backward recovery to any selected run-unit begin checkpoint.

3.11. DIAGNOSTIC SOFTWARE

The diagnostic library consists of a series of programs that provide testing of both central and subsystem hardware. The library is divided into the following areas:

1. Central complex diagnostics and software maintenance panel
2. Peripheral diagnostic
3. Peripheral prognostic

3.11.1. Central Complex Diagnostics and Software Maintenance Panels

The central complex consists of the processor, main storage, system console, integrated peripheral channels, and the selector and multiplexer channels. The diagnostic library to support the maintenance of these components of the central complex consists of a series of freestanding programs that run independently of the OS/3 supervisor and microdiagnostics. For the integrated peripherals, software is provided to replace the function of the extensive hardware in the maintenance panel. This software utilizes the *SOFTSCOPE* feature, which allows the system console to be used as an oscilloscope to enable the user to monitor, analyze, and produce graphs of timing signals from the integrated subsystems. Information may be displayed at the system console, or hand copy can be obtained at a printer for diagnostic and adjustment purposes.

3.11.2. Peripheral Subsystem Programs

- Subsystem Testing

  The programs provided for testing subsystems are run under control of the OS/3. The operator can run any of the subsystem programs in an offline state dedicated to diagnostic testing and maintenance.

- Program Structure

  The programs have two parts: the control section and the test section. The control section accepts and displays parameters, provides an interface with OS/3, issues requests for I/O, provides error recovery and reporting, calls in test segments as required, and handles all other housekeeping functions. The test section consists of several tests that reside in one or more overlay segments. Each test exercises some particular function or group of functions of the subsystem.

- Test Structure

  The tests for a given subsystem use a building block approach, starting the basic functions and extending to the more complex functions that can be performed. Testing is divided into three parts:

  1. The control unit

  2. Subsystem handler control functions

  3. Data recording and reading

  In addition, there are tests unique to given subsystems, such as interchangeability, compatibility for magnetic tape and disc drives, and loop control and rewind for magnetic tape drives.

- Communications Subsystem Program

  The library of programs for communication testing is designed to support all Sperry Univac products that may be a part of the 90/30 system.

3.11.3. Subsystem Prognostic Maintenance

The unique *SOFTSCOPE* feature in the 90/30 system provides extensive prognostic capability for the integrated peripherals.

Prognostic maintenance is based on the premise that the majority of electromechanical failures result from gradual changes over an extended period of time. Prognostic maintenance uses analytical routines to allow a computer system to detect, analyze, and record on hard copy the performance of its electromechanical devices. These analytical routines exercise the system hardware and include the predefined parameters required to display, on hard copy, the characteristics profile of each unit's operation. The routines are used with special test devices, such as a pattern of MYLAR* punched cards, which are used to test the performance and drift of electromechanical devices.

**Trademark of E.I. du Pont de Nemours & Co.*

4. 90/30 System Hardware Summary

4.1. GENERAL

The 90/30 system hardware components consist of four types of equipment based upon functions within the system: these types are processor, main storage, input/output (I/O) control, and peripheral equipment. Each equipment type relies on advanced design features to provide maximum utility and efficiency in its particular function. In keeping with an established expandable system concept, each component is compatible with similar units or with units having similar functions in the system. In this way, a large selection of peripheral subsystems and different storage capacities can be configured with the processor to make the system responsive to a wide range of business applications.

Information, in summary form, concerning the salient features of the system hardware is presented in this section. Information regarding the system's characteristics, such as speed and capacity, is presented in Section 5. Also, additional information about the processor and peripheral subsystems may be found in manuals written specifically for these components.

4.2. 90/30 SYSTEM PROCESSOR

General business and economic conditions have given rise to the need for a processing system which offers high performance at a low cost. The 90/30 processor is the nucleus of a low-cost, disc-oriented system capable of solving a wide range of data processing problems. For example, it can perform random, sequential batch, communications, scientific, or inquiry/response information processing.

The processor includes an arithmetic section, main storage section, and an input/output section. The basis of control for its major function — arithmetic operations — is microprogrammed instructions. Each instruction is resident in a separate control storage area and is made up of a set of microcodes that manipulates the logical circuitry of the system to perform the specified operation.

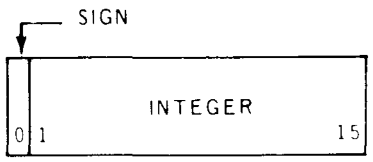
4.2.1. Arithmetic Section

The arithmetic section performs all logical operations, arithmetic operations, data comparisons, and shifting. Fixing-point binary arithmetic uses twos complement number representation. Floating-point and decimal arithmetic use signed, absolute magnitude number representation.

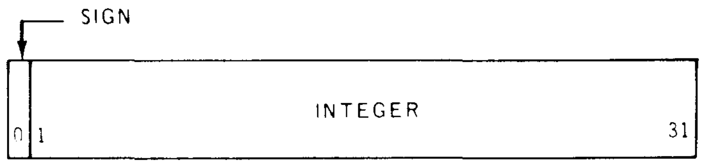
4.2.1.1. Fixed-Point Arithmetic

Fixed-point numbers have a fixed-length format comprised of a sign bit followed by an integer field. When the sign bit is 1, the integer represents a negative value; when the sign bit is 0, the integer represents a positive value.

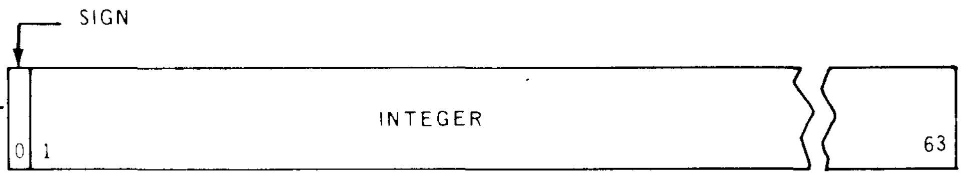
HALF-WORD FORMAT



FULL-WORD FORMAT



DOUBLE-WORD FORMAT



When held in one of the 32 general registers, a fixed-point number is generally treated as a 32-bit operand. When a half-word (16-bit), fixed-point number is called from main storage and loaded into a register, the sign is extended to the left to fill the full-word (32-bit) register. The contents of the register are then handled as a full-word operand in fixed-point arithmetic operations.

Certain operations use a double-word (64-bit) operand comprising one sign bit followed by a 63-bit integer field. The 64-bit operand is located in two adjacent general registers and is addressed by an even-numbered address.

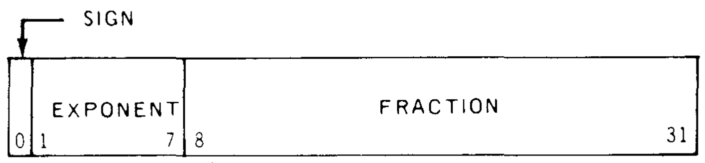
When fixed-point data is located in main storage, it may be stored as a half word, full word, or double word. This data must be located on the integral storage boundary of its associated format.

4.2.1.2. Floating-Point Arithmetic

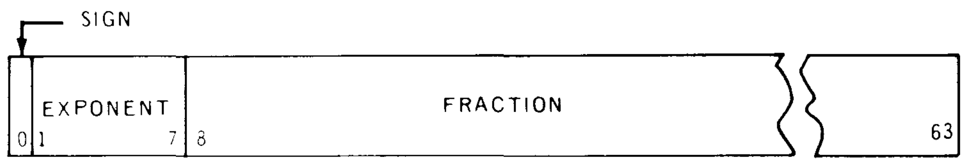
Floating-point arithmetic is an optional feature (micrologic expansion). A floating-point number comprises a biased exponent (characteristic) and a signed fraction (mantissa). The biased exponent is expressed in excess-64 binary notation; the fraction is expressed as a hexadecimal number having a radix point to the left of the high-order digit. The quantity expressed by the full floating-point number is the product of the fraction and the number 16 raised to the power minus 64 of the biased exponent.

Floating-point numbers are either a full word (short format) or a double word (long format) in length. Both formats can be used in main storage or in the floating-point registers.

SHORT FORMAT



LONG FORMAT



The floating-point instruction set provides for loading, adding, subtracting, comparing, multiplying, dividing, storing, and sign control of short- or long-format operands. Short-format operands provide faster processing and require less storage space than long-format operands. Long-format operands provide greater precision in computation.

To increase the precision of certain short- and long-format operand computations, an additional low-order digit (guard digit) is carried within the hardware on the intermediate result of the following operations: add normalized, subtract normalized, add unnormalized, subtract unnormalized, compare, halve, and multiply.

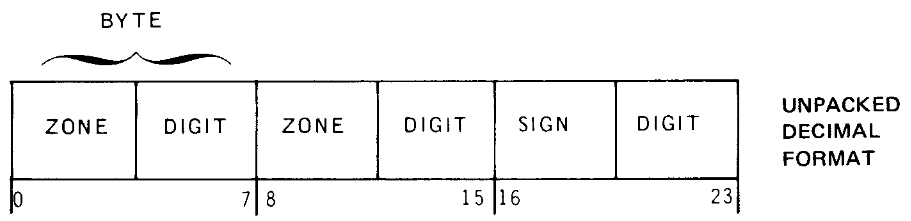
A normalized floating-point number has a nonzero, high-order, hexadecimal fraction digit and is the most significant representation of a given quantity. The process of normalization consists of shifting the fraction to the left until the high-order hexadecimal digit is nonzero and reducing the exponent by the number of shifts. A fraction with one or more high-order digits of 1 is unnormalized.

Unnormalized number:	$.0093 \times 10^2$
Normalized number:	$.93 \times 10^0$

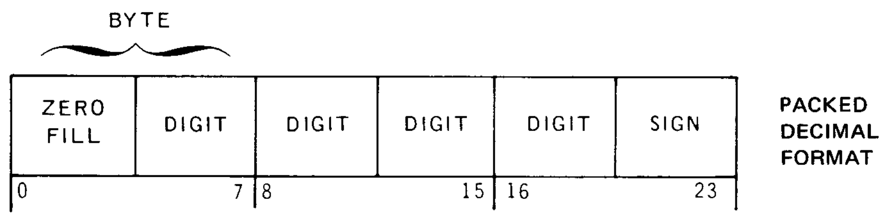
4.2.1.3. Decimal Arithmetic

Decimal number fields can be variable in length and can be in two formats: unpacked decimal and packed decimal. Instructions are provided for converting decimal numbers from unpacked to packed and from packed to unpacked formats.

In the unpacked decimal format, each byte contains one digit of a multidigit number. The byte is divided into two equal fields, a zone field and a digit field. A zone value is represented in the most significant four bits, and the digit is represented in the least significant four bits. The zone field is 1111_2 when operations are performed in EBCDIC mode and 0101_2 when in ASCII mode. The zone portion of the least significant byte specifies the sign of the number. Except for certain I/O devices, such as the printer, all decimal operations must be performed by using packed format. The format of a 3-digit unpacked number is:



In the packed format, each byte contains two digits. The least significant four bits of the least significant byte provide the sign of the number. The format of a 4-digit packed number is:



The binary values 1011 and 1101 represent a minus sign, and the binary values 1010, 1100, 1110, and 1111 represent a plus sign. This assignment of sign codes permits the use of either of two conventions: American National Standard Code for Information Interchange (ASCII) modified to eight bits or Extended Binary Coded Decimal Interchange Code (EBCDIC). A control bit in the program status word determines whether the system is to handle the ASCII or the EBCDIC code.

4.2.1.4. Logical Operations

Logical operations consist of comparing, translating, editing, bit testing, and bit manipulation and shifting. Logical information is processed as fixed- or variable-length data. The fixed-length data, consisting of one or four bytes, is processed in the general registers. In some operations, only the low-order eight bits of a register are used and the remaining 24 bits are left unchanged.

In storage-to-register operations, the main storage data can occupy a full word, a half word, or a single byte. If a full word is addressed, it must be located on a word boundary; if a half word is addressed, it must be on a word or half-word boundary.

4.3. INSTRUCTION REPERTOIRE

The basic instruction repertoire for the 90/30 system consists of 84 instructions. Sixty-four instructions can be added as part of the micrologic expansion feature. This feature includes 44 floating-point instructions and 20 instructions of various types to increase the power of the 90/30 system. The storage protection feature adds two additional instructions.

There are five different instruction formats used in the 90/30 processor:

RR	(register to register)
RX	(register to indexed storage)
RS	(register to storage)
SI	(storage and immediate operand)
SS	(storage to storage)

Each format consists of an operation code and two or more fields which specify the addresses of the operands.

4.3.1. Instruction Types

Instructions can be two, four, or six bytes in length. All instructions must start on an even address. The object formats of the five instruction types are shown in Figure 4—1. The symbols used in Figure 4—1 are explained in Table 4—1.

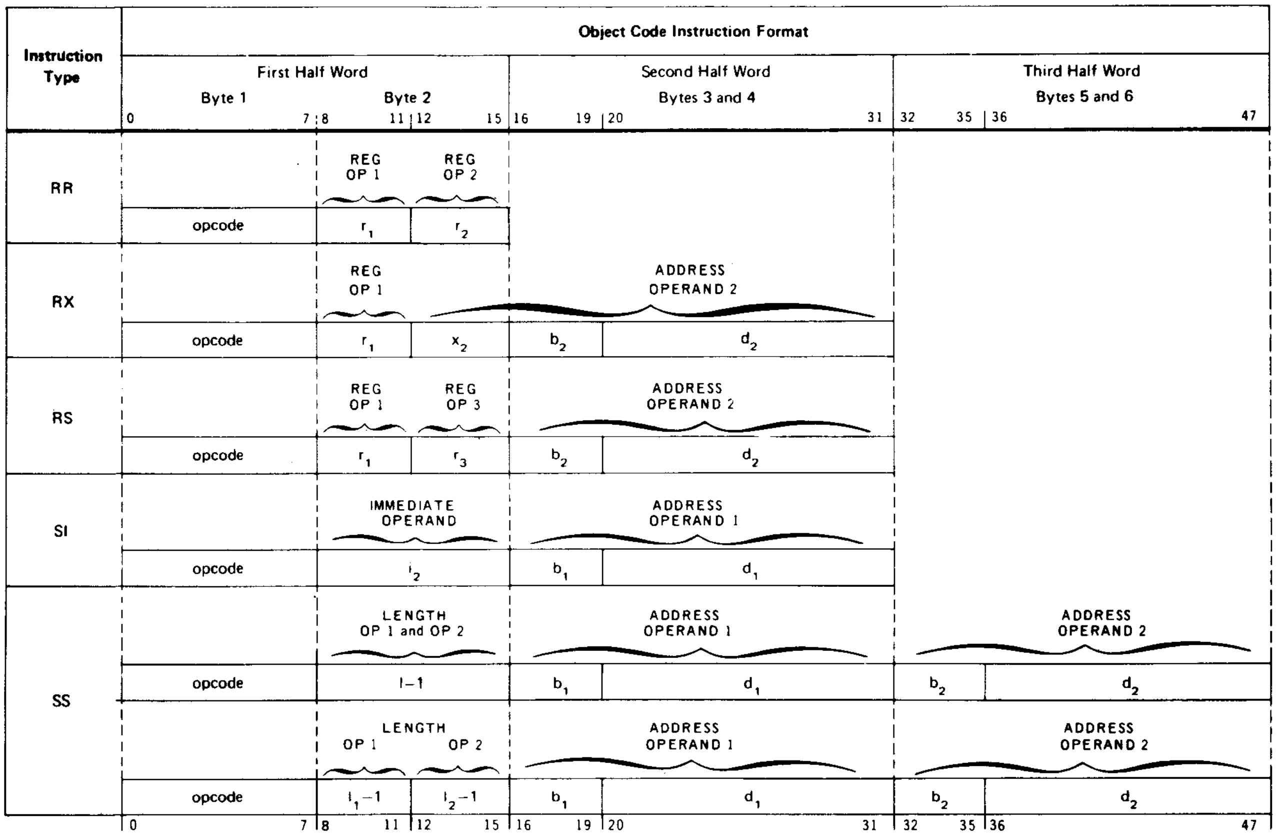


Figure 4—1. Basic Instruction Formats (Object Code Form)

Table 4—1. Symbols Used to Describe Operand Formats

Symbol	Meaning
opcode	Instruction operation code
r_1	Number of the register addressed as operand 1, a mask, or a register which is the first register of a multiregister group
r_2	Number of the register addressed as operand 2
r_3	An expression representing a register which is the last register in a multiregister group, an increment, an operand address, or a control storage address
x_2	Number of the register to be used as an index for operand 2 of an RX instruction
i_2	Immediate data used as operand 2 of an SI instruction
l	Length of operands 1 and 2 as stated in source code*
l_1	Length of operand 1 as stated in source code*
l_2	Length of operand 2 as stated in source code*
b_1	Base register for operand 1
b_2	Base register for operand 2
d_1	Displacement for operand 1
d_2	Displacement for operand 2
OP1	Operand 1
OP2	Operand 2
OP3	Operand 3

*This is coded as the true source code length of the operand, not the length less 1, as required in object code. The assembler makes a reduction of 1 in the length when converting source to object code.

4.3.1.1. Register-to-Register (RR) Instructions

The RR instructions are used to process data contained in registers. The maximum length of the data that can be handled is a double word in floating-point registers and a full word in general registers. The data can be a signed or unsigned binary number, a short- or long-format floating-point number, or a decimal number, depending on the specified operation. Operand 1 specifies either a register or a mask. Operand 2 specifies a register.

Some RR instructions use both operands as an immediate data operand.

4.3.1.2. Register-to-Indexed-Storage (RX) Instructions

The RX instructions are used to process data between registers and indexed storage. A double word is the maximum length of data that can be handled. The data can be a signed or unsigned binary number, a short- or long-format floating-point number, or a decimal number. Operand 1 specifies a register or a mask. Operand 2 specifies a main storage location.

4.3.1.3. Register-to-Storage (RS) Instructions

The RS instructions are used to perform multiple register and storage operations, as well as data shifting. The first and third operands specify the numbers of two general registers or the boundaries for general multiregister usage. Operand 2 specifies a main storage location.

4.3.1.4. Storage-and-Immediate-Operand (SI) Instructions

The SI instructions are used to perform operations on an 8-bit value, called immediate data, and an operand in main storage. Operand 2 specifies the immediate data or mask. Operand 1 specifies a 1-byte or half-word main storage location.

4.3.1.5. Storage-to-Storage (SS) Instructions

The SS instructions are used to perform operations on two operands located in main storage. In logical operations, the operands are assumed to be equal in length and can vary from 1 to 256 bytes. In decimal operations, the operands can be of different lengths and can vary from 1 to 16 bytes.

4.3.2. Nonprivileged Instructions Set

The nonprivileged instructions are used to process fixed-length binary numbers, floating-point numbers, packed and unpacked decimal numbers, and EBCDIC or ASCII characters. Data can be transferred between main storage and the user program set of general registers and from one location in main storage to another main storage location. The operations of shifting, branching, and logical functions also are included. All nonprivileged instructions are listed in Appendix A.

4.3.3. Privileged Instructions Set

The privileged instructions set is used by the software operating system when operating in the supervisory state. In this state, all installed instructions are valid and can be executed. This set of instructions includes the facilities to load and store the contents of low-order main storage and to load the writable section of the microprogram control storage. Instructions in the privileged set cannot be executed in a user program. The privileged instructions are listed in Appendix A.

4.4. SYSTEM CONSOLE

The system console is an input/output device for directing and monitoring the operation of the system. The system console is in a centralized location for initial load control, run/stop control, and system status monitoring. The system console consists of a keyboard and visual display unit, switches, and indicators. Communication with the system is through the integrated peripheral channel. A system console printer is available as an option to provide a hard copy of the system console displays.

4.5. MAIN STORAGE

The main storage component of the system consists of semiconductor main storage developed for the Series 90 Processors. Main storage operates in a 600-nanosecond cycle time per two bytes.

Minimum storage size is 32,768 bytes and can be expanded to any of the following sizes:

49,152 bytes

65,536 bytes

98,304 bytes

131,072 bytes

163,840 bytes

196,608 bytes

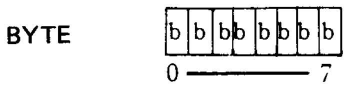
229,376 bytes

262,144 bytes

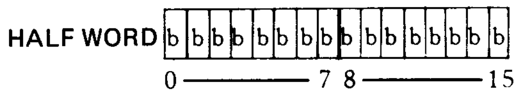
Address and data parity checking are provided.

4.5.1. Information Positioning

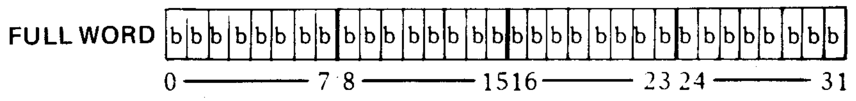
Locations in main storage are addressed consecutively from 0 through a maximum of 262,143. Bytes may be accessed separately or in groups. A group of bytes is addressed by the leftmost byte of the group. The bits in a byte are numbered from left to right starting with 0.



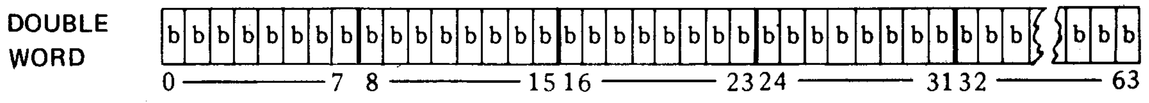
Half-word formats consist of two consecutive bytes.



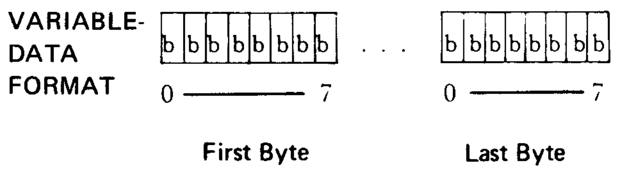
Full-word formats consist of four consecutive bytes.



Double-word formats consist of eight consecutive bytes.



Variable formats consist of a variable number of consecutive bytes.



4.5.2. Low-Order Main Storage

The low-order 640 bytes of main storage are reserved for specific operating information. This information is accessed by the hardware and the supervisor during execution of appropriate functions. The supervisor provides for loading and protecting the information in these locations.

4.5.3. Storage Protection

Program integrity in a multijobbing environment is guaranteed by the optional storage protect feature. Eight keys are provided for this purpose. Each 512-byte segment of main storage (1024 bytes in configurations larger than 131K bytes) may be protected on a write or read/write basis.

4.6. INPUT/OUTPUT SECTION

The input/output (I/O) section of the system initiates, directs and monitors the transfer of data between main storage and the subsystems. After an I/O instruction is initiated, the data transfer is performed concurrently with other processor functions. All I/O channels and the processor operate concurrently.

A full line of equipment is available and supported by OS/3 software. These subsystems are described with the I/O channels provided for their connection. The following is a list of SPERRY UNIVAC 90/30 System I/O channels and related subsystems.

- Communications Adapter

  UNISCOPE 100/200 Display Terminal
  DCT 500/524 Data Communications Terminal
  DCT 1000 Data Communications Terminal
  DCT 2000 Data Communications Terminal
  1004/1005 Card Processor System
  IBM Binary Synchronous Communications Device
  Commercial communications teletypewriters

- Integrated Peripheral Channel

  System Console/System Console Printer
  0717 Card Reader Subsystem
  0605 Card Punch Subsystem
  0773 Printer Subsystem
  Communications Adapter

- Integrated Disc Adapter

  8416 Disc Subsystem*
  8418 Disc Subsystem

- Selector Channel

  UNISERVO 10 Magnetic Tape Subsystem
  UNISERVO 12 Magnetic Tape Subsystem
  UNISERVO 14 Magnetic Tape Subsystem
  UNISERVO 16 Magnetic Tape Subsystem
  UNISERVO 20 Magnetic Tape Subsystem
  8411 Disc Subsystem*
  8414 Disc Subsystem*
  8430 Disc Subsystem

- Multiplexer Channel

  0716 Card Reader Subsystem
  0604 Card Punch Subsystem
  0770 Printer Subsystem
  UNISERVO VI-C Magnetic Tape Subsystem*
  UNISERVO 10 Magnetic Tape Subsystem
  0768 Printer Subsystem*
  0920 Paper Tape Subsystem
  2703 Optical Document Reader (Physical level support only)
  9000 Series Channel Adapter

Subject to availability

4.7. COMMUNICATIONS ADAPTER

The 90/30 communications adapter (CA) is mounted in the console stand. The CA terminates up to 6 full-duplex or 12 half-duplex communications lines. With an expansion feature, up to 12 full-duplex or 24 half-duplex communications lines are available. It provides the proper sequencing of the communication lines when either the IPC issues an SIO connect signal to the CA or a device issues a service request signal to the CA. Data transfer sequences include data input, data output, and status.

The CA controls message discipline for either 6 full-duplex or 12 half-duplex communications lines. The CA contains the CA interface, the communications multiplexer module (CMM) and the line adapters (LAs). The CMM performs the functions required for message discipline. The available LAs accommodate various synchronous, asynchronous, wideband, auto-dial, asynchronous relay, TWX, and TELEX communication lines. The interfaces include RS-232, wideband (TELEX), MIL STD 188C, and telegraph (contact closure).

4.7.1. UNISCOPE 100/200 Display Terminal

The UNISCOPE 100/200 Display Terminal is a 2-way remote terminal device that is keyboard-operated and contains a cathode-ray tube for message display. The keyboard includes typewriter, cursor control, and editing function keys. The cathode-ray tube displays the system output messages and displays operator input messages for composition and editing before being transmitted to the system. Each character entered by the operator is immediately displayed and stored in the control unit of the display terminal. The characters are generated by a read-only storage digital stroke generator, which produces large, clear characters.

The terminal provides input/output message buffering, refresh storage, character generation, and control logic. Special interfaces for hardcopy output are available. A variety of presentation formats provides display capacities of 960 or 1024 characters for the UNISCOPE 100 Display Terminal or 1536 or 1920 characters for the UNISCOPE 200 Display Terminal. The terminal can be used as a data entry device or as a display device.

4.7.2. DCT 500/524 Data Communications Terminal

The 500/524 Data Communications Terminal is an unbuffered, asynchronous keyboard and print device, similar in operation to a teletypewriter, providing up to 132 print positions. The terminal can replace existing teletypewriters with little or no change in operation routine. It communicates with the central processor over voice-grade telephone lines. In a multiparty polled environment, the terminal operates in accordance with ASCII procedures. The terminal can operate in a receive- only mode, a keyboard send/receive mode, or an automatic send/receive mode. The basic minimum printer system can be expanded to include a keyboard and a 1-inch paper tape read/punch unit at any time. The DCT 524 also has a tape cassette capability.

4.7.3. DCT 1000 Data Communications Terminal

The DCT 1000 Data Commuunications Terminal is a fully buffered, 30-character per second incremental printer that can be expanded to include a keyboard, card reader, card punch, paper tape reader/punch, and an auxiliary printer. The terminal transmits or receives data in conversational or batch mode. Two 160-character buffers facilitate:

- automatic blocking for eliminating complicated and time-consuming functions and minimizing training;
- automatic error correction for eliminating manual correction procedures, such as reloading cards and retyping input data;

- error-free output because all messages are completely checked for character errors, block errors, duplicate blocks, or lost blocks, with the result that no errors are entered into the output; and

- high transmission speed, which enables full utilization of line capacity because the transmission rate can be much higher than the I/O rate. On party line systems, this yields data throughput on a line as the sum of the throughputs of the individual terminals.

The terminal also has complete polling and address recognition capabilities, which permit the processor to completely control up to 31 DCT 1000 terminals on a single line. The terminals may be connected in a series string in different geographical locations or at a single point on the terminal's multiplexer interface.

4.7.4. DCT 2000 Data Communications Terminal

The DCT 2000 Data Communications Terminal is a combination printer and reader/punch that is capable of transferring quantities of data efficiently in batch mode over voice-grade communications lines. It is also available for use as a receive-only printer terminal without the combination card reader/punch.

No programming is required at the terminal location, which makes the terminal relatively simple to install and operate. Either a private line connection at a maximum rate of 2400 bits per second or a dial facility at a maximum rate of 2000 bits per second can be installed according to the user's requirements since the terminal meets the EIA RS232-C standard communications interface for industry.

4.7.5. 1004/1005 Card Processor System

The 1004/1005 card processor system consists of a card reader, a processor, and a printer housed in a single unit. It performs the functions of card reading, data processing, printing, and punching. The card processor edits and accumulates totals from data punched in 80-column cards and prints the results in any desired format. It has the ability to perform arithmetic, transfer, and compare operations and contains a reliable fast-access magnetic core storage with a cycle time of 8 microseconds. Data is stored in the form of 6-bit characters. The major difference between the 1004 and 1005 card processor systems is that the 1005 card processor system contains an electronic program module which is internally programmed. Both card processors are available in three models, with the models differing in processing and I/O speeds. Model III also includes magnetic tape storage.

4.8. INTEGRATED PERIPHERAL CHANNEL

The integrated peripheral channel (IPC) is an I/O channel and an integral part of the processor. The IPC coordinates all of the information transferred between main storage and its attached unit record subsystems. Communications subsystems may also be attached to the IPC through the CA (see 4.7).

4.8.1. System Console

The system console is a modified UNISCOPE 100 Display Terminal which accepts data from the keyboard of the console control unit, displays the data, and transfers the data via the IPC. It provides a fast, accurate method of operator communication with the system. Messages for the operator are displayed until the screen is full and the earliest messages not requiring a reply are deleted. The operator may optionally recall deleted messages from a message log maintained on disc storage. Provision is made to print the contents of the message log on the high-speed printer for historical purposes.

Data entered into the console buffer is displayed on the screen in a 64-character per line by 16 line format providing a total display of 1024 characters. Displayable characters consist of the 64-character (including space) ASCII set plus control characters.

The protected format capability provides a means of protecting selected data from operator alteration. The format or data on any position of the screen may be protected while certain unprotected areas are left for operator data entry.

Operator control of data is provided by the keyboard and various switches and indicators located above the keyboard or on the console operator panel.

A console printer can be added by attaching a communications output printer (COP). Under processor or operator control, the data displayed on the screen can be transferred to the COP and printed at the rate of 30 characters per second.

4.8.2. 0717 Card Reader Subsystem

The 0717 card reader subsystem provides online input to the processor. It is an integrated unit that reads standard 80-column punched cards at a rate of 500 cards per minute.

The cards are read on a column-by-column basis and are subject to a standard read check feature that ensures correct input. Information in cards punched in extended card code is transferred to main storage in EBCDIC (translate mode). Cards punched in column binary format can also be read and information transferred to main storage in image mode (no translation).

The basic subsystem contains a single input hopper with a capacity of 2400 eighty-column cards and a single output stacker with a capacity of 2000 cards.

4.8.3. 0605 Card Punch Subsystem

The 0605 card punch subsystem punches standard 80-column punched cards. The punch mechanism is an 80-column serial punch that feeds and punches cards row by row at the rate of 75 to 160 cards per minute (cpm), the punch rate being dependent on the number of columns punched in each card. After a card is punched, it is checked to ensure that it was correctly punched. An incorrectly punched card is rejected and the next blank card is repunched under program control. Cards can be punched in either extended card code (EBCDIC translation) or column binary code (image mode).

A read station is available as an optional feature, enabling read or read/punch operations.

4.8.4. 0773 Printer Subsystem

The 0773 printer subsystem includes the print mechanism, actuator drive electronics, actuator drive power supply, paper feed control, interface electronics (buffers), power-up control, and an operator panel.

The printer prints 120 columns of data at specified throughput rates dependent on the number of lines spaced/skipped and the number of characters in the type array (500 LPM, 48 characters; 400 LPM, 64 characters). An optional feature is available that prints 132 columns of data on 144 print positions. The printer is the conventional impact type; that is, print hammers impact the rear of a forms pack (up to six parts thick) and force the forms pack into contact with an inked ribbon and a moving type media. Printing occurs asynchronously (not dependent on initial position of moving type media) when a selected character lines up with a selected column position. The type media consists of a flexible, horizontal-moving metallic band containing etched characters. This type band is mounted in an interchangeable swing-out cartridge assembly for operator convenience and accessibility. See Table 5—6 for printer specifications.

4.9. INTEGRATED DISC ADAPTER

The integrated disc adapter (IDA) is a combination of a channel and a control unit designed to operate with the 8416 or 8418 disc subsystems. The IDA is capable of storing or retrieving information on a removable disc pack mounted on an attached disc drive. The IDA has the ability to search for specific disc information before transferring data to main storage. It also provides the controls necessary to perform accessor movement under program control so that information on any concentric cylinders on the disc surface is available.

The IDA interfaces both the processor and disc drive units, which provides from up to 28.9 million bytes of storage for a single device to a total of 463.3 million bytes of online storage for the maximum configuration of eight disc units. The transfer rate for these devices is nominally 625k bytes per second. The IDA receives or transmits data at the processor interface in half-word parallel form, while it sends or receives information at the disc drive interface in bit-serial fashion. At a given time, data is transferred in only one direction. Checking is performed within the IDA with odd parity employed and carried thoughout the IDA. Error correction codes are used after each field on the disc surface.

4.9.1. 8416 Disc Subsystem*

The 8416 disc subsystem comprises self-contained disc drives with removable, interchangeable disc packs that permit online and offline storage. Each removable disc pack, which consists of four discs (eight recording surfaces) and two protective discs (no recording surfaces) 14 inches in diameter, has a maximum net storage capacity of over 28.9 million bytes. The disc pack mounts on a vertical shaft that rotates at a speed of 2800 rpm. Each recording surface of the removable disc pack contains 411 recording tracks. Each track has track error flagging capability, which (under program control) permits accessing one of the assigned spare tracks in case an addressed track is flagged defective. Up to eight disc units may be attached to the IDA.

4.9.2. 8418 Disc Subsystem

The 8418 disc subsystem comprises self-contained disc drives with removable, interchangeable disc packs that permit online and offline storage. Each removable disc pack, which consists of four discs (eight recording surfaces) and two protective discs (no recording surfaces) 14 inches in diameter, has a maximum net storage capacity of either 28.9 or 57.9 million bytes depending on the model being used. The disc pack mounts on a vertical shaft that rotates at a speed of 2800 rpm. Each recording surface of the removable disc pack contains 815 recording tracks. Each track has track error flagging capability, which (under program control) permits accessing one of the assigned spare tracks in case an addressed track is flagged defective. Up to eight disc units may be attached to the IDA.

4.10. SELECTOR CHANNEL

The selector channel provides input and output capability between the processor and up to eight high-speed peripheral subsystems. The selector channel operates in burst mode only; that is, one of the eight possible subsystems retains control of the interface for the duration of the I/O operation. The processor initiates all I/O operations by issuing I/O instructions to a selected channel and a selected subsystem connected to the channel. Detailed control of the required I/O operation is provided to the channel in software-generated control words. Once the operation is successfully initiated, the channel maintains control of data transfer between main storage and the subsystem independently of the processor. Upon completion of the I/O operation, the state of the channel and subsystem is presented to the software by way of the appropriate status words by means of the I/O status tables.

**Subject to availability*

Devices which can be attached to the selector channel are:

- UNISERVO 10 Magnetic Tape Subsystem
- UNISERVO 12 Magnetic Tape Subsystem
- UNISERVO 14 Magnetic Tape Subsystem
- UNISERVO 16 Magnetic Tape Subsystem
- UNISERVO 20 Magnetic Tape Subsystem
- 8411 Disc Subsystem*
- 8414 Disc Subsystem*
- 8430 Disc Subsystem

4.10.1. UNISERVO 10 Magnetic Tape Subsystem

The UNISERVO 10 Magnetic Tape Subsystem is a freestanding, self-contained unit which provides an external storage medium capable of reading from (forward or backward) or writing to (forward) a magnetic tape. The subsystem consists of a control unit and cabinet which has space to mount the two required tape units. A maximum of eight tape units, in increments of one or two units (beyond the two required) can be attached to the control through the use of up to three auxiliary cabinets.

The UNISERVO 10 Magnetic Tape Subsystem is available in either 9-track or 7-track models. The 9-track units operate in the phase-encoded mode at 1600 bits per inch, and, if the optional dual density feature is added, in the NRZI mode at 800 bits per inch. The 7-track units operate in the NRZI mode at recording densities of 800, 556, or 200 bits per inch, depending on the density selected.

Physical tape speed is 25 inches per second. This results in a 9-track transfer rate of 40,000 frames per second in the phase-encoded mode and 20,000 frames per second in the NRZI mode. The 7-track NRZI transfer rate is 20,000; 13,900; or 5,000 frames per second, depending on the density selected.

The dual channel feature permits nonsimultaneous operation on two channels of one processor or on one channel on each of two processors. Simultaneous operation is not a feature of the UNISERVO 10 Magnetic Tape Subsystem.

4.10.2. UNISERVO 12 Magnetic Tape Subsystem

The UNISERVO 12 Magnetic Tape Subsystem is a freestanding, self-contained unit which provides an external storage medium capable of reading from (forward or backward) or writing to (forward) a magnetic tape. The subsystem consists of a control unit and from 1 to 16 magnetic tape units.

The UNISERVO 12 Magnetic Tape Subsystem is available in either 9-track or 7-track models. The 9-track tape units read and write data in the phase-encoded mode at a density of 1600 bits per inch. If the optional dual density feature is added, the 9-track unit can also write in NRZI mode at a density rate of 800 bits per inch. The 7-track UNISERVO 12 Magnetic Tape Unit reads and writes in NRZI mode only. These units can be programmed to read and write data at densities of 200, 556, or 800 bits per inch. The physical tape speed is 42.7 inches per second, for a 9-track tape transfer rate of 68,320 frames per second in the phase-encoded mode or 34,160 frames per second in NRZI mode. The 7-track tape unit transfers data in NRZI mode at 34,160; 23,740; or 8540 frames per second, depending upon the density selected.

**Subject to availability*

The addition of a second control unit and appropriate features in the associated tape units permit simultaneous write/read and read/read operation on the UNISERVO 12 tape units. Data validity-checking facilities include longitudinal redundancy check for 7-track and 9-track NRZI tapes and cyclic redundancy check for 9-track NRZI tapes.

4.10.3. UNISERVO 14 Magnetic Tape Subsystem

The UNISERVO 14 Magnetic Tape Subsystem is a freestanding, self-contained unit which provides an external storage medium capable of reading from (forward or backward) or writing to (forward) a magnetic tape. The subsystem consists of a control unit and cabinet which has space to mount the two required tape units. A maximum of eight tape units, in increments of one or two units (beyond the two required) can be attached to the control through the use of up to three auxiliary cabinets.

The UNISERVO 14 Magnetic Tape Subsystem is available in either 9-track or 7-track models. The 9-track units operate in the phase-encoded mode at 1600 bits per inch, and, if the optional dual density feature is added, in the NRZI mode at 800 bits per inch. The 7-track units operate in the NRZI mode at recording densities of 800, 556, or 200 bits per inch, depending on the density selected.

Physical tape speed is 60 inches per second. This results in a 9-track transfer rate of 96,000 frames per second in the phase-encoded mode and 48,000 frames per second in the NRZI mode. The 7-track NRZI transfer rate is 48,000; 33,360; or 12,000 frames per second, depending on the density selected.

The dual channel feature permits nonsimultaneous operation on two channels of one processor or on one channel on each of two processors. (See 4.11.10.) Simultaneous operation is not a feature of the UNISERVO 14 Magnetic Tape Subsystem.

4.10.4. UNISERVO 16 Magnetic Tape Subsystem

The UNISERVO 16 Magnetic Tape Subsystem is a freestanding, self-contained unit which provides an external storage medium capable of reading from (forward or backward) or writing to (forward) a magnetic tape. The subsystem consists of a control unit and from 1 to 16 magnetic tape units.

The UNISERVO 16 Magnetic Tape Subsystem is available in either 9-track or 7-track models. The 9-track tape units read and write in phase-encoded mode at 1600 bits per inch. If the optional dual density feature is added, the unit can also write data in NRZI mode at 800 bits per inch. The 7-track model reads and records data in NRZI mode only at a density of 800, 556, or 200 bits per inch, depending on the density selected. Data validity-checking facilities include longitudinal redundancy check for 7-track and 9-track NRZI tapes and cylic redundancy check for 9-track NRZI tapes.

Physical tape speed is 120 inches per second, with the 9-track tape unit having a maximum transfer rate of 192,000 frames per second in the phase-encoded mode or 96,000 frames per second when the NRZI mode is used. Seven-track tape has a data transfer rate of 96,000; 66,720; or 24,000 frames per second, depending on the density selected.

4.10.5. UNISERVO 20 Magnetic Tape Subsystem

The UNISERVO 20 Magnetic Tape Subsystem is a freestanding, self-contained unit which provides an external storage medium capable of reading from (forward or backward) or writing to (forward) a magnetic tape. The subsystem consists of a control unit and from 1 to 16 magnetic tape units. A second control unit may be used to achieve simultaneous operation when the simultaneity option is added to the tape units. The UNISERVO 20 Magnetic Tape Unit is available in 9-track only and reads and writes data in phase-encoded mode at 1600 bits per inch. Physical tape speed is 200 inches per second, with a maximum transfer rate of 320,000 frames per second. The UNISERVO 30 Magnetic Tape Unit provides features such as a power window, automatic tape threading, and wraparound tape cartridge.

The simultaneous dual access configuration provides for read/write, read/read, write/read, and write/write operation on any two individual UNISERVO 20 Magnetic Tape Control Units. In addition to doubling the performance of the subsystem, complete redundancy is achieved by virtue of individual power supplies for each control unit and independent access paths to each UNISERVO 20 Magnetic Tape Unit.

4.10.6. 8411 Disc Subsystem

The 8411 disc subsystem consists of a control unit and from two to eight disc units. The subsystem is a disc storage device that provides random access or sequential access to large data files. The subsystem uses a 6-disc interchangeable magnetic disc pack capable of storing data on 10 inside disc surfaces, for a total storage capacity of 7.25 million 8-bit bytes. The disc pack is loaded onto the drive spindle from the top of the unit by raising the top cover.

Within the disc unit, 10 read/write heads are mounted on a single accessor mechanism that moves the heads in unison between the periphery and the central area of each disc. The access mechanism can assume any one of 203 positions across each disc surface; this simultaneous head movement on the 10 disc surfaces permits 200 addressable data recording cylinders in the disc pack, three cylinders being reserved for replacement tracks. Each cylinder, therefore, contains 10 tracks numbered 0 through 9. Individual tracks are addressed by cylinder number (000 to 202) and by read/write head number (0 to 9).

4.10.7. 8414 Disc Subsystem

The 8414 disc subsystem consists of a control unit and from two to eight disc units. The subsystem is a disc storage device that provides random access or sequential access to large data files. The subsystem uses an 11-disc interchangeable magnetic disc pack capable of storing data on 20 inside disc surfaces, for a total storage capacity of 29,176 million 8-bit bytes. The disc pack is loaded onto the drive spindle from the top of the unit by raising the top cover.

Witin the disc unit, 20 read/write heads are mounted on a single accessor mechanism that moves the heads in unison between the periphery and the central area of each disc. The accessor mechanism can assume any one of 203 positions across each disc surface; this simultaneous head movement on the 20 disc surfaces permits 200 addressable data recording cylinders in the disc pack, three cylinders being reserved for replacement tracks. Each cylinder contains 20 tracks numbered 00 through 19. Addressing of an individual track is by cylinder number (000 through 202) and by read/write head number (00 through 19). This subsystem provides file scan and record overflow as standard items.

4.10.8. 8430 Disc Subsystem

The 8430 disc subsystem consists of a control unit and from one to eight disc drive units. By adding a 16-device addressing feature, eight additional disc drive units may be controlled. The attached disc drives use removable disc packs and operate in a sequential and random access fashion.

The disc pack is the storage medium of the subsystem. Each disc pack contains 11 discs. Nineteen of the disc surfaces are used for the recording of data, and two outside surfaces serve for protection of the interior surfaces. The remaining surface is used to position the accessor mechanism. The disc packs are interchangeable between 8430 disc drives, and each pack provides the user with a maximum storage of 100 million bytes of information. There are 19 read/write heads and a positioning head mounted on a single accessor mechanism which moves in unison between the periphery and the central area of the disc. Each recording surface has its own unique head assigned to it. The accessor can assume any one of 411 positions, resulting in 411 tracks of data on each disc surface. At each position of the accessor mechanism, 19 data surfaces are available via the 19 heads (a cylinder of information). There are 7809 addressable tracks in a disc pack assembly. Addressing of an individual track is defined by cylinder (000 through 410) and head number (00 through 18). Data capacity figures are based on 404 cylinders, thus allowing the possibility of seven spare cylinders per disc pack. See Table 5—6 for complete specifications.

4.11. MULTIPLEXER CHANNEL

The multiplexer channel provides input/output capability between the processor and up to eight low-speed subsystems. The channel services several concurrently operating subsystems by assigning the I/O interface to a subsystem only long enough to transfer one byte of information and servicing other subsystems in a similar manner, if necessary, before servicing the same subsystem again.

The processor initiates all I/O operations by issuing I/O instructions to a selected multiplexer channel subsystem. Once the operation is successfully initiated, the channel maintains control of data transfers between main storage and the subsystem independently of the processor. Upon completion of the I/O operation, the state of the channel and subsystem is presented to the software by way of the appropriate status words by means of the I/O Status Tabler.

Devices which can be attached to the multiplexer channel are:

- 0716 Card Reader Subsystem — 80-column or 80/96-column
- 0604 Card Punch Subsystem
- 0770 Printer Subsystem
- UNISERVO VI-C Magnetic Tape Subsystem*
- UNISERVO 10 Magnetic Tape Subsystem
- 0768 Printer Subsystem*
- 0920 Paper Tape Subsystem
- 2703 Optical Document Reader (Physical level support only)
- 9000 Series Channel Adapter

4.11.1. 0716 Card Reader Subsystem

The 0716 Card Reader Subsystem provides online input to the processor. It is a freestanding, self-contained unit capable of reading standard 80-column or 96-column computer cards at rates of 600 or 1000 cards per minute on a column-by-column basis. The card reader contains a standard read check feature that ensures correct input. Information read from the card is transferred to the processor in either image or translate mode. Either mode includes ASCII, EBCDIC, and compressed code. Image mode and a selection of ASCII, EBCDIC, or compressed code translation are standard to the subsystem.

The basic subsystem contains a single input hopper with a capacity of 2400 eighty-column cards or 2000 ninety-six-column cards and two output stackers with a capacity of 2000 cards each. A comprehensive set of two diagnostic data bytes provides offline operation for maintenance purposes.

Card loading and unloading may be accomplished without stopping the reader. Conversion to and from 96-column operation is easily accomplished by the operator.

**Subject to availability*

4.11.2. 0604 Card Punch Subsystem

The 0604 card punch subsystem is a freestanding, self-contained unit that punches standard 80-column computer cards. The punch mechanism is an 80-column punch that feeds and punches the cards row by row at the rate of 250 cards per minute (cpm). After a card is punched, it is checked to ensure it was correctly punched. An incorrectly punched card is rejected and the next blank card is repunched under program control. Correctly punched cards are directed by program control into one of the two output card stackers. Punching is provided in either image mode or compressed code translation.

4.11.3. 0770 Printer Subsystem

The 0770 printer and control is a freestanding, self-contained unit that houses the printer mechanism and its associated control circuitry and a 132-character print buffer. The printer prints output from the processor in 132-character (columns) lines at speeds up to 2000 lines per minute (lpm). The printer is a band printer of the impact type, with interchangeable print cartridges. Three different models are available, which print alphanumeric characters at the rate of 800, 1400, or 2000 lines per minute, single spaced, using the standard 48-character set. Other character set sizes available are: 64, 96, 128, and 256.

Impact printing is defined as the driving of the hammer against the rear of the paper to force the paper in contact with a ribbon and type cartridge. Print rates of up to 3000 lpm are possible if the character set is restricted to 24 contiguous numeric characters. A print line is 132 characters on approximately 0.1-inch centers, or 10 characters per inch. Vertical spacing is six or eight lines per inch. The spacing is controlled by the vertical format buffer, which is loaded by the program with the proper spacing format. The forms stacker uses powered rotating paddles to draw the printed forms onto an automatically adjustable shelf. This stacker can accommodate forms up to 22 inches in width and 18.75 inches in length; however, paper up to 24 inches in length can be accommodated. The forms are advanced at a slow rate of 50, 75, or 100 inches per second, depending on model selected. When the 48-character type cartridge is being used, a complete line of print and form advancement is accomplished in 30 milliseconds.

4.11.4. UNISERVO VI-C Magnetic Tape Subsystem

The UNISERVO VI-C Magnetic Tape Subsystem is a freestanding, self-contained unit which provides an external storage medium capable of reading from (forward or backward) or writing to (forward) a magnetic tape. The subsystem consists of a control unit and from two to eight magnetic tape units referred to as the master and slave units. The master unit contains the power supply and control circuitry to govern the functions of up to three slave magnetic tape units. The control unit, one master unit, and one slave unit are housed in a single freestanding cabinet.

The UNISERVO VI-C Magnetic Tape Units are available in either 9-track or 7-track models. The 9-track model reads and records data in 8-bit EBCDIC, with each frame that records across the width of the tape containing eight data bits plus a parity bit (one byte). Data is recorded in NRZI mode at a density of 800 bits per inch.

The 7-track model reads and writes data in 6-bit binary coded decimal, with each frame that records across the width of the tape containing six data bits plus a parity bit. Data is recorded in NRZI mode at densities of 800, 556, or 200 bits per inch, with tape instructions establishing applicable density.

4.11.5. UNISERVO 10 Magnetic Tape Subsystem

The UNISERVO 10 Magnetic Tape Subsystem is a freestanding, self-contained unit which provides an external storage medium capable of reading from (forward or backward) or writing to (forward) a magnetic tape. The subsystem consists of a control unit and cabinet which has space to mount the two required tape units. A maximum of eight tape units, in increments of one or two units (beyond the two required) can be attached to the control through the use of up to three auxiliary cabinets.

The UNISERVO 10 Magnetic Tape Subsystem is available in either 9-track or 7-track models. The 9-track units operate in the phase-encoded mode at 1600 bits per inch, and, if the optional dual density feature is added, in the NRZI mode at 800 bits per inch. The 7-track units operate in the NRZI mode at recording densities of 800, 556, or 200 bits per inch, depending on the density selected.

Physical tape speed is 25 inches per second. This results in a 9-track transfer rate of 40,000 frames per second in the phase-encoded mode and 20,000 frames per second in the NRZI mode. The 7-track NRZI transfer rate is 20,000; 13,900; or 5,000 frames per second, depending on the density selected.

The dual channel feature permits nonsimultaneous operation on two channels of one processor or on one channel on each of two processors. (See 4.11.10.) Simultaneous operation is not a feature of the UNISERVO 10 Magnetic Tape Subsystem.

4.11.6. 0768 Printer Subsystem

The 0768 printer and control is a freestanding, self-contained drum printer that houses the control and synchronizing circuitry, a 132-character print buffer, and the print mechanism. A container at the base of the unit houses the forms being fed into the printer. Controls are provided for the manual adjustment of paper tension, form thickness, paper alignment, vertical print positioning, horizontal print positioning, and the advancement of forms. The various printer types available are:

- Type 0768—00

  Prints a maximum of 1100 lines per minute, depending on the number of characters used. The full character set consists of 63 printable characters on a drum 3 inches in diameter. If all characters to be printed on a line are contained within a 49-contiguous-character subset on the drum and single spacing is specified, printing is at the rate of 1100 lpm. If more than 49 contiguous characters are specified or if spacing other than single spacing is desired, printing speed decreases accordingly.

- Type 0768—02

  Prints at a maximum rate of 840, 1000, or 2000 lines per minute, depending on the number of characters used. A full character set consists of 94 printable characters on a drum 5 inches in diameter. The subsystem prints at the rate of 840 lpm when 94 contiguous characters are used, 1000 lpm when 87 contiguous characters are used, and 2000 lpm when a numeric set of 14 characters is used.

- Type 0768—99

  Prints at a maximum rate of 1200 and 1600 lpm, again depending on the number of characters used. A full character set consists of 132 printable characters on a drum 5 inches in diameter. The subsystem prints at the rate of 1200 lpm when 63 contiguous characters are used, and 1600 lpm when 43 contiguous characters are used.

4.11.7. 0920 Paper Tape Subsystem

The 0920 paper tape subsystem is a freestanding, self-contained unit that houses the necessary controls and logic to communicate with the processor. The paper tape subsystem can read perforated tape of five, six, seven, or eight channels at a rate of 300 characters per second and punch paper tape at the rate of 110 characters per second. Essentially, the paper tape subsystem consists of a control unit, a tape reader and reader synchronizer, and/or a tape punch and punch synchronizer. The control unit provides the necessary synchronization and interface between the reader and punch synchronizers and the multiplexer channel. The synchronizer units regulate the transfer of data characters between the tape reader or tape punch and the control unit. The transfer rate is determined by the mechanical speed of the reader mechanism or punch mechanism. Tape parity checking or generation, as well as data bit and character manipulations, is performed in the synchronizers in accordance with the hard wiring of a program connector. Spooling features are available for both the tape reader and tape punch portions of the subsystem.

4.11.8. 2703 Optical Document Reader

The 2703 optical document reader (ODR) is a freestanding, self-contained subsystem controlled by the commands from the processor through the multiplexer channel. The ODR feeds, reads, sorts, and stacks documents containing a variety of data in the form of printed optical characters, or mark-encoded data. The combination of a dual-belt document feed, a solid-state photoelectric sensing device, and carousel stackers provide for gentle document handling. The ODR is capable of reading 300 documents per minute. The hopper capacity is 2000 documents and the stacker capacity is 1000 documents. In addition, optional features available are:

- Mark read — a second read station having a sensing head which detects vertical pencil marks arranged in columns on the document.
- Punch-card feed — provides additional capability for reading 80-column punched cards.
- Validity check — allows the ODR to detect more than one hole or mark in rows one through seven in any column of a punched card or a mark read document.
- Modulus 10 check digit — verifies correctness of the numeric optical reading.
- Speed upgrade — enables document reading at up to 600 documents per minute.

See Table 5—13 for further specifications.

4.11.9. Channel Adapter

The channel adapter provides communication between the 90/30 system processor and 9200/9300 series processors by way of their respective multiplexer channels. The channel adapter is housed in the processor cabinet of the external system being connected to the 90/30 system. External systems capable of being attached to the channel adapter are the 9200/9200 II/9300/9300 II systems. These systems can be used to provide card reading, card punching, and printing capability for the 90/30 system.

4.11.10. Cochanneling

The 90/30 system offers cochanneling of two selector channels. Cochanneling connects two selector channels and a subsystem so that the subsystem can be accessed through either of the two channels. I/O operations in this type of connection can be simultaneous or nonsimultaneous, depending on the configuration. Simultaneous operations require two control units, one on each channel, with each device connected to both control units. This configuration is called dual access cochanneling. Another configuration which is used for nonsimultaneous I/O operations is called dual channel cochanneling. This setup requires only one control unit connected to both selector channels and devices. Dual channel connection to two different processors offers a convenient method of switching a subsystem from one CPU to another. However, there is no software capability to allow sharing or concurrent use of a subsystem by two processors.

5. 90/30 System Specifications

5.1. GENERAL

This section summarizes the specifications for the 90/30 system. Condensed data about the design features is presented in tabular form to provide a general understanding of the hardware characteristics. In addition to the basic configuration, information is given for optional features of the processor and peripherals.

5.2. SYSTEM PROCESSOR SPECIFICATIONS

The basic equipment types which are an integral part of the 90/30 processor are the processing unit, main storage, and I/O control. Table 5—1 gives the characteristics of the basic components. For more detailed information, consult the 90/30 system processor programmer reference, UP-8052 (current version).

Table 5—1. 90/30 System Basic Components (Part 1 of 2)

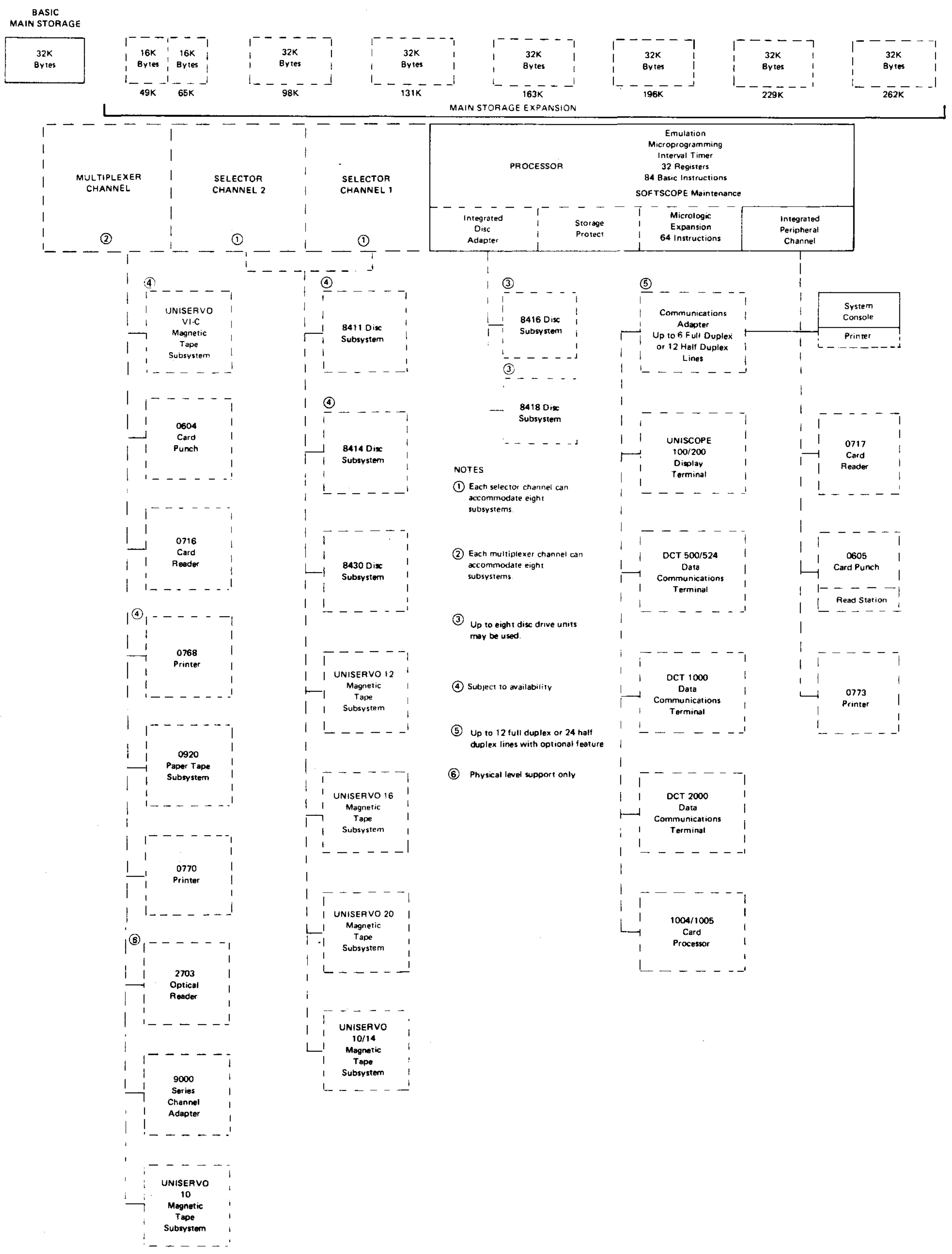
Characteristic	Description
System orientation	Disc
Processing unit	Versatile instruction set under microprogram control Interval timer Hardware-assisted dynamic storage relocation General register stack Six interrupt levels
Main storage capacity	32K, 49K, 65K, 98K, 131K, 163K, 196K, 229K, or 262K bytes
Main storage performance	A cycle time of 600 nanoseconds for a 2-byte access
Data organization	8-bit bytes; 2 bytes per main storage access
Modes of operation	Native mode 9200/9300 compatibility mode 360/20 compatibility mode
Floating-point arithmetic	Optional feature

Table 5—1. 90/30 System Basic Components (Part 2 of 2)

Characteristic	Description
Registers	32 general-purpose (4 bytes each) 24 working (4 bytes each) 4 floating-point (8 bytes each) optional feature
System console	UNISCOPE 100 Display Terminal: Monitors operation of system Provides keyboard and a visual display screen Provides interface to connect a communications printer to display unit Keyboard/video display: 64-character capability Program controlled Protected format Display screen: 16 lines, 64 characters per line Communications output printer (COP) 64-character capability Print rate of 30 characters per second Printing from system console buffer Printing initiated by program
Integrated peripheral channel	Data transfer rate of up to 50K bytes per second Up to 28 device addresses
Integrated disc adapter	Data transfer rate of 625K bytes per second Up to 8 disc units
Selector channel (2)	Data transfer rate of up to 825K bytes per second Up to 8 standard control units Burst mode of operation (second selector channel is identical)
Multiplexer channel	Data transfer rate of up to 83K bytes per second 9000 series compatible interface Up to 8 standard control units Operates in multiplex (byte) mode

5.3. SYSTEM PERIPHERAL SPECIFICATIONS

A complete line of peripheral equipment is available for use with the 90/30 system. Figure 5—1 provides a graphic presentation of the 90/30 system configuration which shows the range of SPERRY UNIVAC subsystems available. Table 5—2 lists the names of these subsystems, their related channel connections, and references to tables that present their specifications. By grouping the specifications according to the function the subsystems perform, the user can make comparisons prior to selecting the peripheral equipment.



NOTES

① Each selector channel can accommodate eight subsystems.

② Each multiplexer channel can accommodate eight subsystems.

③ Up to eight disc drive units may be used.

④ Subject to availability

⑤ Up to 12 full duplex or 24 half duplex lines with optional feature

⑥ Physical level support only

Figure 5—1. 90/30 System Configuration

Table 5—2. Peripheral Equipment Summary

SPERRY UNIVAC Subsystem	Channel Connector	Reference
0717 Card Reader 0716 Card Reader	Integrated Peripheral Multiplexer	Table 5—3
0605 Card Punch 0604 Card Punch	Integrated Peripheral Multiplexer	Table 5—4
0773 Printer 0768 Printer 0770 Printer	Integrated Peripheral Multiplexer Multiplexer	Table 5—5
8416 Disc 8418 Disc 8411 Disc 8414 Disc 8430 Disc	Integrated Disc Adapter Integrated Disc Adapter Selector Selector Selector	Table 5—6
UNISERVO 10 Magnetic Tape UNISERVO 12 Magnetic Tape UNISERVO 14 Magnetic Tape UNISERVO 16 Magnetic Tape UNISERVO 20 Magnetic Tape UNISERVO VI-C Magnetic Tape	Multiplexer/Selector Selector Selector Selector Selector Multiplexer	Table 5—7
0920 Paper Tape	Multiplexer	Table 5—8
UNISCOPE 100/200 Display Terminal	Communications Adapter	Table 5—9
DCT 500/524 Data Communications Terminal DCT 1000 Data Communications Terminal DCT 2000 Data Communications Terminal	Communications Adapter Communications Adapter Communications Adapter	Table 5—10
1004/1005 Card Processor	Communications Adapter	Table 5—11
90/30 Communications Adapter	Integrated Peripheral	Table 5—12
2703 Optical Reader	Multiplexer	Table 5—13

Table 5–3. Card Reader Subsystems Characteristics

Characteristic	Description	
	0716 Card Reader Subsystem	**0717 Card Reader Subsystem**
Card orientation (80-, 66-, and 51-column cards)	Face in, with column 1 leading and row 9 down (Model available to read 96-column cards)	Face in, with column 1 leading and row 9 down
Card rate	600 or 1000 cpm	500 cpm (maximum)
Read technique	Dual, redundant, solar cell technique, using photo transistors Column 0 amplifier checking	Dual, redundant, solar cell technique, using photo transistors Column 0 amplifier checking
Read modes	Image mode — 160 six-bit characters per card Translate mode — 80 characters per card Three available codes: ■ 8-bit ASCII ■ 8-bit EBCDIC (Required) ■ Compressed code	Image mode: 160 six-bit characters per card Translate mode: 80 characters per card Available code: 8-bit EBCDIC
Read station sensing	Column by column	Column by column
Hopper capacity	2400 80-column cards 2000 96-column cards	2400 cards
Stacker capacity Normal Reject	 2000 cards (stacker 2) 2000 cards (stacker 1)	 2000 cards —

Table 5—4. Card Punch Subsystems Characteristics

Characteristic	Description	
	0604 Card Punch Subsystem	**0605 Card Punch Subsystem**
Media	80-column cards	80-column cards
Punch mode	Row	2-column serial
Check mode	Read of punched data	Punch motion check
Feed mode	On demand	On demand
Punch rate	250 cpm	75 cpm (full card) 160 cpm (28 columns only)
Input hopper capacity	1000 cards	700 cards
Output stacker capacity	1000 cards (normal and select stacker)	700 cards (primary stacker) 100 cards (reject stacker)
Reading	Optional	Optional
Read rate	250 cpm	160 cpm
Punch translation		
Image mode	160 six-bit characters per card	160 six-bit characters per card
Compressed mode	80 characters per card	80 characters per card
Available code	———	EBCDIC

Table 5–5. *Printer Subsystems Characteristics (Part 1 of 3)*

0773 Printer Subsystem			
Characteristic	**Description**		
Print speed	110 to 680 lpm depending on character contingencies:		
	Available character sets	Number of sets per band	Nominal print rate (lpm)
	48-character business	5	500
	63-character print	4	400
	48/16-character print	4	400/670
	85-character print	3 (plus 1 character)	310
	128-character special	2	217
	96/(16-16)-character ASCII	2	217/500
	256-character special	1	114
Line advance timing	Advance and Print	Time in ms	
		6 lpi	8 lpi
	1 line	120.0	118.0
	2 lines	127.6	123.7
	3 lines	135.2	129.4
	n+1 lines	120+7.6n	118+5.7n
Number of print positions	120 print positions (columns) by standard printer; 132 or 144 columns by feature		
Form advance control	Vertical format buffer		
Line advance rate	Single space only, 22 inches/second		
Form dimensions	4 to 18.75 inches wide 1 to 24 inches long		
Character set	Standard 48-character set. Any number of characters up to 256 with options		
Horizontal spacing	10 characters per inch		
Vertical spacing	6 or 8 lines per inch, operator selectable		

Table 5–5. Printer Subsystems Characteristics (Part 2 of 3)

0770 Printer Subsystem			
Characteristic	**Description**		
Print speed	Type 0770–00	Type 0770–02	Type 0770–04
	112 to 1435 lpm, depending on character contingencies 112 lpm 384-character set 800 lpm 48- character set 1435 lpm 24-character set	213 to 2320 lpm, depending on character contingencies 213 lpm 348-character set 1400 lpm 48-character set 2320 lpm 24-character set	337 to 3000 lpm, depending on character contingencies 337 lpm 384-character set 2000 lpm 48-character set 3000 lpm 24-character set
Line advance timing	8.75 ms for spacing first line; for skipping each subsequent line: 3.33 ms at 6 lpi 2.50 ms at 8 lpi	8.75 ms for spacing first line; for skipping each subsequent line: 2.22 ms at 6 lpi 1.67 ms at 8 lpi	8.75 ms for spacing first line; for skipping each subsequent line: 1.67 ms at 6 lpi 1.25 ms at 8 lpi
Number of print positions	Full print width of 132 print positions placed anywhere on a 16.5 inch form. With 22-inch form, only central 13.2 inch portion can be used (160 print positions with feature)		
Form advance control	Vertical Format Buffer		
Line advance rate	50 ips	75 ips	100 ips
Form dimensions	Continuous forms with standard edge sprocket holes from 4 to 22 inches in width. Carbons may be attached or unattached with multicopy forms to a maximum of six parts. Recommended pack thickness not to exceed .0155 inch for high quality print		
Character set	Standard 48-character set. Any number of characters up to 384 with options		
Horizontal spacing	10 characters per inch		
Vertical spacing	6 to 8 lines per inch as determined by program		

Table 5–5. Printer Subsystems Characteristics (Part 3 of 3)

0768 Printer Subsystem			
Characteristic	**Description**		
Print speed	Type 0768–00	Type 0768–02	Type 0768–99
	900 through 1100 lpm	840 through 2000 lpm	1200 through 1600 lpm
Line advance timing	11.5 + 5.1 (n–1) ms – 6 lines per inch 11.5 + 5.7 (n–1) ms – 8 lines per inch where: n = number of lines advanced		
Number of print positions	132 character print positions		
Form advance control	Loop control; up to 132 lines per command		
Line advance rate	25 ips		
Form dimensions	4 to 22 inches wide 1 to 22 inches long		
Character set	0768–00 63 characters 0768–02 94 characters 0768–99 63 characters		
Horizontal spacing	10 characters per inch		
Vertical spacing	6 to 8 lines per inch		

Table 5–6. Disc Subsystems Characteristics

Characteristics	Description				
	8416 Disc Subsystem	**8418 Disc Subsystem**	**8411 Disc Subsystem**	**8414 Disc Subsystem**	**8430 Disc Subsystem**
Data capacity (8-bit bytes) per pack	28.95 million	28.9 million or 57.9 million	7.25 million	29.17 million	100 million
Number of disc units	2 to 8	2 to 8	1 to 8	2 to 8	1 to 8 (with optional feature up to 16)
Disc speed	2800 rpm	2800 rpm	2400 rpm	2400 rpm	3600 rpm
Data bit rate	5.0 MHz	5.0 MHz	1.25 MHz	2.5 MHz	6.45 MHz
Bit density	4040 pulse per inch (ppi)	4040 ppi	1100 ppi	2200 ppi	4040 ppi
Track density	192 tracks per inch	370 tracks per inch	100 tracks per inch	200 tracks per inch (free format)	192 tracks per inch
Number of tracks	404+7 spare usable tracks per disc surface	404 or 808+7 spare usable tracks per disc surface	200+3 spare usable tracks per disc surface	200+3 spare usable tracks per disc surface	404+7 spare tracks per disc surface
Number of surfaces per disc unit	Data 7 Positioning 1	Data 7 Positioning 1	10	20	Data 19 Positioning 1
Positioning time Minimum Average Maximum	 10 ms 30 ms 60 ms	 10 ms 27 ms 45 ms	 25 ms 75 ms 135 ms	 25 ms 60 ms 130 ms	 7 ms 27 ms 50 ms
Transfer rate	625 kilobytes	628 kilobytes	156 kilobytes	312 kilobytes	806 kilobytes

Table 5–7. UNISERVO Subsystems Characteristics

Characteristic	Description										
	UNISERVO 10		UNISERVO 12		UNISERVO 14		UNISERVO 16		UNISERVO 20	UNISERVO VI-C	
Tape units per subsystem	2 to 8		1 to 16		2 to 8		1 to 16		1 to 16	2 to 8	
Data transfer rate (maximum)	40,000 frames per second		68,320 frames per second		96,000 frames per second		192,000 frames per second		320,000 frames per second	34,160 frames per second	
Tape speed	25 inches per second		42.7 inches per second		60 inches per second		120 inches per second		200 inches per second	42.7 inches per second	
Tape direction											
Reading	Forward or backward		Forward or backward		Forward or backward		Forward or backward		Forward or bakcward	Forward or backward	
Writing	Forward		Forward		Forward		Forward		Forward	Forward	
Tape length (maximum)	2400 ft		2400 ft		2400 ft		2400 ft		2400 ft	2400 ft	
Tape thickness	1.5 mils		1.5 mils		1.5 mils		1.5 mils		1.5 mils	1.5 mils	
Block length	Variable		Variable		Variable		Variable		Variable	Variable	
Interblock gap	9-track	7-track	9-track	7-track	9-track	7-track	9-track	7-track	0.6 in.	9-track	7-track
	0.6 in.	0.75 in.	0.6 in.	0.75 in.	0.6 in.	0.75 in.	0.6 in.	0.75 in.		0.6 in.	0.75 in.
Interblock gap time											
Nonstop	24 ms	30 ms	14.1 ms	17.6 ms	11 ms	12.5 ms	5.0 ms	6.25	3.0 ms	14.1 ms	NA
Start-stop	41 ms	47 ms	20.1 ms	23.6 ms	17 ms	18.5 ms	8.0 ms	9.25	5.0 ms	20.1 ms	
Pulse density	1600 ppi 800 ppi	800 ppi 556 ppi 200 ppi	1600 ppi 800 ppi	800 ppi 556 ppi 200 ppi	1600 ppi 800 ppi	800 ppi 556 ppi 200 ppi	1600 ppi 800 ppi	800 ppi	1600 ppi	800 ppi	800 ppi 556 ppi 200 ppi
Recording mode	Phase encoded or NRZI	NRZI	Phase encoded or NRZI	NRZI	Phase encoded or NRZI	NRZI	Phase encoded or NRZI	NRZI	Phase encoded	NRZI	
Reversal time	16 ms		25 ms		6.3 ms		10 ms		16 ms	25 ms	
Rewind time	3 min		3 min		3 min		2 min		1 min	3 min	
Simultaneous operation	NA		Optional		NA		Optional		Optional	Optional	

Table 5–8. 0920 Paper Tape Subsystem Characteristics

Characteristic	Description
Reader mounting	Mounted on a 7- by 9-inch panel having a pin spindle for handling reels containing up to 50 feet of tape (for tape reader without an optional spooler)
Tape read	Unidirectional (right to left)
Tape channel capacity	Capable of reading 11/16-inch, 7/8-inch, or 1-inch paper tape; 3-position tape guide available to adjust to tape width used
Read speed	300 characters per second at 10 characters per inch
Type of tape	All conventional perforated tapes with a light transmissivity of 40% or less
Stop and start capacities	Can stop on character or before next character; on start, unit reaches full speed within two characters
Tape spooler	Up to 5-inch reels can be used with the spooler to allow reeling of approximately 300 feet of paper tape
Tape leader	Approximately 3 feet of tape leader when spooler mechanism is used
Tape trailer	A 12-inch trailer is provided to prevent false broken tape indication
Punch mounting	Mounted within a 14- by 19-inch panel
Tape channel capacity	Handles paper tape width of 11/16 inch or 1 inch; five levels of tape characters with 11/16-inch paper tape being used; or 5, 6, 7, or 8 levels of tape characters with 1-inch paper tape in use. Tape guide adjusts to conform to paper tape width.
Punch speed	110 characters per second at 10 characters per inch
Type of tape	Oil base paper tape is provided. A compatible tape utilizing a paper-plastic-paper sandwich is also available.
Stop and start capabilities	Punching is performed one character at a time. Tape punch is capable of stopping and starting between characters.
Tape feeding	The tape punch handles a paper tape reel of 1000 feet with sensing signals to indicate low paper tape supply.

Table 5–9. UNISCOPE 100/200 Display Terminal Characteristics

Characteristic	Description	
	UNISCOPE 100	**UNISCOPE 200**
Number of terminals (max.)	256	256
Display capacity	960 or 1024 characters	1536 or 1920 characters
Display format	64 characters per line by 16 lines 80 characters per line by 12 lines	64 characters per line by 24 lines 80 characters per line by 24 lines
Display character set	64 or 96 symbols (ASCII)	64 or 96 symbols (ASCII)
Keyboard	Numeric, alphanumeric, or combination of numeric and alphanumeric, uppercase and lowercase Cursor control keys Editing keys Protected and unprotected format keys Special function keys	Numeric, alphanumeric, or combination of numeric and alphanumeric, uppercase and lowercase Cursor control keys Editing keys Protected and unprotected format keys Special function keys
Interface type with data transfer rate	Communications line (telephone) RS232B Synchronous Western Electric 201A – 2000 bps Western Electric 201B – 2400 bps Western Electric 203, up to 9600 bps Collins TE 216A, up to 4800 bps Lenkurt 26C, up to 2400 bps Milgo 4400/4800, up to 4800 bps Rixon Sebit, up to 4800 bps	Communications line (telephone) RS232B Synchronous Western Electric 201A – 2000 bps Western Electric 201B – 2400 bps Western Electric 203, up to 9600 bps Collins TE 216A, up to 4800 bps Lenkurt 26C, up to 2400 bps Milgo 4400/4800, up to 4800 bps Rixon Sebit, up to 4800 bps
Character size Height Width	 0.14 inch nominal 0.10 inch nominal	 0.14 inch nominal 0.10 inch nominal
Refresh rate	60 cycles per second	60 cycles per second
Scan method	Digital	Digital
Character generation	Closed stroke, maximum of 8 per character	Closed stroke, maximum of 8 per character

Table 5—10. Data Communications Terminals Characteristics

Data Communications Terminal			
Characteristic	Description		
	DCT 500/524	DCT 2000	DCT 1000
Transmission code	8-level ASCII	ASCII XS-3 (DLT compatible)	
Transmission mode	Half-duplex or full-duplex 2- or 4-wire	Half-duplex; 2- or 4-wire (nonsimultaneous 2-way transmission)	Half-duplex, 2- or 4-wire (non-simultaneous 2-way transmission)
Transmission rate	110, 150, or 300 bits per second	2400 bauds per second (private line) 2000 bauds per second (switched telephone network)	Asynchronous 300, 1200, or 1800 bauds per second; synchronous up to 9600 bauds per second
Printing rate	30 characters per second	250 lines per minute	30 characters per second
Font selection	ASCII, EBCDIC, A (business), H (scientific)	ASCII, EBCDIC, A (business), H (scientific), ECMA/ISO*	ASCII, EBCDIC, A (business), H (scientific), ECMA/ISO*
Printable characters	63 plus space	63 plus space	63 plus space
Print positions per line	132 (adjustable tractor)	80 or 128	132 (adjustable tractor)
Paper tape reader/ punch rate	50 characters per second		50 characters per second
Transmission mode	Voice grade telephone exchange or private line	Voice grade telephone exchange or private line	Voice grade telephone exchange or private line
Card reading rate	NA	200 cards per minute	40 cards per minute
Card punching rate	NA	75 to 200 cards per minute	35 cards per minute
Buffer storage	NA	256-character capacity in two buffers, 128 characters each	320-character capacity in two buffers, 160 characters each
Card read translator selections	NA	EBCDIC/ASCII, A (business), H (scientific)	EBCDIC/ASCII, A (business), H (scientific)
Transmission method	Character-by-character	Block by block	Block by block

*European Computer Manufacturers Association/International Standards Organization
NA — Not applicable

Table 5–11. 1004/1005 Card Processor Characteristics

Characteristic	Model I	Model II	Model III
Card columns	80/90	80/90	80/90
Processor access time, microseconds	8	6.5	6.5
Read card rate, cards per minute	400	615	615
Printing, lines per minute	400	600	200
Card punch rate, cards per minute	200	200	200
Magnetic tape units, characters per second	–	–	up to 34,160
Storage capacity (locations)	961, 2048 or 4096	961, 2048 or 4096	2048 or 4096

Table 5–12. 90/30 Communications Adapter

Type	Description
Synchronous Line Adapter (RS232, CCITT)	Provides a full-duplex or half-duplex interface to synchronous data sets conforming to RS232 and CCITT.
Synchronous Line Adapter (MIL 188C)	Permits exact compliance with MIL 188C low level interface. Control line polarity is RS232.
Asynchronous Line Adapter (RS232, CCITT)	Provides a full-duplex or half-duplex interface to asynchronous data sets conforming to RS232 and CCITT. Compatible with MIL 188C low level interface electrical characteristics.
Asynchronous Line Adapter (MIL 188C)	Permits exact compliance with MIL 188C low level interface. Control line polarity is RS232.
U.S. Wideband Line Adapter	Provides a synchronous full duplex interface to the 301B2 data set operating at 40.8k bits per second or the Bell 303 data set. Maximum data rate of the adapter is 56k bits per second.
Asynchronous Relay Line Adapter	Provides an asynchronous full duplex interface optionally compatible with either 20-75MA neutral or 10-40MA polar telegraph lines.
TWX Line Adapter	Provides an interface to the TWX network in the U.S.
U.S. TELEX Line Adapter	Provides an interface to the TELEX network in the U.S.
TELEX Adapter, International	Provides an interface to international TELEX lines.

NOTE:

The communications adapter (CA) supports up to 12 half-duplex or 6 full-duplex communication lines at speeds up to 56k bits per second. An optional feature is available that will support up to 24 half-duplex or 12 full-duplex lines. The active line indicator for the CA provides a display panel to display individual line activity.

Table 5–13. 2703 Optical Document Reader Characteristics

Characteristic	Description
Document speed	150 inches per second (from feed station, past read station, to stacker).
Document sizes: Length Width Thickness	 30 to 8.75 inches 2.75 to 4.25 inches .003 to .010 inch
Document color	White background
Document weight	20-pound minimum 62-pound maximum (Standard punched card is 44-pound paper.)
Printed image dimensions	In accordance with USASCSOCR and UNIVAC H–14 font.
Document rate	300 documents per minute (dpm) for 6-inch documents, or 600 dpm with speed-up feature 1108.
Character reading rate	1667 characters per second when a minimum of .090-inch spacing for each character is used. System receiving data from the ODR must be capable of accepting characters at an average rate of 1500 characters per second.
Character read error and reject rates	On typical format of 30 characters: ■ error rate is 0.3 percent without check digit verification; and ■ reject rate is 3 percent.
Hopper capacity	2000 documents
Stacker selection	One of three stackers is selected after each read command, or one stacker is preselected by stacker mode command.
Stacker capacity	1000 documents

Appendix A. System Instructions

The 90/30 system instructions are listed alphabetically by function set. The mnemonic source code and hexadecimal operation code also are given.

Branching			
Instruction	**Mnemonic Code**	**Operation Code**	**Type**
Branch and link	BAL	45	RX
Branch and link	BALR	05	RR
Branch and store*	BAS	4D	RX
Branch and store*	BASR	0D	RR
Branch on condition	BC	47	RX
Branch on condition	BCR	07	RR
Branch on count	BCT	46	RX
Branch on count	BCTR	06	RR
Branch on index high	BXH	86	RS
Branch on index low or equal	BXLE	87	RS
Execute	EX	44	RX
Add decimal	AP	FA	SS
Compare decimal	CP	F9	SS
Divide decimal	DP	FD	SS
Multiply decimal	MP	FC	SS
Move with offset	MVO	F1	SS
Pack	PACK	F2	SS
Subtract decimal	SP	FB	SS
Unpack	UNPK	F3	SS
Zero and add	ZAP	F8	SS

*IBM 360/20 compatibility mode only

Fixed Point			
Instruction	Mnemonic Code	Operation Code	Type
Add	A	5A	RX
Add half word	AH	4A	RX
Add immediate	AI	9A	SI
Add	AR	1A	RR
Compare	C	59	RX
Compare half word	CH	49	RX
Compare	CR	19	RR
Convert to binary	CVB	4F	RX
Convert to decimal	CVD	4E	RX
Divide	D	5D	RX
Divide	DR	1D	RR
Load	L	58	RX
Load complement	LCR	13	RR
Load half word	LH	48	RX
Load multiple	LM	98	RS
Load negative	LNR	11	RR
Load positive	LPR	10	RR
Load	LR	18	RR
Load and test	LTR	12	RR
Multiply	M	5C	RX
Multiply half word	MH	4C	RX
Multiply	MR	1C	RR
Subtract	S	5B	RX
Subtract half word	SH	4B	RX
Shift left single	SLA	8B	RS
Shift left double	SLDA	8F	RS
Subtract	SR	1B	RR
Shift right single	SRA	8A	RS
Shift right double	SRDA	8E	RS
Store	ST	50	RX
Store half word	STH	40	RX
Store multiple	STM	90	RS

Floating Point			
Instruction	**Mnemonic Code**	**Operation Code**	**Type**
Add normalized (long format)	AD	6A	RX
Add normalized (long format)	ADR	2A	RR
Add normalized (short format)	AE	7A	RX
Add normalized (short format)	AER	3A	RR
Add unnormalized (long format)	AW	6E	RX
Add unnormalized (long format)	AWR	2E	RR
Add unnormalized (short format)	AU	7E	RX
Add unnormalized (short format)	AUR	3E	RR
Compare (long format)	CD	69	RX
Compare (long format)	CDR	29	RR
Compare (short format)	CE	79	RX
Compare (short format)	CER	39	RR
Divide (long format)	DD	6D	RX
Divide (long format)	DDR	2D	RR
Divide (short format)	DE	7D	RX
Divide (short format)	DER	3D	RR
Halve (long format)	HDR	24	RR
Halve (short format)	HER	34	RR
Load complement (long format)	LCDR	23	RR
Load complement (short format)	LCER	33	RR
Load (long format)	LD	68	RX
Load (long format)	LDR	28	RR
Load (short format)	LE	78	RX
Load (short format)	LER	38	RR
Load negative (long format)	LNDR	21	RR
Load negative (short format)	LNER	31	RR
Load positive (long format)	LPDR	20	RR
Load positive (long format)	LPER	30	RR
Load and test (long format)	LTDR	22	RR
Load and test (short format)	LTER	32	RR
Multiply (long format)	MD	6C	RX
Multiply (long format)	MDR	2C	RR
Multiply (short format)	ME	7C	RX
Multiply (short format)	MER	3C	RR
Subtract normalized (long format)	SD	6B	RX
Subtract normalized (long format)	SDR	2B	RR
Subtract normalized (short format)	SE	7B	RX
Subtract normalized (short format)	SER	3B	RR
Store (long format)	STD	60	RX
Store (short format)	STE	70	RX
Subtract unnormalized (long format)	SW	6F	RX
Subtract unnormalized (long format)	SWR	2F	RR
Subtract unnormalized (short format)	SU	7F	RX
Subtract unnormalized (short format)	SUR	3F	RR

Logical			
Instruction	**Mnemonic Code**	**Operation Code**	**Type**
Add logical	AL	5E	RX
Add logical	ALR	1E	RR
Compare logical	CL	55	RX
Compare logical	CLC	D5	SS
Compare logical	CLI	95	SI
Compare logical	CLR	15	RR
Edit	ED	DE	SS
Edit and mark	EDMK	DF	SS
Insert character	IC	43	RX
Load address	LA	41	RX
Move character	MVC	D2	SS
Move immediate	MVI	92	SI
Move numeric	MVN	D1	SS
Move zones	MVZ	D3	SS
AND	N	54	RX
AND	NC	D4	SS
AND	NI	94	SI
AND	NR	14	RR
OR	O	56	RX
OR	OC	D6	SS
OR	OI	96	SI
OR	OR	16	RR
Subtract logical	SL	5F	RX
Shift left double logical	SLDL	8D	RS
Shift left single logical	SLL	89	RS
Subtract logical	SLR	1F	RR
Shift right double logical	SRDL	8C	RS
Shift right single logical	SRL	88	RS
Store character	STC	42	RX
Test under mask	TM	91	SI
Translate	TR	DC	SS
Translate and test	TRT	DD	SS
Exclusive OR	X	57	RX
Exclusive OR	XC	D7	SS
Exclusive OR	XI	97	SI
Exclusive OR	XR	17	RR

Nonprivileged Status Switching			
Instruction	**Mnemonic Code**	**Operation Code**	**Type**
Set program mask	SPM	04	RR
Supervisor call	SVC	0A	RR
Test and set	TS	93	SI
Privileged			
Diagnose	DIAG	83	SI
Halt and proceed	HPR	99	SI
Insert storage key	ISK	09	RR
Load control storage	LCS	BI	RS
Load program status word	LPSW	82	SI
Start I/O	SIO	9C	SI
Supervisor load multiple	SLM	B8	RS
Service timer register	STR	03	RR
Softscope forward scan	SSFS	A2	RS
Set storage key	SSK	08	RR
Set system mask	SSM	80	SI
Softscope reverse scan	SSRS	A3	RS
Supervisor store multiple	SSTM	B0	RS

Appendix B. Instruction Execution Timings

Table B–1. Basic Instructions (Part 1 of 3)

Instruction	Mnemonic Code	Operation Code	Type	Time in Microseconds*
Add	AR	1A	RR	3.0
Add	AR (360/20)	1A	RR	3.6
Add	A	5A	RX	5.4
Add decimal	AP	FA	SS	36.6 + 0.75n1 + 0.375n2 + 6.0t1 + 3.0s5
Add half word	AH	4A	RX	5.4
Add half word	AH (9300 only)	AA	RX	5.4
Add immediate	AI	9A	SI	6.0
Add immediate	AI (9300 only)	A6	SI	6.0
AND	NR	14	RR	3.0
AND	N	54	RX	5.4
AND	NI	94	SI	6.0
AND	NC	D4	SS	10.2 + 1.5n
Branch and link	BALR	05	RR	3.6 + 0.6s
Branch and link	BAL	45	RX	6.0
Branch and store	BASR	0D	RR	3.0 + 0.6s
Branch and store	BAS	4D	RX	5.4
Branch on condition	BCR	07	RR	3.0
Branch on condition	BC	47	RX	3.6
Branch on count	BCTR	06	RR	3.6
Branch on count	BCT	46	RX	4.2
Compare	CR	19	RR	3.0
Compare	C	59	RX	5.4
Compare decimal	CP	F9	SS	31.8 +.375n1 + 0.375n2 + 2.4s6
Compare half word	CH	49	RX	5.4
Compare logical	CLR	15	RR	3.0
Compare logical	CL	55	RX	5.4
Compare logical	CLI	95	SI	4.8
Compare logical	CLC	D5	SS	9.6 + 1.2b
Convert to binary	CVB	41	RX	36.0
Convert to decimal	CVD	4E	RX	66.0 + 6.0s4

*See Table B–4.

Table B–1. Basic Instructions (Part 2 of 3)

Instruction	Mnemonic Code	Operation Code	Type	Time in Microseconds*
Divide	D	5D	RX	65.4 + 1.2s1 + 0.6rn
Divide decimal	DP	FD	SS	37.8 + 0.75n1 + 6.375n2 + 24.6 (n1–n2)
Edit	ED	DE	SS	9.0 + 3.0n + 0.6n3 + 3.0n4 + 0.6n6
Exclusive OR	XR	17	RR	3.0
Exclusive OR	X	57	RX	5.4
Exclusive OR	XI	97	SI	6.0
Exclusive OR	XC	D7	SS	10.2 + 1.5n
Execute	EX	44	RX	3.6 + 0.6r + 0.6nrr + e
Halt and proceed (privileged)	HPR	99	SI	3.6
Insert character	IC	43	RX	4.2
Insert storage key (privileged)	ISK**	09	RR	4.2
Load	LR	18	RR	3.0
Load	L	58	RX	4.8
Load address	LA	41	RX	4.2
Load and test	LTR	12	RR	3.0
Load control storage (privileged)	LCS	B1	RS	5.4 + 24.0w + 4.8s8
Load half word	LH	48	RX	5.4
Load multiple	LM	98	RS	3.0 + 1.8gr
Load PSW privileged	LPSW	82	SI	11.4
Move	MVC	D2	SS	7.6 + 0.6n + 0.6t4 (n – 1)
Move	MVI	92	SI	4.8
Move numerics	MVN	D1	SS	10.2 + 2.1n
Move with offset	MVO	F1	SS	10.2 + 1.2n1 + 1.2n2
Move zones	MVZ	D3	SS	10.2 + 2.1n
Multiply	M	5C	RX	39.6 + 0.6s1 + 0.6s2 + 0.6rn
Multiply decimal	MP	FC	SS	36.4 + 0.75n1 + 14.4(n1 – n2) + 0.375n2
OR	OR	16	RR	3.0
OR	O	56	RX	5.4
OR	OI	96	SI	6.0
OR	OC	D6	SS	10.2 + 1.5n
Pack	PACK	F2	SS	12.0 + 1.2(n1 – 1) + 1.2(n2 – 1)
Service timer register (privileged)	STR	03	RR	6.0 + 0.6t3
Set program mask	SPM	04	RR	3.0
Set storage key (privileged)	SSK**	08	RR	4.2
Set system mask (privileged)	SSM	80	SI	4.8
Shift left single logical	SLL	89	RS	5.4 + 0.6p + 0.6q
Shift right single logical	SRL	88	RS	5.4 + 0.6p + 0.6q
SOFTSCOPE forward scan (privileged)	SSFS	A2	RS	7.2
SOFTSCOPE reverse scan (privileged)	SSRS	A3	RS	7.2

*See Table B–4.
**Storage protect feature.

Table B–1. Basic Instructions (Part 3 of 3)

Instruction	Mnemonic Code	Operation Code	Type	Time in Microseconds*
Start I/O (privileged)	SIO	9C	SI	
IDA				10.2 to 12.6
IPC paper peripheral				15 to 59
Multiplexer				16.8 to 18.6†
Selector				25.2 to 26.4†
Store	ST	50	RX	5.4
Store character	STC	42	RX	4.8
Store half word	STH	40	RX	4.8
Store multiple	STM	90	RS	4.2 + 1.2gr
Subtract	SR	1B	RR	3.0
Subtract	SR (360/20)	1B	RR	3.6
Subtract	S	5B	RX	5.4
Subtract decimal	SP	FB	SS	36.6 + 0.75n1 + 0.375n2 + 6t1 + 3.0s6
Subtract half word	SH	4B	RX	5.4
Subtract half word	SH (9300 only)	AB	RX	5.4
Supervisor call	SVC	0A	RR	15.0
Supervisor load multiple (privileged)	SLM	B8	RS	4.2 + 1.8gr
Supervisor store multiple (privileged)	SSTM	B0	RS	4.2 + 1.2gr
Test under mask	TM	91	SI	6.0
Translate	TR	DD	SS	7.2 + 2.4n
Translate and test	TRT	DD	SS	8.4 + 1.8b
Unpack	UNPK	F3	SS	12.0 + 1.2(n1 – 1) + 1.2(n2 – 1)
Zero and add	ZAP	F8	SS	16.2 + 1.8n7 + 1.2n8 + 1.8t2(n2 – n1)

*See Table B–4.
†Realistic times, not maximum.

Table B–2. Extended Instructions (Part 1 of 2)

Instruction	Mnemonic Code	Operation Code	Type	Time in Microseconds*
Add logical	ALR	1E	RR	3.0
Add logical	AL	5E	RX	5.4
Add normalized (long)	ADR	2A	RR	$16.2 + 1.2ce + 1.2pr + 1.2t_1 + 1.2rp$
Add normalized (long)	AD	6A	RX	$19.2 + 1.2ce + 1.2pr + 1.2t_1 + 1.2rp$
Add normalized (short)	AER	3A	RR	$14.4 + 1.2ce + 1.2pr + 1.2t_1 + 1.2rp$
Add normalized (short)	AE	7A	RX	$16.8 + 1.2ce + 1.2pr + 1.2t_1 + 1.2rp$
Add unnormalized (long)	AWR	2E	RR	16.2 + 1.2ce + 0.6rp
Add unnormalized (long)	AW	6E	RX	19.2 + 1.2ce + 0.6rp
Add unnormalized (short)	AUR	3E	RR	14.4 + 1.2ce + 0.6rp
Add unnormalized (short)	AU	7E	RX	16.8 + 1.2ce + 0.6rp
Branch on index high	BXH	86	RS	7.2 – 1.2s3
Branch on index low or equal	BXLE	87	RS	7.2 – 1.2s3
Compare (long)	CDR	29	RR	18.0 + 1.2ce
Compare (long)	CD	69	RX	21.6 + 1.2ce
Compare (short)	CER	39	RR	15.6 + 1.2ce
Compare (short)	CE	79	RX	18.0 + 1.2ce

*See Table B–4.

Table B–2. Extended Instructions (Part 2 of 2)

Instruction	Mnemonic Code	Operation Code	Type	Time in Microseconds*
Divide	DR	1D	RR	64.8 + 1.2s1
Divide (long)	DDR	2D	RR	205.2 + 0.6p1 + 0.6p2 + 15.0pn + 0.6rn
Divide (long)	DD	6D	RX	208.2 + 0.6p1 + 0.6p2 + 15.0pn + 0.6rn
Divide (short)	DER	3D	RR	45.0 + 0.6p1 + 0.6p2 + 6.6pn + 0.6rn
Divide (short)	DE	7D	RX	47.4 + 0.6p1 + 0.6p2 + 6.6pn + 0.6rn
Edit and mark	EDMK	DF	SS	9.0 + 3.0n + 1.2n3 + 3.0n4 + 1.2n6
Halve (long)	HDR	24	RR	7.8 + 1.2pr + 0.6pn + 0.6(s2)
Halve (short)	HER	34	RR	7.2 + 1.2pr + 0.6pn
Load and test (long)	LTDR	22	RR	4.8
Load and test (short)	LTER	32	RR	4.2
Load complement	LCR	13	RR	3.0
Load complement (long)	LCDR	23	RR	4.8
Load complement (short)	LCER	33	RR	4.2
Load (long)	LDR	28	RR	4.2
Load (long)	LD	68	RX	6.6
Load negative	LNR	11	RR	4.2
Load negative (long)	LNDR	21	RR	4.2
Load negative (short)	LNER	31	RR	3.6
Load positive	LPR	10	RR	4.2
Load positive (long)	LPDR	20	RR	4.2
Load positive (short)	LPER	30	RR	3.6
Load (short)	LER	38	RR	3.6
Load (short)	LE	78	RX	5.4
Multiply	MR	1C	RR	39.0 + 0.6s1 + 0.6s2 + 0.6rn
Multiply half word	MH	4C	RX	24.0 + 0.6s1 + 1.8s2 + 0.6rn
Multiply (long)	MDR	2C	RR	115.2 + 0.6p1 + 0.6p2 + 1.2pn + 0.6rn
Multiply (long)	MD	6C	RX	118.2 + 0.6p1 + 0.6p2 + 1.2pn + 0.6rn
Multiply (short)	MER	3C	RR	39.0 + 0.6p1 + 0.6p2 + 0.6pn + 0.6rn
Multiply (short)	ME	7C	RX	41.4 + 0.6p1 + 0.6p2 + 0.6pn + 0.6rn
Shift left double logical	SLDL	8D	RS	4.8 + 1.2p + 1.2q
Shift left double	SLDA	8F	RS	7.8 + 1.2p + 1.2q
Shift left single	SLA	8B	RS	7.2 + 0.6p + 0.6q
Shift right double logical	SRDL	8C	RS	4.8 + 1.2p + 1.2q
Shift right double	SRDA	8E	RS	6.0 + 1.2p + 1.2q
Shift right single	SRA	8A	RS	5.4 + 0.6p + 0.6q
Store (long)	STD	60	RX	7.2
Store (short)	STE	70	RX	6.0
Subtract logical	SLR	1F	RR	3.0
Subtract logical	SL	5F	RX	5.4
Subtract normalized (long)	SDR	2B	RR	16.2 + 1.2ce + 1.2pr + 1.2t1 + 1.2rp
Subtract normalized (long)	SD	6B	RX	19.2 + 1.2ce + 1.2pr + 1.2t1 + 1.2rp
Subtract normalized (short)	SER	3B	RR	14.4 + 1.2ce + 1.2pr + 1.2t1 + 1.2rp
Subtract normalized (short)	SE	7B	RX	16.8 + 1.2ce + 1.2pr + 1.2t1 + 1.2rp
Subtract unnormalized (long)	SWR	2F	RR	16.2 + 1.2ce + 0.6rp
Subtract unnormalized (long)	SW	6F	RX	19.2 + 1.2ce + 0.6rp
Subtract unnormalized (short)	SUR	3F	RR	14.4 + 1.2ce – 0.6a
Subtract unnormalized (short)	SU	7E	RX	16.8 + 1.2ce – 0.6a
Test and set	TS	93	SI	6.0

*See Table B–4.

Table B–3. Instruction Execution Timing Assumptions

1.	Times are given from start of staticize of one instruction to the start of staticize of the next instruction, unless otherwise specified.
2.	No allowance is included for performance degradation due to the refresh cycles required by the storage units. This typically reduces performance by approximately 1 %.
3.	The time to perform single indexing of each operand address is included in the times given. Double indexing on RX instructions is not included. For each double indexing operation, add 0.6 μs to the execution time.
4.	Storage relocation is included in the specified times.
5.	Execution times apply to native mode operation. The times also apply to the 9200/9300 and 360/20 compatibility modes unless otherwise specified.
6.	Instructions may start on even or odd half-word boundaries with equal probability.
7.	Operands for MVN, MVZ, NC, OC, ZC are processed a half word at a time, unless the first operand address is greater than the second operand address by 1, in which case the operands are processed a byte at a time. When processed a byte at a time, the execution time, dependent on the length of the operands (i.e., n), doubles.
8.	On decimal multiply instructions, a digit of the multiplier (op1) may be any decimal digit with equal probability. The average multiplication time is used in the equation.
9.	On decimal divide instructions, a given digit of the quotient may be any decimal digit with equal probability. The average time to produce a digit of the quotient is used in the equation.
10.	Operands in main storage for AP, SP, CP, MP, and DP may be located on byte boundaries or half-word boundaries with equal probability.
11.	Decimal add and subtract assume a first operand more than eight bytes long and a positive result.
12.	The execution time given for SSFS is from the start of staticize to the point where the sync condition and the first softscope bus sample (after sync) is stored. Static presence of the sync condition, an initial sweep delay of zero, and a scan rate of zero are assumed. Additional time for completion of the instruction depends on the scan rate of nonzero, and the number of bytes to be stored. When the initial sweep delay is nonzero, this time also contributes to the overall execution time.
13.	The execution time given for SSRS is from the start of staticize to the point where the first softscope bus sample is taken. An initial sweep delay and scan rate of zero are assumed. When the initial sweep delay is nonzero, an additional 0.6 μs should be added to the time listed, as well as the specified initial sweep delay value, to determine the time to the first sample. When the initial sweep delay is zero and the scan rate is nonzero, an additional 0.6 μs should be added to the time listed, as well as the specified scan rate value, to determine the time to the first sample. The completion of the remainder of the instruction depends on the specified scan rate, the byte count, and the detection of a sync condition on the softscope bus.
14.	The execution time listed for the LCS instruction corresponds to the case for cc=0 (i.e., no sentinel is detected). For the cases when a sentinel is detected (cc = 1,3), an additional 4.8 μs should be added to the execution time.
15.	Decimal multiply timing assumes an average of 12 cycles per digit, which equals 14.4 μs per multiplier byte.
16.	SS operand fetch or store timing components assume 4-byte operands averaged over byte and half-word addressing routines.
17.	Decimal divide timing assumes an average of 20.5 cycles per quotient digit, which equals 26.6 μs per byte of quotient.

Table B–4. Legend for Instruction Execution Timings (Part 1 of 2)

a	A 1, if overflow adjustment is necessary; otherwise, 0.
b	Number of bytes in the first operand which are processed.
ce	Number of digit shifts required to equalize the characteristics.
d1	Number of zero addresses in switch list.
d2	A 1 if initial r odd general register has a nonzero value; otherwise, 0.
d3	A 1 if sentinel found; otherwise, zero.
d4	Number of task control blocks scrutinized.
d5	Number of linked task control blocks scrutinized.
d6	A 1 when exclusive search is specified; otherwise, 0.
d7	A 1 when a match is found; otherwise, 0.
d8	Number of control blocks with absolute wait bits set.
d9	Number of control blocks with wait bits set and ICOR bit clear.
d10	A 1 if ICOR = 1; otherwise, 0.
d11	A 1 if ICOR = 0 and no wait bits set; otherwise, 0.
e	Execution time of the subject instruction of an execute instruction.
gr	Number of general registers loaded or stored.
n	Number of bytes in the first operand (for instructions with a single field length).
n1	Number of bytes in the first operand.
n2	Number of bytes in the second operand.
n3	Number of field separator characters in the pattern.
n4	Number of digit select or significance starter characters in the pattern.
n6	Number of significant digits encountered when the significance indicator is not set before the digit is examined.
n7	Lowest number of bytes specified by l1 or l2.
n8	Number of bytes l1 exceeds l2 = 0 if $l1 \leqslant l2$.
nrr	A 1 if subject instruction of an execute instruction is not an RR type; otherwise, 0.
p	Number of 4-place shifts.
p1	Number of digit shifts required to prenormalize op1.
p2	Number of digit shifts required to prenormalize op2.

Table B–4. Legend for Instruction Execution Timings (Part 2 of 2)

pn	A 1 if the result requires postnormalization; otherwise, 0.
pr	Number of digit shifts required to postnormalize result.
q	Number of 1-place shifts.
r	A 1 if r1 ≠ 0; otherwise, 0.
rn	A 1 if the result (product or quotient) is negative.
rp	A 1 if recomplementing without postnormalization is required; otherwise, 0.
s	A 1 if branch is successful; otherwise, 0.
s1	A 1 if the sign of op1 is negative; otherwise, 0.
s2	A 1 if the sign of op2 is negative; otherwise, 0.
s3	A 1 if the sum of the first and third operand is equal to the comparand; otherwise, 0.
s4	A 1 if the result is greater than one word (eight decimal digits); otherwise, 0.
s5	A 1 if the signs of op1 and op2 are the same; otherwise, 0.
s6	A 1 if the signs of op1 and op2 are different; otherwise, 0.
s8	A 1 if sentinel detected; otherwise, 0.
t1	A 1 if the result is recomplemented; otherwise, 0.
t2	A 1 if n2 > n1; otherwise, 0.
t3	A 1 if timer stored; otherwise, 0.
t4	A 1 if one operand address is even and the other is odd; otherwise, 0.
w	Number of control storage words loaded.
w1	Number of channel status words.
y	A zero for byte count = 0; a 1 for byte count ≠0.
z	Number of half words in sum.